ViewDS Directory:
Installation and Operation Guide

Published: May 2025
Version: 7.5.2

ViewDS Directory: Installation and Operation Guide

For ViewDS Directory release 7.5.2

May 2025

Document Lifecycle

ViewDS may occasionally update documentation between software releases. Therefore, please visit www.viewds.com to ensure you have the PDF with most recent publication date. The site also hosts the most recent version of this document in HTML format.

Contents

About this guide

This guide includes an overview of ViewDS Directory, instructions for installing and configuring, and instructions for day-to-day operational tasks. It also includes ‘key concepts’ chapters that provide the background theory required to work with different aspects of ViewDS Directory.

This section includes:

- Who should read this guide
- Related documents
- How this guide is organized

Who should read this guide

Read this guide if you are responsible for installing or administrating ViewDS Directory.

It provides the following:

- an overview of ViewDS Directory, its components and how they work together
- how to install and configure ViewDS Directory
- how to install and explore an example directory
- an overview of the day-to-day tasks including how to start, stop, and monitor operations
- an overview of ViewDS Directory concepts
- an overview of adapting ViewDS Directory to your requirements

NOTE: You will need some familiarity with the host operating system (Linux or Windows).

Related documents

The other documents relating ViewDS Directory are:

- ViewDS Directory Server: Technical Reference Guide
- ViewDS Access Presence: Technical Reference Guide
- ViewDS Management Agent In-application Help
- ViewDS Access Proxy: Installation Guide
- ViewDS Access Sentinel: Installation and Reference Guide

How this guide is organized

This guide contains the following:

About this guide
Provides an overview of this guide.

System overview
Provides an overview of the system and an introduction to the processes, utilities and files in a ViewDS installation.

Installing ViewDS
Provides instructions for installing ViewDS Directory.

Configuring ViewDS
Provides instructions for configuring ViewDS Directory.

Exploring ViewDS
Provides instructions for exploring the example directory supplied with ViewDS Directory.

Operating the Directory
Provides instructions for day-to-day operations including how to start and stop ViewDS Directory, and how to backup and restore.

Key concepts: Schema
Introduces the concepts required to work with schema, how to manage schema through the ViewDS Management Agent, and includes high-level guidance to help you adapt ViewDS Directory to your requirements.

Key concepts: Security
Introduces the concepts required to work with security, how to manage security through the ViewDS Management Agent, and includes high-level guidance to help you adapt ViewDS Directory to your requirements.

Key concepts: XACML
Introduces the concepts required to work with XACML access controls, and includes how to manage XACML security through the ViewDS Management Agent.

Key concepts: Distribution and replication
Introduces the concepts required to work with distribution and replication, and how to manage distribution and replication through the ViewDS Management Agent.

System overview

This section provides an overview of the major features and components of ViewDS Directory, which is required before installing and configuring. Topics covered include: What is ViewDS Directory, and ViewDS Directory components.

What is ViewDS Directory

ViewDS Directory is a standards-based directory service. It delivers superior capabilities in both traditional (white pages) and emerging (XML and B2B) technologies, and provides high-performance complex searching on object-oriented hierarchical data.

By complying with X.500, LDAP and XML Enabled Directory (XED) standards, ViewDS Directory provides a rich set of features offering many benefits for directory users and administrators. The major features are outlined in the following topics:

- Improved user support
- Data represents your real-world hierarchy
- XACML integration
- Flexible data search
- PKI matching rules
- Windows-based management tool
- Certificate look-up service
- Reliability with scalability
- Integrates with your existing technology

Improved user support

ViewDS makes users' access to a directory easier by providing a web-based client and by supporting advanced approximate matching.

Web-based client

The web-based client, Access Presence, reduces implementation time and support by providing functionality that can be customized to your requirements.

This functionality includes:

- Organization charts
- Self-service portals

- Reporting interfaces
- Certificate management
- Chinese language approximate matching

Approximate matching

Users are often imprecise when searching – they might, for example, misspell or mistype names, or use acronyms or abbreviations.

ViewDS Directory delivers superior approximate matching that supports a range of strategies and provides better service to users:

- phonetic matching – for example, the search criteria ‘pane’ would match ‘payne’
- typing correction – ‘Dircetor’ would match 'Director'
- stem matching – ‘optics’ would match ‘optical’
- synonym matching – 'Bob' would match 'Robert', and 'road' would match 'street'
- abbreviation matching – 'NSW' would match 'New South Wales'

Additionally, specialized indexes are used to rapidly evaluate approximate matches.

Data represents your real-world hierarchy

ViewDS Directory minimizes the time, resources and planning required to make large-scale changes to a Directory Information Tree (DIT).

Directory Information Tree

Directory entries are arranged in a hierarchy called the DIT. A directory is of most use when its DIT mirrors a hierarchy in the real-world, such as the structure of a company, health trust or government department.

Real world hierarchies, however, tend to change very frequently – for example, departments can be formed, dissolved, moved, merged or split. This can lead directory administrators to flattening and simplifying the DIT to avoid the complexity of maintaining it. Unfortunately, the result can be little more than a list of entries, rather than a resource that helps people understand the structure of their organization.

ViewDS overcomes this by providing seamless support for changes to complex hierarchies. Directory administrators can change a DIT simply and rapidly so that it always accurately reflects the real-world structure of their organization.

Move and rename

Unlike many directories, ViewDS allows DIT entries with subordinates (non-leaf entries) to be moved or renamed, while looking after all the associated links such as ‘managed by’ and ‘manager of’ relationships. ViewDS can be configured to enforce constraints on the referential integrity of the DIT.

XACML integration

ViewDS Directory leverages its XML capabilities to allow XACML policy to be defined that conforms to XACML Version 3.0. This standard describes a language for expressing access controls and a protocol for requesting access-control decisions, both expressed in XML.

ViewDS includes the XACML Access Control scheme, which allows XACML policy to be applied to the directory.

ViewDS Access Sentinel extends the capability to allow XACML policy to be applied to applications external to ViewDS Directory. This allows unified access control across disparate applications and data sources, improving the user experience and reducing administration and support. For more information about Access Sentinel see the *ViewDS Access Sentinel: Installation and Reference Guide*.

Flexible data search

The component-matching rules allow users to search specific parts of an attribute with a complex syntax – for example, digital certificates, XML documents and certificate-revocation lists.

To illustrate, consider a Human Resources department that stores employees' resumes as XML documents. Component matching would allow a specific area of the resumes to be searched individually, and therefore, efficiently. For example, a user might search on just the 'qualifications' component of the resumes to find employees with a qualification in nuclear physics.

Without component matching, an application developer would generally need to either scan an entire directory or filter search results in order to find the required data. Both options are inefficient and slow.

PKI matching rules

X.509 certificates can be stored by most directories, but few support matching rules for the X.509 Public Key Infrastructure (PKI) attribute syntaxes. Consequently, environments with large certificate-revocation lists can suffer unacceptable performance.

ViewDS Directory supports the PKI matching rules, and also supports component matching for the PKI syntaxes. A PKI application that uses ViewDS will process certificates faster and with less effort. ViewDS also supports strong authentication with X.509 certificates for client-to-server and server-to-server authentication.

Windows-based management tool

The ViewDS Management Agent is a Windows-based application that makes managing local and remote ViewDS servers simpler and more efficient.

Certificate lookup service

ViewDS Access Proxy provides a certificate lookup service that complies with TSCP specifications. For more information about Access Proxy, see the *ViewDS Access Proxy: Installation Guide*.

Reliability with scalability

ViewDS Directory is designed to accommodate 'mission-critical' directory applications and is designed for continuous operation. During routine maintenance – such as backing up the database and check-pointing update logs – ViewDS Directory continues to process queries.

ViewDS Directory is designed to be scalable:

- A single server can accommodate many millions of entries.
- There are no restrictions on the number of entries, size or number of attributes, depth of the DIT or number of connected users (other than those imposed by the host's operating system and hardware).
- Includes an optimized tool for fast loading of entries. During bulk loading, the difference between the load-speed for the first and last entries loaded is negligible.
- Restarts rapidly after a power failure, irrespective of the number of entries.

Integrates with your existing technology

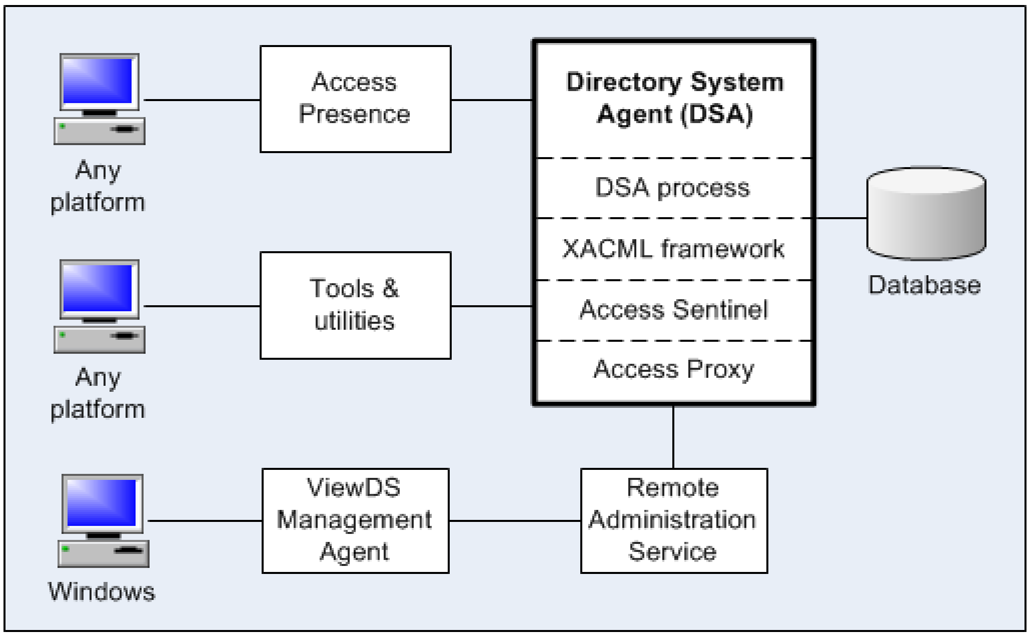
The standards-based architecture ensures seamless communication with third-party directory clients and servers:

- Implements OSI Stack (including RFC1006 over TCP/IP) and X.500 Internet Directly Mapped Protocol (IDMP).
- Supports LDAP, XLDAP, IDMP, XIDMP, SPML, SNMP and HTTP access.

Additionally, ViewDS components communicate using an efficient proprietary protocol running over TCP/IP.

ViewDS Directory components

The following illustration shows the main components.



The following topics provide an overview of these components:

- DSA process
- XACML framework
- Access Sentinel
- Access Proxy
- Access Presence
- Tools and utilities
- ViewDS Management Agent
- Remote Administration Service
- SNMP Proxy Agent

DSA process

The DSA process is a multi-threaded process comprising the main dsa thread and one or more dot threads.

The dsa (Directory System Agent) thread handles all communications with client applications and peer servers. It initiates and controls the state of the dot threads, queues and coordinates the processing of requests, and can manage the state of the database.

The dot (Directory Operation Thread) threads receive and process requests from the dsa. The dot threads will retrieve or update data from the database in order to satisfy requests from client applications. They also provide ViewDS's flexible searching capabilities. The number of dot threads running can be configured.

XACML framework

The XACML framework conforms to the XACML Version 3.0 standard and facilitates the XACML Access Control scheme. This scheme allows you to define XACML policy that imposes fine-grained access control on the directory.

For information about the XACML framework and writing XACML policy, see Key concepts: XACML.

Access Sentinel

ViewDS Access Sentinel extends the XACML framework to allow you to apply XACML policy to applications external to ViewDS Directory. It also includes an additional user interface, the Authorization Policy Manager, which allows XACML policy to be managed from any platform.

For more information about Access Sentinel, see the *ViewDS Access Sentinel: Installation and Reference Guide*.

NOTE: Access Sentinel requires additional licensing.

Access Proxy

ViewDS Access Proxy provides a certificate lookup service that complies with TSCP specifications. For more information about Access Proxy, see the *ViewDS Access Proxy: Installation Guide*.

NOTE: Access Proxy requires additional licensing.

Access Presence

Access Presence (webdua.cgi) operates with a standard web server and allows users to access a ViewDS directory through their web browser.

You can configure many aspects of Access Presence on the client and server side. The ViewDS Management Agent provides access to the server-side configuration, including specification of how entries and their attributes are displayed.

Tools and utilities

ViewDS Directory includes the following command-line tools and utilities:

- Stream DUA
- DSA Controller
- Printing DUA
- ViewDS tools

Stream DUA

The Stream DUA is a tool for managing a directory when the ViewDS Management Agent is unavailable. It is a batch-oriented Directory User Agent (DUA) for accessing directory data and managing backups, schema and knowledge. It is a text-oriented DUA that accepts input commands and data in a stream form and can therefore run non-interactively.

DSA Controller

The DSA Controller is a single process, the dsac process, and an important tool for controlling a DSA when the ViewDS Management Agent is unavailable. It runs on the same host as the DSA, and allows you to query and modify the DSA's status. This tool should only be made available to the ViewDS system administrator.

Printing DUA

The Printing DUA extracts data and prepares it for other applications. It can sort data, tag data items, insert text and other formatting information. The resulting data can then be used, for example, with a desktop publishing package to produce a printed directory listing.

ViewDS tools

As well as the ViewDS Management Agent, there are several command-line tools for backup, communications, and data loading – some are shell scripts, others are stand-alone programs:

- `dbbackup` – this script performs an incremental or full backup of a directory to a tape device (Linux only).
- `smerge` – this script sorts and merges the entries in the update log files so that a database can be restored from backup files without losing recent updates.
- `vfload` – this is the ViewDS Fast Load utility, which allows dump files to be loaded into the directory.

The utilities are described fully in the *ViewDS Directory Server: Technical Reference Guide*.

ViewDS Management Agent

The ViewDS Management Agent is a Windows-based application that runs either locally or on a different computer to the DSA's host. It allows you to manage the status of one or more DSAs remotely and access their configuration parameters, log files, directory data, schema, knowledge and access controls.

To ensure secure administration, SSL/TLS connections and certificate-based authentication is used between the ViewDS Management Agent and DSA, and between the ViewDS Management Agent and Remote Administration Service (RAS).

Remote Administration Service

The Remote Administration Service (RAS) runs on the DSA’s host computer. It allows the ViewDS Management Agent, installed on a remote computer, to start and stop the DSA, modify its configuration and view its log files.

SNMP Proxy Agent

The SNMP Proxy Agent collects MIB (management information base) objects from multiple DSAs and presents them as a single MIB object to an SNMP management console. This allows a distributed DSA environment to appear as a single DSA to the SNMP manager.

Installing ViewDS

This section describes how to install ViewDS Directory. The installation includes the server (DSA and RAS) and ViewDS Management Agent, plus other components that require additional licensing (including Access Sentinel).

Upgrade from a previous version

To upgrade from a previous version:

1. Read Before you start.
2. Complete either Upgrade from an old version or Upgrading from View500.

Installation scenarios

ViewDS Directory can be installed in several ways according to your requirements. This subsection includes the procedures for the following common deployment scenarios:

- Install all components on a single Windows server
- Install ViewDS server and Access Presence on Linux with remote VMA
- Install ViewDS server and Access Presence on Windows with remote VMA
- Install ViewDS server on Linux with remote Access Presence and VMA

Follow one of the scenarios as it appears below, adapt one to suit your needs, or contact your reseller for advice about your installation requirements.

NOTE: Before following any of the installation scenarios, read Before you start.

Install all components on single Windows server

In this scenario all ViewDS Directory components are installed on the same Windows server.

- Install all components on Windows

Install ViewDS server and Access Presence on Linux with remote VMA

To implement this scenario:

1. Install the ViewDS server on a Linux host
2. Install the ViewDS Management Agent on Windows

3. Install credentials
4. Connect the VMA to the ViewDS server
5. Enter licence information

NOTE: Additionally, if you wish to use Access Presence you must configure an Apache Server on the Linux host.

Install ViewDS server and Access Presence on Windows with remote VMA

In this scenario the ViewDS server and Access Presence are installed on one Windows server and the VMA is installed on another. This type of deployment would be suitable if you want to install the VMA on an administrator's desktop PC, allowing them to remotely manage multiple ViewDS server instances across a number of platforms:

1. Install ViewDS server and Access Presence on Windows
2. Install the ViewDS Management Agent on Windows
3. Install credentials
4. Connect the VMA to the ViewDS server
5. Enter licence information

NOTE: Additionally, if you wish to use Access Presence you must configure an Apache HTTP Server or Microsoft IIS on the Windows host.

Install ViewDS server on Linux with remote Access Presence and VMA

In this scenario the ViewDS server is installed on a Linux server, Access Presence is installed on one Windows server and the VMA is installed on another. This type of deployment might be suitable for an internet facing Access Presence instance, where the Access Presence machine is in the DMZ but the ViewDS server machine is within the internal network:

1. Install the ViewDS server on Linux
2. Install Access Presence (along with the ViewDS server) on Windows
3. Configure Access Presence on a remote host
4. Install the ViewDS Management Agent on Windows
5. Install credentials
6. Connect the VMA to the ViewDS server
7. Enter licence information

Before you start

This subsection describes requirements for the following:

- License requirements
- ViewDS server requirements
- Access Presence requirements
- ViewDS Management Agent requirements

Along with the above components, the following are also installed according to your licence: Access Presence, Access Sentinel, Stream DUA and other tools. A ViewDS server installation also includes a demonstration directory, Deltawing.

NOTE: Before starting, check the ViewDS Release Notes for any known issues.

License requirements

The ViewDS Directory license key has several components that enable each of the following:

- ViewDS server DSA
- Access Sentinel
- Access Proxy

Obtaining a license

To obtain a license key, supply your ViewDS vendor with host-specific information for the computer on which ViewDS will be installed. Your vendor will return a license key that you will need during installation.

Platform	Command	Returns host-specific information
Linux	ifconfig –a	Hardware address of any Ethernet interfaces reported by this command. This address is usually represented as a colon separated list of six hexadecimal fields.
Windows	ipconfig /all	Physical address of the Ethernet adaptor reported by this command. This address is usually six hexadecimal fields, each separated by a dash (e.g. 00-22-BA-7D-92-DE).

Upgrading from ViewDS 7.3 or earlier

Due to changes in the format of the ViewDS license key made in ViewDS 7.4, you will not be able to use your existing license key when upgrading from ViewDS 7.3 or earlier. Instead you must contact your ViewDS vendor with your old license in order to obtain a replacement key. Your old license key is stored in the configuration file located by default in either:

- `${VFHOME}/setup/config` (Linux)
- `%VFHOME%\setup\config` (Windows)

Where `${VFHOME}` or `%VFHOME%` is the ViewDS install directory.

ViewDS server requirements

This subsection describes the following requirements for the ViewDS server:

- Platform requirements
- Memory requirements
- Disk space requirements

Platform requirements

To install ViewDS server, you need one of the following:

- 64-bit Windows 7; or 64-bit Windows Server 2008 R2 or above
- 32 or 64-bit Intel running RedHat Enterprise Linux 6 or later

Other platforms may also be suitable (contact your ViewDS vendor for more information).

Memory requirements

For optimum performance there should be sufficient real memory to run the DSA process suite, and to provide the DSA with a cache sufficient to hold all entries with subordinates (non-leaf entries) in memory.

Recommendation for a medium size directory

The size of processes varies with the processor type, and the software and directory configuration. A typical minimum for a medium size organization is 32 MB, with 64 MB preferred. Additional memory should also be allocated to the DSA disk cache.

If this cache is as large as the database, the entire database will reside in the cache and disk accesses minimized. This configuration gives optimum performance and is the recommended configuration for smaller databases.

Recommendation for a larger directory

For a very large database (millions of entries), the performance difference between having all non-leaf entries and indexes in cache and all entries and indexes in cache is too small to justify a large cache. The recommendation is for sufficient memory to hold only all non-leaf entries and indexes in cache, perhaps 10% of the database size.

Disk space requirements

The disk space required depends on the number of entries, the amount of data they hold, and the amount of indexing configured. There must be enough space to hold the database and the dumped database.

A typical minimum requirement is approximately 30 MB plus 2 KB for each entry. This space must be available after providing for operating system requirements including swap space. This implies actual disk capacities in the order of 300 MB for medium organizations and 500 MB for larger organizations.

Disk reliability and redundancy measures, such as implementing a RAID-1 or RAID-5 disk array, should also be considered.

Access Presence requirements

Access Presence (webdua.cgi) is a web-based client included in the installation of the ViewDS server. If you intend to implement Access Presence, a web server is required on either the same or a different host to the ViewDS server. The supported web servers are Apache HTTP Server and Microsoft Internet Information Services (IIS) Versions 6, 7 and 8.

NOTE: The installation of the ViewDS server includes Access Presence by default. When ViewDS server is running on one host and Access Presence on another, you do not need a licence key for the Access Presence host.

ViewDS Management Agent requirements

The ViewDS Management Agent (VMA) is a Windows-based application that allows you to manage local and remote ViewDS servers.

There are two installers available for the VMA:

- *ViewDS Suite installer*, which allows you to install one or more of the following on the same computer: VMA, ViewDS server, Access Presence.
  Default location: `/ProgramData/ViewDS Suite`
- *ViewDS VMA installer*, which allows you to install the VMA locally (interactive mode) or to a remote computer (unattended mode).
  Default location: `/ProgramData/ViewDS/Directory Server`

Installing or removing one package has no effect on the other.

Whichever installer is used, the VMA's installation requirements are the same.

Installation requirements

NOTE: The VMA is only compatible with ViewDS server version 7.5 or above.

The requirements are:

- 32/64-bit Windows version 7 or above; or 32/64-bit Windows Server 2008 R2 or above
- Microsoft .NET Framework Version 3.5 SP1

For a Windows 10 host:

- .NET Framework 3.5 SP1 is installed with Windows 10 by default, but must be enabled before installing the VMA. It can be enabled through the Windows Features area of the host's Control Panel, or by granting 'Install Windows Features' access to the host's user.

For a non Windows 10 host:

- If not already installed, the VMA installer will install and enable the .NET Framework 3.5 SP1 on the host.

Upgrading from an old version of ViewDS

To upgrade from ViewDS 7.4 or earlier:

- Dump database and back up
- Uninstall the existing version
- Install the new version
- Load the database

Note if upgrading from ViewDS 7.3 or earlier

If you are upgrading from ViewDS 7.3 or earlier, then the following notes apply.

WARNING: As a result of changes made to the way in which passwords are encrypted in dumps and logs in ViewDS Version 7.4, it is essential that you know the key used to encrypt passwords in the files that you are about to dump.

This key is available through the ViewDS Management Agent, on the **Configuration** tab under the **Runtime Settings** tab. If no key is included on the **Runtime Settings** pane, then the default ViewDS key `MePh2P` is used. You will need this key when loading the dumped files into ViewDS 7.5.

WARNING: As a result of changes to the format of the ViewDS license key made in ViewDS Version 7.4, you will not be able to use your existing license key when you upgrade from a previous version.

Before upgrading you must contact your ViewDS vendor with your old license in order to obtain a replacement key. See License requirements for further details.

Dump database and back up

To dump the database of your existing installation and back it up:

1. From the ViewDS Management Agent, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Graphical View** button.
4. Right-click the **Database** (DB) icon and then click **Dump**. The dump files are written to the dump directory.
5. Finally, set the logging level for the error log to **debug** to facilitate troubleshooting later:
   a. At the bottom of the left pane, click **Server View**.
   b. In left pane, click the server.

   c. In the right pane, click the **Configuration** tab and then click **Operational**. This screen includes Error-level parameter.
   d. In the value column for the **Error level**, select **debug**.
   e. At the bottom of the screen, click **Set**.
6. Check that the dump files have been created in the dump directory and back them up. The location of the dump directory is set by the configuration-file parameter `dumpdir`, which by default is `${VFHOME}/dump` (Linux) or `%VFHOME%\dump` (Windows). Where `${VFHOME}` or `%VFHOME%` is the install folder of the existing version of ViewDS.
7. Back up your installation directory by archiving it and its sub-directories (this includes the licence information stored in `${VFHOME}/setup/config`).
8. Perform the steps below to uninstall the existing version of ViewDS.

Uninstall the existing version

To uninstall the existing version:

1. From the ViewDS Management Agent's host PC, open the **Windows Control Panel** and uninstall the **ViewDS Management Agent**.
2. From the ViewDS server's host computer, uninstall the DSA and RAS by following the appropriate steps below.

Linux

1. Stop the RAS if it is running (also stops the DSA):
   ```
   ras stop
   ```
2. Delete the ViewDS installation directory and the ViewDS administration account.
3. Perform the steps below to install the new version of ViewDS.

Windows

1. From the command line, stop the RAS and remove it as a service:
   ```
   ras stop
   ras -u
   ```
2. Uninstall ViewDS from the Windows **Control Panel**. This removes all unmodified files in the `%VFHOME%` folder.
3. Perform the steps below to install the new version of ViewDS.

Install the new version

1. Follow the steps in one of the remaining ViewDS installation scenarios.
2. Confirm the DSA's configuration parameters (particularly the runtime parameters).
3. Perform the steps below to load the database.

Load the database

To load your database into the new installation:

1. Copy all the previously backed up files that are installation-specific to the appropriate locations in the new ViewDS installation. These files include, for example:
   - configuration file (by default, `${VFHOME}/setup/config`)
   - Access Presence files (by default, located in `${VFHOME}/webdir/`)
   - administration support scripts
   - certificates
2. Remove the following deprecated parameters (if they exist) from the configuration file (by default, `${VFHOME}/setup/config`):
   - `xentries`
   - `xexpiry`
   - `xhostid`
   - `xkeybase`
   - `xlicensekey`
   - `xlimit`
   - `xschema`
   - `xsearches`
   - `xdsa`
   - `xpdua`
   - `xsduasync`
   - `xwebdua`
   - `xenforce`
   - `xtscpproxy`
   - `xpdp`
3. Copy the dump files of your existing database into the new ViewDS dump directory. The location of the dump directory is set by the configuration-file parameter `dumpdir`, which by default is:
   - `${VFHOME}/dump` (Linux)
   - `%VFHOME%\dump` (Windows)
4. Enter the following on the command line to stop the RAS and the DSA:
   ```
   ras stop
   ```
5. To load the dumped database files:

If upgrading from...	Then enter this command
ViewDS 7.4	`vfload -m dump/dib.*`
ViewDS 7.3 or earlier	`vfload -K "encryption key" -m dump/dib.*` Where the `encryption key` is the key used to encrypt passwords in dump files in your old version of ViewDS (see the Note if upgrading from ViewDS 7.3 or earlier).

You may need to address any schema inconsistencies before the dump files can be loaded successfully. Details of any schema parsing problems will be reported in the error log (if the logging level is set to debug).

6. Finally to restart the RAS and the DSA use the following command:
   ```
   ras
   ```
7. Start the ViewDS Management Agent.
8. Click **Server View**.
9. In the left pane, click the server.
10. In the right pane, click the **Status** tab followed by the **Error Log** tab.
11. Review the contents of the error log.

Upgrading from View500

To upgrade from View500:

- Dump database and back up
- Uninstall View500 and install ViewDS
- Load the database

Dump database and back up

WARNING: As a result of changes made to the way in which passwords are encrypted in dumps and logs in ViewDS Version 7.4, it is essential that you know the key used to encrypt passwords in the files that you are about to dump.

This key is available in the file `VFHOME/general/deity` when the DSA is running. The key is the string that appears after the comma in this file. So, for example, the file might contain the following: `745403800,MePh2P` where `MePh2P` is the key.

As a result of changes to the format of the ViewDS license key made in ViewDS 7.4, you will not be able to use your existing license key when you upgrade from a previous version. Before upgrading you must contact your ViewDS vendor with your old license in order to obtain a replacement key. See License requirements for further details.

To dump the database of your existing installation and back it up:

1. Enter the following at the command line to stop the DSA and prevent any further database transactions:
   ```
   dsa stop
   ```
2. Dump the database:
   ```
   vfload -c dump
   ```

3. Check that the dump files have been created in the dump directory and back them up. The location of the dump directory is set by the configuration-file parameter dumpdir, which by default is:
   - `${VFHOME}/dump` (Linux)
   - `%VFHOME%\dump` (Windows)

   where `${VFHOME}` or `%VFHOME%` is the location of the previous version.
4. Back up your installation directory by archiving it and its sub-directories (this includes your licence information stored in `${VFHOME}/setup/config`).
5. Perform the steps below to uninstall the existing version and install ViewDS.

Uninstall View500 and install ViewDS

To uninstall View500 and install ViewDS:

1. From the ViewDS server's host, do one of the following:
   - For Windows, uninstall the DSA service (`dsa -u`) and uninstall View500 from the Windows **Control Panel**.
   - For Linux, stop the View500 processes.
2. If implemented, remove all references to the Web Admin interface from your web server. (This application is not supported in ViewDS.)
3. Perform the tasks outlined in one of the remaining ViewDS installation scenarios.
4. Confirm the DSA's configuration parameters, particularly the runtime parameters.
5. Perform the steps below to load the database.

Load the database

To load your database into the new installation:

1. Copy all the previously backed up files that are installation-specific to the appropriate locations in the new ViewDS installation.
   These files include, for example:
   - configuration file (by default, `${VFHOME}/setup/config`)
   - Access Presence files (by default, located in `${VFHOME}/webdir/`)
   - administration support scripts
   - certificates
2. Remove the following deprecated parameters (if they exist) from the configuration file (by default, `${VFHOME}/setup/config`):
   - `Webaddauxoctemplate`
   - `xentries`
   - `xexpiry`
   - `xhostid`
   - `xkeybase`

- `xlicensekey`
- `xlimit`
- `xschema`
- `xsearches`
- `xdsa`
- `xpdua`
- `xsduasync`
- `xwebdua`
- `xenforce`
- `xtscpproxy`
- `xpdp`

3. Copy the dump files of your existing database into the ViewDS dump directory. The location of the dump directory is set by the configuration-file parameter `dumpdir`, which by default is: `${VFHOME}/dump` (Linux) and `%VFHOME%\dump` (Windows).
4. Start the ViewDS Management Agent and set the logging level for the error log to **debug** to facilitate troubleshooting later:
   a. At the bottom of the left pane, click **Server View**.
   b. In left pane, click the server.
   c. In the right pane, click the **Configuration** tab and then click **Operational**. This screen includes Error-level parameter.
   d. In the value column for the **Error level**, select **debug**.
   e. At the bottom of the screen, click **Set**.
5. Enter the following on the command line to stop the RAS and the DSA:
   ```
   ras stop
   ```
6. Then enter this command to load the dumped database files:
   ```
   vfload -K "encryption key" -m dump/dib.*
   ```
   Where the `encryption key` is the key used to encrypt passwords in dump files in your old version of ViewDS. See the note at the beginning of Dump database and back up for details.

   You may need to address any schema inconsistencies before the dump files can be loaded successfully. Details of any schema parsing problems will be reported in the error log (if the logging level is set to **debug**).
7. Finally to restart the RAS and the DSA use the following command:
   ```
   ras
   ```
8. Restart the ViewDS Management Agent.
9. Click **Server View**.
10. In the left pane, click the server.
11. In the right pane, click the **Status** tab followed by the **Error Log** tab.
12. Review the contents of the error log.

Installing ViewDS components on Windows

The *ViewDS Suite installer* allows you to install one or more of the following ViewDS components on a Windows computer:

- ViewDS server
- Access Presence
- ViewDS Management Agent (VMA)

Depending on the components to be installed, the installer also:

- installs and configures all prerequisite software for the VMA and Access Presence
- installs temporary keys and the license key
- creates a connection between the ViewDS server and VMA

To install one or more components of the ViewDS suite on Windows:

1. Log on as a user with Administrator privileges.
2. Run `ViewDS Suite x64.exe` supplied on the ViewDS installation media. The ViewDS Suite Setup Wizard is displayed.
3. Click **Next**. The ViewDS Identity Solutions License Terms are displayed.
4. Read the license agreement and select the check the box to accept its terms, then click **Next**. The default file location for ViewDS is displayed.
5. Accept the default or select an alternative location, then click **Next**. The ViewDS components are displayed: ViewDS Server, ViewDS Access Presence, and ViewDS Management Agent (VMA).
6. Optionally, click to clear the checkbox for each component you do not want to install, then click **Next**. If you are installing the ViewDS server, then the License Key dialog is displayed and you should move to step 7 below. Otherwise, the Create application shortcuts dialog is displayed and you should move to step 8.
7. If you have a license key, paste the text of your license key into the box, then click **Next**. The Create application shortcuts dialog is displayed.

   NOTE: If you install ViewDS without a license key, then you will be able to launch the VMA but you will not be able to create a connection to the ViewDS server. To complete an installation in these circumstances, you should provide license key information as described in Entering licence information.

8. If you do not want to create a default shortcut, click to clear a checkbox. Otherwise, depending on which ViewDS components are being installed, shortcuts will be created. A shortcut will be created for the VMA on the desktop; and the Windows Start menu will include shortcuts to the VMA, the Access Presence page showing the ViewDS example Deltawing, and the ViewDS website. Click **Next**. The Begin Installation of ViewDS Suite dialog is displayed.

9. If you are installing the VMA, then you have the option to click to clear the **Install the default certificate** checkbox. Otherwise, the default PKI certificates will be installed.

   NOTE: Default certificates are required in order to create a connection between the ViewDS server and the VMA.

   NOTE: The ViewDS suite installer will install .NET Framework 3.5 and IIS (required by Access Presence) if supported versions are not currently installed or enabled.

10. Click **Install**.
11. If default certificates are being installed, then the Certificate Import Wizard opens in a separate window. Accept all the default settings (including the blank password box) by clicking the **Next** button until the wizard completes the import and closes.
12. If the ViewDS server is being installed, you have the option to click to clear **Configure Management Agent** to prevent the installer from configuring a connection between the ViewDS Management Agent (VMA) and ViewDS server.
13. If the VMA is being installed, you have the option to click to clear **Run Management Agent** to prevent the installer from launching the VMA.
14. Click **Close**. According to your selections in the above steps, the settings to connect the VMA to the ViewDS server are displayed, and the VMA starts.

Installing the ViewDS server on Linux

To install the ViewDS server with the default configuration:

1. Create a Linux account for the ViewDS System Administrator.

   The ViewDS processes on Linux run under a single account name (the ViewDS System Administrator), which should have an account with a login name indicative of the ViewDS application (for example, viewds).

   By default, the locations of the ViewDS directories are relative to this user's home account.

   The administrator does not need to be a super-user.
2. Log on as the ViewDS System Administrator and create directory where ViewDS will be installed (for example, `viewds`).
3. From the installation media, copy the appropriate tar file for your platform to the installation directory and un-tar it. The ViewDS directories and files are created in the current directory (one of the files is the installation script `vfinstall`).
4. Run the installation script by entering `vfinstall` at the command line. You will be prompted to confirm the installation location.

   When the installation completes, the required values for the following environment settings are displayed: `VFHOME`, `MANPATH`, `PATH` and `LD_LIBRARY_PATH`. (Note that Bash and Bourne shells use the same format.)

5. Update the `.profile` file with the environment settings displayed when the installation completed.
6. Apply the new settings by logging out and in again, or by entering the following to update the Bash profile: `. .bash_profile`
7. Enter `echo $VFHOME` to confirm that the environment settings have been applied.
8. Enter the following commands to start and configure the RAS:
   a. `ras` – starts the RAS.
   b. `ras status` – confirms whether the process has started.
   c. `ras add default` – defines a new service called 'default'.
   d. `ras default dsa configure` – configures the RAS service to run the DSA.
   e. `ras status` – reports that the DSA is not licensed and is not active or running.
9. Apply a license key:
   a. Enter `ifconfig -a` to obtain the system's MAC address.
   b. Provide the MAC address to your ViewDS vendor and request a license file.
   c. Rename the license file `license.xml` and copy it to the `$VFHOME/setup` directory.
   d. Confirm that the license has been applied by entering `dsa`. If the license has been applied, the message `dsa server has started` is displayed.
   e. Stop the DSA by entering `dsa stop`.
10. Enter `ras default dsa start`. The RAS and then DSA are started, which means that the DSA can be controlled by the ViewDS Management Agent.
11. Enter `ras status`. The response reports whether the DSA is licensed, active and running.

Installing the ViewDS Management Agent

The ViewDS Management Agent (VMA) is a Windows-based application that allows you to manage local and remote ViewDS servers.

There are two installers available for the VMA:

- *ViewDS Suite installer*, which allows you to install one or more of the following on the same computer: VMA, ViewDS server, Access Presence.
  Default location: `/ProgramData/ViewDS Suite`
- *ViewDS VMA installer*, which allows you to install the VMA locally (interactive mode) or to a remote computer (unattended mode).
  Default location: `/ProgramData/ViewDS/Directory Server`

Installing or removing one package has no effect on the other.

For more about the ViewDS Suite installer, see Installing ViewDS on Windows. For the installation requirements, see ViewDS Management Agent requirements.

This subsection describes how to install the VMA using the ViewDS VMA installer.

Installation modes

The ViewDS VMA installer has two modes: **interactive** and **unattended**. Unattended mode allows you to complete a remote installation through the likes of Active Directory GPO.

The main difference between the modes is that interactive mode installs default PKI certificates (see Installing credentials).

The ViewDS VMA installer also provides the option to generate an installation log.

Installing the VMA: Interactive mode

To install the VMA using interactive mode:

1. Log in as a user with Administrator privileges.
2. Run the ViewDS VMA installer supplied on the ViewDS installation media:
   ```
    VMA.msi
   ```
   Or to run the installer with an installation log, enter the following at the command line:
   ```
   " VMA.msi" /L*v install.log
   ```
   Where `install.log` is the name (and path if included) of the log file.
   The ViewDS Management Agent Setup wizard is displayed.
3. In the ViewDS Management Agent Setup wizard, click **Next**. The license agreement is displayed.
4. Read the license agreement and to accept its terms, click **Accept**. The default installation folder is displayed.
5. Accept the default ViewDS install folder or select an alternative, then click **Next**. A confirmation message is displayed.
6. Click **Install**.
7. The Certificate Import Wizard starts in a separate window. Accept all the default settings by clicking **Next** until the wizard completes the import and closes.

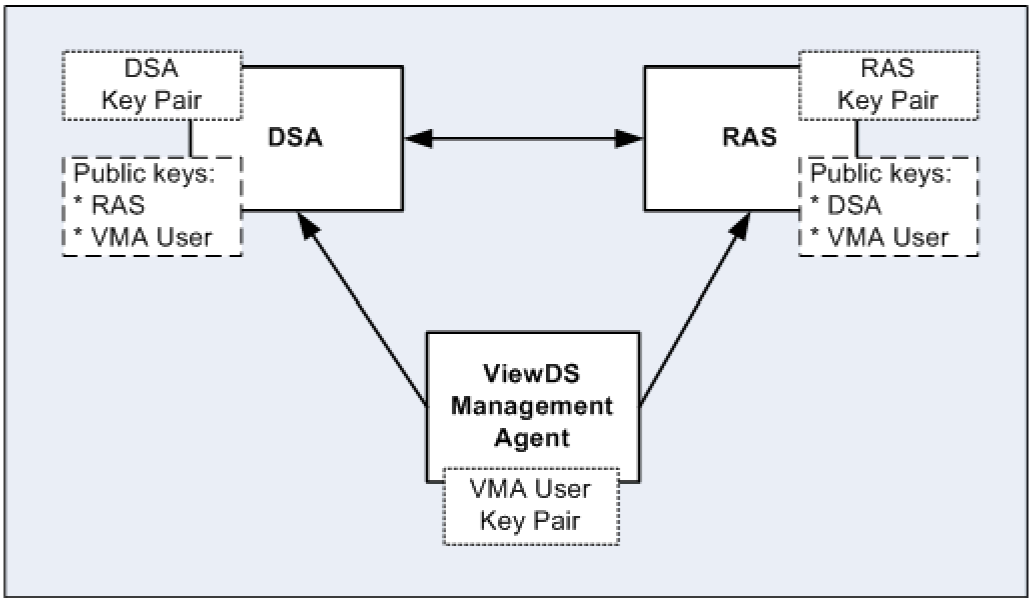
Installing the VMA: Unattended mode

To install the VMA using unattended mode:

1. Log in as a user with Administrator privileges.
2. Run the ViewDS VMA installer supplied on the ViewDS installation media:
   ```
   " VMA.msi" /qn
   ```
   Or to run the installer with an installation log:
   ```
   " VMA.msi" /qn /L*v install.log
   ```
   Where `install.log` is the name (and path if included) of the log file.
   The installer runs without displaying any prompts.

Installing credentials

ViewDS uses SSL/TLS connections and certificate-based authentication. This ensures secure communications between the ViewDS Management Agent and the ViewDS server (DSA and RAS) and between the DSA and RAS.



ViewDS is distributed and can be installed with a default key pair – public key and private key. However, it is strongly recommended that you obtain and install your own public–private key pairs for each DSA and RAS. (Every DSA in a distributed or replicated environment must have a certificate with a unique subject name.)

Each user of the ViewDS Management Agent must also have a key pair. Otherwise, the ViewDS Management Agent will not be able to connect to either the DSA or RAS.

Obtaining SSL server certificates

There may already be processes in place within your organization to generate certificates. If not, new SSL server certificates can be purchased online from a Certificate Authority or generated using an open source tool (such as OpenSSL). If required, please contact your ViewDS vendor for further advice.

Private keys for the DSA and RAS must be in either a PKCS#8 (BER) or PKCS#12 (DER) file.

Installing new key pairs

Installing new key pairs involves:

- Installing a key pair for a ViewDS Management Agent user
- Installing a new key pair for the DSA and RAS

Installing a key pair for a ViewDS Management Agent user

To install a public and private key pair from a PKCS#12 format file:

1. On the ViewDS Management Agent's host computer, install the user's keys:
   a. Copy the PKCS#12 file to the ViewDS Management Agent's host computer.
   b. Double-click the PKCS#12 file. A certificate window is displayed.
   c. Follow the instructions on the screen to import the certificate.
2. Export the public key certificate:
   a. Start the ViewDS Management Agent.
   b. From the **File** menu, click **New Connection**. The New Connection window is displayed.
   c. In the **Certificate for DSA** box, click the certificate that you have just installed.
   d. Click the **View** button. The Certificate window is displayed.
   e. Click the **Details** tab.
   f. In the **Field** column, click **Public Key** and then click the **Copy to File** button. The Certificate Export Wizard is displayed.
   g. Follow the instructions on the screen and export the public key to a DER file. When the export has completed, the Wizard closes but the New Connection window is still displayed.
   h. In the New Connection window, click **Cancel**.
3. On the DSA's host, copy the exported public-key certificate to the following location `${VFHOME}/setup/trusted` or `%VFHOME%\setup\trusted` (`VFHOME` is the ViewDS installation path).

   NOTE: This is the default location for public keys stored by both the DSA and RAS. Each location can be modified through settings on the File System tab of the ViewDS Management Agent: 'Directory of DSA administrator certificates' and 'Directory of remote administration service administrator certificates'.

4. Follow the steps below to connect to the ViewDS server.

Connecting to the ViewDS Server

1. From the main menu of the ViewDS Management Agent (VMA), click **File** followed by **New Connection**. The New Connection window is displayed.
2. In the New Connection window, complete the following details:
   - **Host** – enter the host name or IP address of the DSA's host.
   - **DSA port** – enter the port number to connect to on the host for the DSA (by default, 3000). The RAS port box is automatically populated (by default, 3018).
   - **Certificate for DSA** – select the certificate used by the VMA to connect to the DSA. The Certificate for RAS box is automatically populated with the same certificate.
   - **Connection name** – enter the label displayed by the VMA for this connection.

3. Click the **Connect** button. A message window is displayed to say that the DSA will not start without a licence key.
4. Click **OK**. A server icon is now displayed in the left pane.

Installing a new key pair for the DSA and RAS

To install a new key pair for the DSA and RAS:

1. From the ViewDS Management Agent, click the server icon in the left pane.

   > **NOTE:** Before installing a new key pair, the DSA should not be running. However, for a new installation, the DSA will not be running because you will not have entered its licensing information at this stage.

2. In the right pane, click the **Configuration** tab followed by the **File System** tab.
   The tab includes the following settings:
   - **DSA public key (certificate)** – identifies the name and location of the public key on the DSA's host. (This setting corresponds to dsacertificate in the DSA's configuration file.)
   - **DSA private key** – identifies the name and location of the private key on the DSA's host. (This setting corresponds to dsaprivkey in the configuration file.)
   - **DSA private password** – identifies the file containing the clear-text password required to decrypt the DSA's private key.
   - **Directory of remote administration service administrator certificates** – identifies the location for public keys used by the RAS. (This setting corresponds to rastrusted in the configuration file.)

   By default, the configuration file is `${VFHOME}/setup/config` or `%VFHOME%\setup\config`.
3. On the DSA's host, copy the new private key to the location set by DSA private key. By default, `${VFHOME}/setup` or `%VFHOME%\setup`.
   The file should be read-only to the owner of the DSA process.
4. Copy the new public key to the locations set by:
   - **DSA public key (certificate)** – by default, `${VFHOME}/setup` or `%VFHOME%\setup`
   - **Directory of remote administration service administrator certificates** – by default, `${VFHOME}/setup/trusted` or `%VFHOME%\setup\trusted`
5. If the private key is encrypted with a password, enter the password into the file identified by **DSA private password**. By default, `${VFHOME}/general/ keyaccess` or `%VFHOME%\general\keyaccess`.
6. From the ViewDS Management Agent, double-click the Value cell for the **DSA private key** setting. The cursor is displayed in the cell.
7. Modify the value to the file name of the new private key (also modify the path if required).
8. Set the values of **DSA public key (certificate)** and **Directory of remote administration service administrator certificates** to the file name of the new public key.

9. At the bottom of the screen, click **Set**.
10. To install a new key pair for the RAS, repeat this task from step 2 using the following settings:
    - **Remote administration service public key (certificate)** – identifies the name and location of the RAS’s public key. (This setting corresponds to rascertificate in the DSA’s configuration file.)
    - **Remote administration service private key** – identifies the name and location of the RAS’s private key. (This setting corresponds to rasprivkey in the configuration file.)
    - **Remote administration service private password** – identifies the file containing the clear-text password required to decrypt the RAS’s private key.
    - **Directory of DSA administrator certificates** – identifies the location of public keys used by the DSA. (This setting corresponds to dsatrusted in the configuration file.)
11. Perform the task Entering licence information.

Installing a temporary key pair

You can implement a working, albeit temporary and non-secure, system by:

- using the default key pairs supplied with the DSA and RAS; and
- using the same default key pair used by the RAS for a user of the VMA. A PKCS#12 file of the default RAS public and private key pair is provided for this purpose.

NOTE: This is a temporary measure. Using the key pairs distributed with ViewDS in a production environment is not recommended. This is because they are available to all customers of ViewDS and cannot be considered secure.

To install the RAS’s key pair for a user of the ViewDS Management Agent (this task may vary slightly according to the operating system):

1. In the folder where ViewDS Management Agent is installed (by default, `C:\Program Files\ViewDS Suite\VMA`), double-click the file `ras.3.p12`. A certificate window is displayed.
2. Follow the instructions on the screen to import the certificate.

   NOTE: A password for the private key is not required.

3. When installation is complete, follow the steps below to connect to the ViewDS server.

Connecting the VMA to the ViewDS server

To create a connection to a ViewDS server:

1. Start the ViewDS Management Agent. If the New Connection window is not open, then from the **File** menu, click **New Connection**. The New Connection window is displayed.

2. Complete the following details:
   - **Host** – the host name or IP address of the of the host on which the DSA is running.
   - **DSA port** – the port number to connect to on the host for the DSA. The default is 3000.
   - **RAS port** – the port number to connect to on the RAS (Remote Administration Service). The default is 3018.
   - **Certificate for DSA** – the certificate used by the ViewDS Management Agent to connect to the DSA.
   - **Certificate for RAS** – the certificate used by the ViewDS Management Agent to connect to the RAS.
   - **Connection name** – the name displayed by the ViewDS Management Agent application.
3. Optionally, test the connection by clicking **Test**. If the test fails, check the settings in step 2 and check network connectivity.
4. Click **Save**.
5. Click **Connect**. The new server connection is added to the left pane. It is displayed as a red box with a cross in it to show that the ViewDS Management Agent is not connected to the ViewDS server.
6. In the left pane, right-click the new server connection. A submenu is displayed.
7. Select **Connect to** and then click **RAS, then DSA**. The server is displayed as a green box with '(connected)' next to it.

Entering licence information

NOTE: You will need licence information (see License requirements) to perform this task.

To enter licence information using the VMA and start the DSA:

1. In the left pane of the ViewDS Management Agent, click the server icon.
2. In the right pane, click the **Configuration** tab.
3. Within the Configuration tab, click the **Licence** tab. An empty Licence screen is displayed.
4. Do one of the following:
   - Open the file containing the licence information, and copy and paste it into the Licence screen.
   - From the Licence screen, click **Import** and then select the file containing the licence information.
5. At the bottom of the right pane, click **Set**. The licence information is saved and you are given the option to restart the server.
6. Click the **Yes** button. The DSA restarts.
7. In the right pane, click the **Status** tab followed by the **General** tab. Confirm that the DSA is running and the database is open.

Configuring ViewDS

This section describes how to configure ViewDS Directory.

It includes the following:

- Configuring the DSA
- Configuring for Access Presence
- Configuring for the XACML framework

Configuring the DSA

This section describes how to use the ViewDS Management Agent to modify the following groups of Directory System Agent (DSA) parameters:

- Runtime settings
- Addresses
- Operational
- Licence

The runtime settings can also be modified through the DSA Controller, Stream DUA or ViewDS Fast Load utility (see the *ViewDS Directory Server: Technical Reference Guide*).

The remaining parameters are stored in the DSA's configuration file, which is a text file in the following default location:

- `${VFHOME}/setup/config` (Linux)
- `%VFHOME%\setup\config` (Windows)

where `${VFHOME}` or `%VFHOME%` is the location of ViewDS.

Runtime settings

To view or modify the local settings through the ViewDS Management Agent:

1. At the bottom or the right pane, click the **Server View** button.
   (When ViewDS is licensed for just Access Proxy, this button is unavailable.)
2. In the left pane, click the server.
3. In the right pane, click the **Configuration** tab.
4. Below the Configuration tab, click **Runtime Settings**.

The more important local settings are described below.

DOT threads

Each DOT (Directory Operation Thread) handles database queries synchronously. By having multiple DOT threads, the DSA can process queries asynchronously – that is, multiple queries can be processed simultaneously.

There are implications to having different numbers of DOT threads:

- **3 DOT threads**: The default setting. Having more than three DOT threads might improve throughput, but it will be at the expense of memory use.
- **2 DOT threads**: If the system is low on memory, try running with two DOT threads.
- **1 DOT thread**: This setting is appropriate for single-user operation, but it will slow throughput with multiple users. More than one DOT thread is required for replication and is strongly recommended for distributed operations in order to prevent the DSA deadlocking.

Three or four DOT threads will deliver the best performance for a non-distributed DSA with a high query load. The maximum setting is 5.

Max DOT size

The maximum size for a DOT thread in MB.

The size of a DOT thread increases according to the size of each query it receives. If it receives a query larger than the value of this parameter, the DOT thread handles the query and then restarts. Its size on startup is the size set by this parameter.

The default is 20 MB.

Max. updates

The number of DOT threads that process update operations simultaneously.

The maximum number of updates:

- cannot exceed the number of DOT threads.
- should be less than the number of DOT threads to ensure that some DOT threads are always available for queries. Otherwise, during heavy updating, the directory may be too slow in responding to queries.
- should be set to 1 for normal operation.
- should be set to 0 to disable updates to the database.
- should be greater than 1 for a DSA in a replication agreement. (For distributed operations, this setting avoids deadlocks while processing a chained update operation. For replication, this setting avoids DAP/LDAP update operations having to wait for considerably longer than normal while replication updates are processed.)

Safe size

The safe file is a database recovery log. It contains details of database transactions that are waiting to be committed. After a crash or improper shutdown, the DSA uses the safe file to recover the database before reopening it.

The safe file should be set to a size that will accommodate the largest anticipated transaction. Even though the file grows to accommodate larger transactions, it is advisable to pre-allocate disk space to ensure efficient performance.

A larger size results in faster database writes, but slower database restarts; and a smaller size results in slower writes, but faster restarts.

The minimum size is 1 MB, and the default is 8 MB. (The size for the demonstration directory, Deltawing, is 2 MB.)

Cache size

Sets the size of the memory cache used by the database.

The larger the cache, the less the database will need to access the disk, and the faster the response time. The cache should therefore be made as large as possible, ideally to the point where the entire database can be held in memory.

An approach to calculating the suggested cache size is as follows:

```
available memory = maximum memory - 6 - (12 x DOTs)
```

where:

- `available memory` is the amount of available memory in MB
- `maximum memory` is the maximum amount of RAM on the DSA's host system in MB
- `6` is the space required in MB for the DSA process (excluding the DOT threads)
- `12` is the space required in MB for each DOT threads
- `DOTs` is the number of DOT threads

If this calculation gives a negative number, consider adding more RAM to the host system. Otherwise, set cache to this number.

If the cache is too small, there is a serious effect on performance. If the available memory for the memory cache is less than 10% of the expected size of the database, then the host system needs more RAM.

Setting the Time limit, Size limit and DAP time-out

These settings apply to clients connected to the DSA. They apply to the VMA, Access Presence, and any LDAP, XLDAP, DAP or administration DUA.

- Time Limit – the DSA's time limit (in seconds) for a DUA user's read, compare, search and list operations. A normal value is 5 seconds. A value of -1 means there is no time limit.
- Size Limit – the maximum number of entries the DSA will return in response to a search or list operation. The default value is 2000.
- DAP Time-out – the time-out period for an inactive connection to a DUA. When there have been no requests from a DUA for the number of seconds set by this parameter, the connection is unbound. If set to zero, there is no time-out period.

A DUA user can override the Time Limit or Size Limit with a lower value. If, however, the user sets a higher value it is ignored by the DSA.

Addresses

The more important address parameters are described below.

DSA Access Point

The address presented by the DSA to allow a DUA or another DSA to access it.

Modify this parameter if the address required by a DUA or another DSA is different to the address used locally. For example, this might be the case if the DSA is behind a firewall that has Network Address Translation (NAT). The IP address used to access the DSA from an external network would therefore be different to the IP address used locally.

To view or modify the DSA Access Point:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by **General** then the **Text View** button.

DSA LDAP Address

The DSA listens on this address for native LDAP connections. If the DSA SLDAP Address is also defined, the two should specify different port numbers.

Recommended values (no default):

- `ldap://localhost:389` (Unix root or Windows)
- `ldap://localhost:{baseport}+6` (all others)

To view or modify the address settings through the ViewDS Management Agent:

1. In **Server View**, click the server in the left pane.
2. In the right pane, click the **Configuration** tab and then click **Addresses**.

Operational

To view or modify the operational parameters through the ViewDS Management Agent:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Configuration** tab and then click **Operational**.

Error level

The error level determines the minimum level of the errors written to the error log:

- error (includes console error messages)
- warnings
- status (the default setting)
- debug

These are minimum levels. Setting the level to 'debug' will report 'debug', 'status', 'warnings' and 'error' messages.

Licence

To view the licence parameters through the ViewDS Management Agent:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Configuration** tab and then click **Licence**. The licence information is displayed.

NOTE: If you modify the licence information provided by your ViewDS vendor, your installation will not operate correctly.

Configuring for Access Presence

Access Presence (webdua.cgi) is a web-based client that is included with an installation of the ViewDS server.

This section describes how to configure a web server so that Access Presence can access the demonstration directory, Deltawing, which is installed with the ViewDS server. It also describes how to configure the web server on a host that is remote to the host where the ViewDS server is installed. This is sometimes desirable for performance or security.

The supported web servers are:

- Apache HTTP Server
- Microsoft IIS Versions 6.0, 7.0, 8.0 and 10.0

Access Presence and ViewDS Suite installer

When the ViewDS server is installed through the ViewDS Suite installer for Windows, the installation includes Access Presence and a default IIS configuration.

The default IIS configuration makes the demonstration directory, Deltawing, available from:

```
http://host:8090/directoryservices/viewds/webdua.cgi
```

When running Access Presence and the ViewDS server on the same host, no further configuration is required to access the demonstration directory. However, instructions to configure IIS are provided here for reference.

Overview

To run Access Presence on a different host to the one running the DSA, see Configuring on a remote host.

To configure Access Presence for Deltawing when it is installed on the same host as your DSA, perform one of the following tasks:

- Configuring Microsoft Internet Information Services
- Configuring Apache HTTP Server

Configuring on a remote host

To install Access Presence on a different host to the one running the DSA:

1. Install the ViewDS server on the host on which Access Presence will be running (see Installing ViewDS components on Windows or Installing the ViewDS server on Linux).

   NOTE: The installation of the ViewDS server includes Access Presence by default.

   NOTE: You do not need a licence key for the Access Presence host.

2. Optionally, delete the **data**, **load** and **dump** folders, as these folders are not required by Access Presence.

For Linux

3. Configure a web server for Access Presence (see Configuring Apache HTTP Server).
4. In the configuration file on Access Presence's host, set the `dsaaddress` to the address of the DSA. By default, the configuration file is: `${VFHOME}/setup/config`.

For Windows

3. Remove the **ViewDS Administration** Windows service, as this is not required for a remote instance of Access Presence.

   NOTE: The ViewDS Suite installer for Windows configures the host's IIS web server for Access Presence so that the demonstration directory, Deltawing, is available at the following URL:
   `http://host:8090/directoryservices/viewds/webdua.cgi`

4. In the configuration file on Access Presence's host, set the `dsaaddress` to the address of the DSA. By default, the configuration file is: `%VFHOME%\setup\config`.

Configuring Microsoft IIS

For Access Presence to work correctly with IIS 7.0 and above, the **CGI** role service must be installed when IIS is enabled. See the Microsoft documentation for your version of IIS. If you use the ViewDS suite installer to enable IIS, then the **CGI** role service is installed automatically.

Overview

To allow Access Presence to connect to the demonstration directory, Deltawing, the following configuration is required for Internet Information Services (IIS):

- Create a website with a suitable name (for example, the ViewDS Suite installer creates a website called 'ViewDS Access Presence')

- Set up the following Virtual Directory below the website:
  - DirectoryServices - `%VFHOME%\webdir`
- Set up the following Virtual Directories below DirectoryServices:
  - ViewDS - `%VFHOME%\webdir\cgi-bin`
  - icons - `%VFHOME%\webdir\icons\newicons`
- Grant the following access rights for the Internet Guest account:
  - all access rights to `%VFHOME%\general`
  - all access rights to `%VFHOME%\print`
  - all access rights to `%VFHOME%\webdir\conf`
  - read-only access to `%VFHOME%\webdir`
- Enable permissions for the `webdua.cgi` process to run

`%VFHOME%` is the ViewDS install folder. The default location is `C:\ProgramData\ViewDS Suite\`.

Configuration for IIS 7.0 or above

To configure IIS to provide access to Deltawing (the following settings are the defaults set by the ViewDS Suite installer):

1. Start Internet Information Services Manager.
2. Under the existing Sites, create a new website:
   a. Right-click **Sites** and click **Add Website**. The Add Website window is displayed.
   b. In the **Site name** box, enter *ViewDS Access Presence*.
   c. In the **Physical path** box, select the install location for Access Presence `%VFHOME%\ViewDS Suite\webdua`.
   d. In the **Port** box, enter *8090*.
3. Create a virtual directory called *DirectoryServices*:
   a. Right-click the **ViewDS Access Presence** site and then click **Add Virtual Directory**. The Add Virtual Directory window is displayed.
   b. In the **Alias** box, enter *DirectoryServices*.
   c. In the **Physical path** box, enter `%VFHOME%\webdir\cgi-bin`.
   d. Click **OK**. An icon for *DirectoryServices* is displayed below the *ViewDS Access Presence* website.
4. Create a virtual directory called *ViewDS* below *DirectoryServices*:
   a. Right-click **DirectoryServices** and then click **Add Virtual Directory**. The Add Virtual Directory window is displayed.
   b. In the **Alias** box, enter *ViewDS*.

   c. In the **Physical path** box, enter `%VFHOME%\webdir\cgi-bin`.

   d. Click **OK**. An icon for *ViewDS* is displayed below the *DirectoryServices* folder.

5. Replace the *icons* folder below *DirectoryService* with a virtual directory called *icons*:

   a. Right-click **DirectoryServices** and then click **Add Virtual Directory**. The Add Virtual Directory window is displayed.

   b. In the **Alias** box, enter *icons*.

   c. In the **Physical path** box, enter `%VFHOME%\webdir\icons\newicons`.

   d. Click **OK**. An *icons* folder is replaced with an *icons* virtual directory.

6. Enable permissions for the webdua.cgi process:

   a. Click the **ViewDS Access Presence** site. The IIS configuration icons are displayed in the middle pane.

   b. In the middle pane, double-click the **Handler Mappings** icon. A list of Handler Mappings is displayed.

   c. In the right pane, click **Add Module Mapping**. The Add Module Mapping window is displayed.

   d. In the Add Module Mapping window:

      i. In the **Request path** box, enter **.cgi*

      ii. In the **Module** box, enter *CgiModule*

      iii. Ensure that the **Executable** box is empty.

      iv. In the **Name** box, enter *cgi*, then click **OK**.

7. Grant the following access rights for the Internet Guest account `IUSR` using the **Security** tab on the Windows **Properties** dialog:

   - Full control on `%VFHOME%\general`
   - Full control on `%VFHOME%\print`
   - Full control on `%VFHOME%\webdir\conf`
   - Read-only on `%VFHOME%\webdir`

   NOTE: The above are the default locations, which can be modified through the ViewDS Management Agent.

8. Test the configuration by starting the website and going to the following URL in a browser:

   ```
   http://host:8090/directoryservices/viewds/webdua.cgi
   ```

   The 'Welcome to Deltawing' webpage should be displayed.

Configuring Apache HTTP Server

To allow Access Presence to connect to the demonstration directory, Deltawing, the following configuration is required for an Apache HTTP Server:

- Add a CGI script handler
- Define the document root `${VFHOME}/webdir`
- Add an alias for the Access Presence directory `${VFHOME}/webdir/icons/newicons/`
- Add an alias for the help directory `${VFHOME}/webdir/help/newface/`
- Add a script alias for the webdua directory `${VFHOME}/webdir/cgi-bin`
- Ensure there is read access to all directories that comprise the `${VFHOME}` path
- Set up the following access to the directories:
  - all access rights to `${VFHOME}/general`
  - all access rights to `${VFHOME}/print`
  - read-only access to `${VFHOME}/webdir/conf`
  - read-only access to `${VFHOME}/webdir`

`${VFHOME}` is the path to the ViewDS installation directory on Unix.
The Windows equivalent is `%VFHOME%` and the defaults are as follows.
Windows 2003: `C:\Documents and Settings\All Users\Application Data\ViewDS Suite`
Windows 2008 onwards: `C:\ProgramData\ViewDS Suite\`

Example configuration for Windows 10

To configure an Apache HTTP Server to provide access to Deltawing on Windows 10:

1. Check that the following line is in the Apache `httpd.conf` file:
   `AddHandler cgi-script .cgi`
2. Add the following lines to the end of the Apache `httpd.conf` file:

```
Alias /directoryservices/icons/ "c:/ProgramData/Suite/webdir/icons/newicons/"
Alias /directoryservices/help/ "c:/ProgramData/Suite/webdir/help/"
ScriptAlias /directoryservices/viewds/ "c:/ProgramData/ Suite/webdir/cgi-bin/"
Alias /directoryservices "c:/ProgramData/ Suite//webdir"
<Directory "c:/ProgramData/ Suite/webdir">
  AllowOverride Option
  Options FollowSymLinks
  AllowOverride None
  Require all granted
</Directory>
```

This example uses the default ViewDS install location on Windows 2010: `C:/ProgramData/ViewDS Suite`.

3. Make sure that the `webdua.cgi` can write to the directory specified by the `webduaparampath` parameter. The default is `${VFHOME}/webdir/conf`.
4. Restart the web server.
5. Test the configuration by going to the following URL on a browser:

   `http://host:8090/directoryservices/viewds/webdua.cgi`

   The 'Welcome to Deltawing' webpage should be displayed.

Configuring for the XACML framework

This subsection describes the following configuration parameters that apply to the XACML framework, plus the steps to modify them through the ViewDS Management Agent (VMA):

- Default version
- User base object
- User attributes
- Resource attributes
- Policy base object

Default version

Every XACML policy has a version number.

By default, when there are multiple policies or policy sets with the same identifier, the Policy Decision Point (PDP) uses the one with the highest version number. Alternatively, if a Default Version is defined, the PDP uses the policy or policy set with the highest version number less than or equal to this value.

This parameter only applies to XACML policy that was not defined through the ViewDS Management Agent or the Authorization Policy Manager.

User attributes

The directory attributes the Policy Decision Point (PDP) will pre-fetch when it needs to obtain a directory attribute from a user's entry.

This parameter applies to all XACML policy.

Resource attributes

The directory attributes the Policy Decision Point (PDP) will pre-fetch when it needs to obtain a directory attribute from a resource entry (see the topic *Attribute look-up* in the VMA in-application help).

This parameter applies to all XACML policy.

User base object

The root of the subtree in the directory that the Policy Decision Point (PDP) will search in order to find a user entry. (The directory acts as a Policy Information Point by storing information that can influence an access decision.)

This parameter applies to all XACML policy.

Policy base object

The root of the subtree in the directory that the Policy Decision Point (PDP) will search in order to find a policy or policy set.

This parameter applies to all XACML policy.

Modifying the XACML configuration

To set the XACML configuration parameters through the ViewDS Management Agent:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the appropriate server.
3. In the right pane, click the **XACML Config** tab.
4. Complete the boxes in the XACML Config tab as required.
5. At the bottom of the tab, click **Set XACML Configuration**.

Exploring ViewDS

This section describes the ViewDS data model to help you become familiar with how information is organized. It then takes you through using the demonstration directory, Deltawing, to help you become familiar with the directory from a user's perspective. It also provides further experience of using the ViewDS Management Agent.

This section covers the following topics:

- ViewDS data model
- About the demonstration directory
- Exploring through Access Presence
- Exploring through the Management Agent

ViewDS data model

This section describes the following aspects of the hierarchical data model that applies to all ViewDS directories:

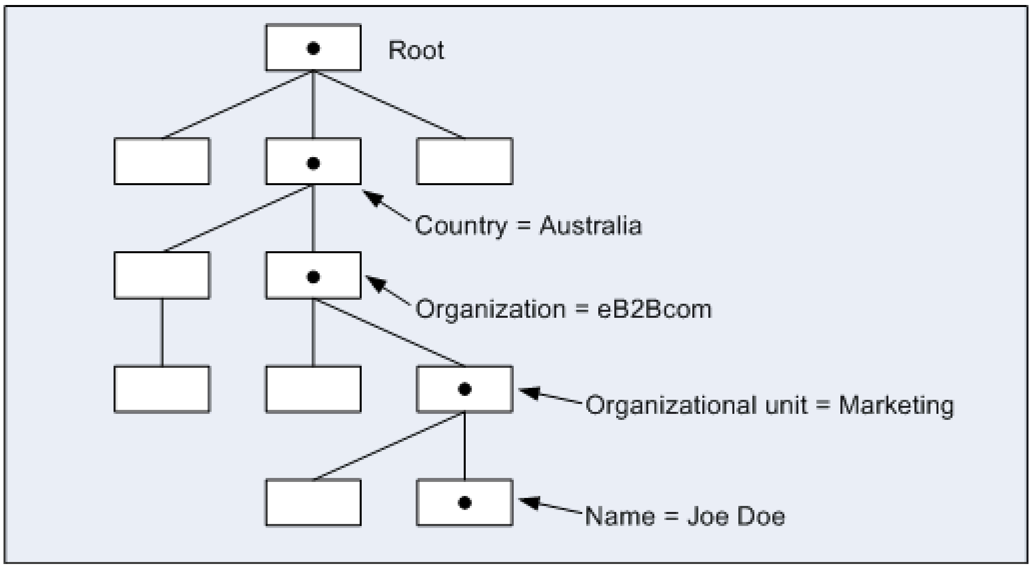
- Directory Information Tree
- Context prefix
- Attributes
- Distinguished Name (DN)

Directory Information Tree

A ViewDS directory includes a collection of information about objects in the real world. These objects might be countries, localities, organizations, organizational units, people and job functions, for example.

The objects are represented in the directory by entries arranged in a tree hierarchy. For example, a country contains localities and national organizations, an organization is divided into a hierarchy of organizational units such as divisions and branches, each containing people.

This hierarchical tree is called the **Directory Information Tree** (DIT). An illustration of a DIT is shown below.



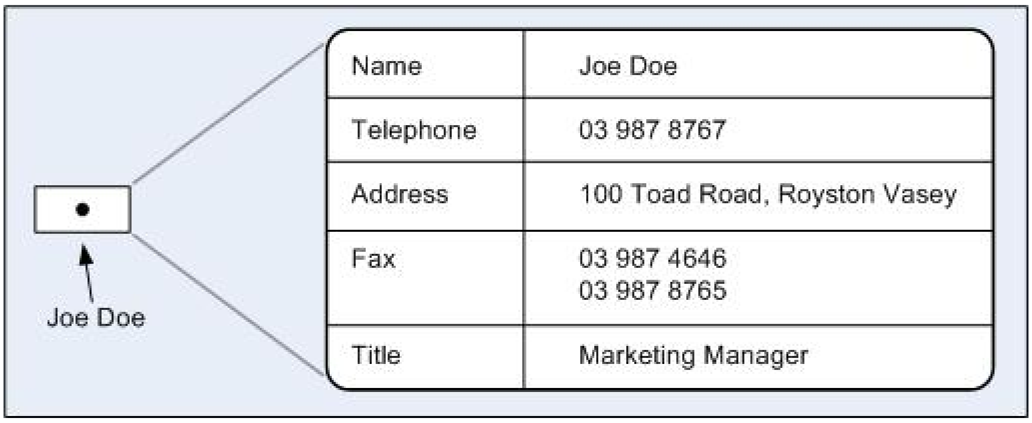
The DIT structure is important when searching a directory because it limits the scope of a search to a specific subtree. The DIT is also the basis for distributing a directory between multiple systems at different sites.

Context prefix

The entry at the top of a discrete area of the DIT is called the context prefix. In the above illustration, for example, there is a context prefix named 'Australia'.

Attributes

Each piece of information held in an entry is called an attribute. An attribute consists of a label identifying the category or type of information (for example, name or telephone number) and a list of one or more values (the actual name or telephone number). This is illustrated below.



Name	Joe Doe
Telephone	03 987 8767
Address	100 Toad Road, Royston Vasey
Fax	03 987 4646 03 987 8765
Title	Marketing Manager

Mandatory and optional attributes

An entry has two sets of attributes: mandatory attributes and optional attributes. An entry is only valid if it has values for its mandatory attributes. The optional attributes without values are not displayed by the DUA.

Distinguished Name (DN)

Every entry has a Relative Distinguished Name (RDN), comprising the values of one or more naming attributes. For example, the naming attribute for the entry in the above illustration might be `Name`.

The sequence of RDNs from the root of the DIT to an entry forms a unique path, which forms the entry's Distinguished Name (DN).

An entry's DN is formed by adding its RDN to the DN of its immediate superior. For a public white pages directory, for example, the superior might be a locality or a telephone zone. The RDN can consist of multiple components (such as name and address) if this is needed to ensure that all RDNs with the same superior can be uniquely identified.

Aliases

Some entries in the DIT are not objects themselves, but instead, point to the name of an object. These entries are called aliases.

Aliases can be used to make the names of entries more user-friendly, or to provide a transition from an old name to a new name. Aliases also make it possible, for example, for a person who sits on several committees to have one entry that can be found through aliases under each committee.

About the demonstration directory

A new installation of ViewDS Directory includes a demonstration directory called Deltawing. It allows you to explore the functionality of ViewDS from a user's perspective through Access Presence, and from an administrator's perspective through the ViewDS Management Agent.

Deltawing is a fictitious, small, and highly diversified company that develops, markets and sells products in the Automotive, Information Systems and Pay TV industries. The directory contains 1014 entries (although a ViewDS directory can store many millions of entries).

Deltawing includes more than just organizational groups and people. The object classes are:

- Units (used for storing organizational departments, branches, sections, etc.)
- Organizational Person (used for storing people who work for an organization)
- Working Party (used to store information about work groups such as committees, or teams)
- Role (used to store information about a specific job role in an organization, such as phone, fax, E-mail, and the incumbent person(s) in the role)
- Device (used to store information about physical devices such as printers)
- Meeting Room (used to store information about conference rooms)
- Alias entries (used to store ‘pointers’ to entries elsewhere in the directory rather than duplicating the entries unnecessarily)

Exploring through Access Presence

This tutorial introduces Access Presence to give you experience of ViewDS Directory from the user's perspective.

Connecting to Access Presence

To connect to Access Presence:

1. Open the following URL:

   ```
   http://host:8090/directoryservices/viewds/webdua.cgi
   ```

2. Log on with the user name asherma and password testpass.

The user names, passwords and access levels for the Deltawing directory are as follows:

User name	Password	Access level
rturnbu	testpass	read
cjoyce	testpass	update
asherma	testpass	superuser

Finding and viewing an entry

To search for and view Mike Smith's entry:

1. From the **Welcome** page, click the **Access** button. The Advanced Search page is displayed.
2. In the **Surname/Name** box, enter *smith* and then click **Show**. Twelve directory entries are returned that approximately match the name.
3. Click the fifth entry (**Mike Smith**).

   The first section of the page lists the superior entries above Mike Smith in the DIT: deltawing, Deltawing Home Media Ltd, Internet Services, World Wide Web Services, and Strategic Relationships. The next section displays Mike Smith's details. There are no subordinate entries below Mike – if there were, they would be listed in another section below his details section.
4. Click **Tree Browsing**. The Deltawing DIT is displayed with Mike Smith's leaf entry highlighted in bold text.
5. Click **Tree Browsing** again to hide the Deltawing DIT.

Finding the manager of a unit

To find the manager of Delta Home Media Ltd:

1. In the first section of the page, click **Delta Home Media Ltd**. The entry for the unit is displayed.

   The first section of the page shows that this entry has one superior entry, deltawing. The next section contains the unit's details, including a link to its manager Karen Johannesen. The final section contains the unit's subordinate entries.
2. Click **Karen Johannesen**. Her details are displayed.

Searching for entries by function

To find all meeting rooms in Deltawing:

1. At the top of the page, click **Change Search Form**. The Welcome page is displayed.
2. In the drop-down box, click **Function Search** and then click **Access**. The Advanced Search page is displayed.
3. In the function box, enter 'meeting' and press the return key. A list of meeting rooms is displayed.
4. Click the third meeting room in the list. The entry for the Sales Meeting Room is displayed. The first section of the page shows that this entry has three superior entries in the DIT. The next section contains the room's details, which includes a link to the person who deals with bookings, Mary Smith.
5. Click **Mary Smith**. Her details are displayed. Note that all details are user attributes except for Last Modified, which is an operational attribute.

Modifying an entry

To add a mobile phone number to Mary Smith's entry:

1. Note the date and time for the Last Modified attribute.
2. At the bottom of the page, click **Modify**. The Modify page is displayed.

   NOTE: All attributes are marked for pre-processing, which is defined through the ViewDS Management Agent. Also note that Access Controls determine whether a user is allowed to modify an entry.

3. Add a number (any number) to the **Mobile** box and then click **Save**. A confirmation dialog box is displayed.
4. Click **OK**. Mary's details page is displayed. Note that her entry now includes:
   - under Mobile, the new number you entered
   - under Last Modified, an new date and time
   - under Last Updated, the name of the user you are logged on as

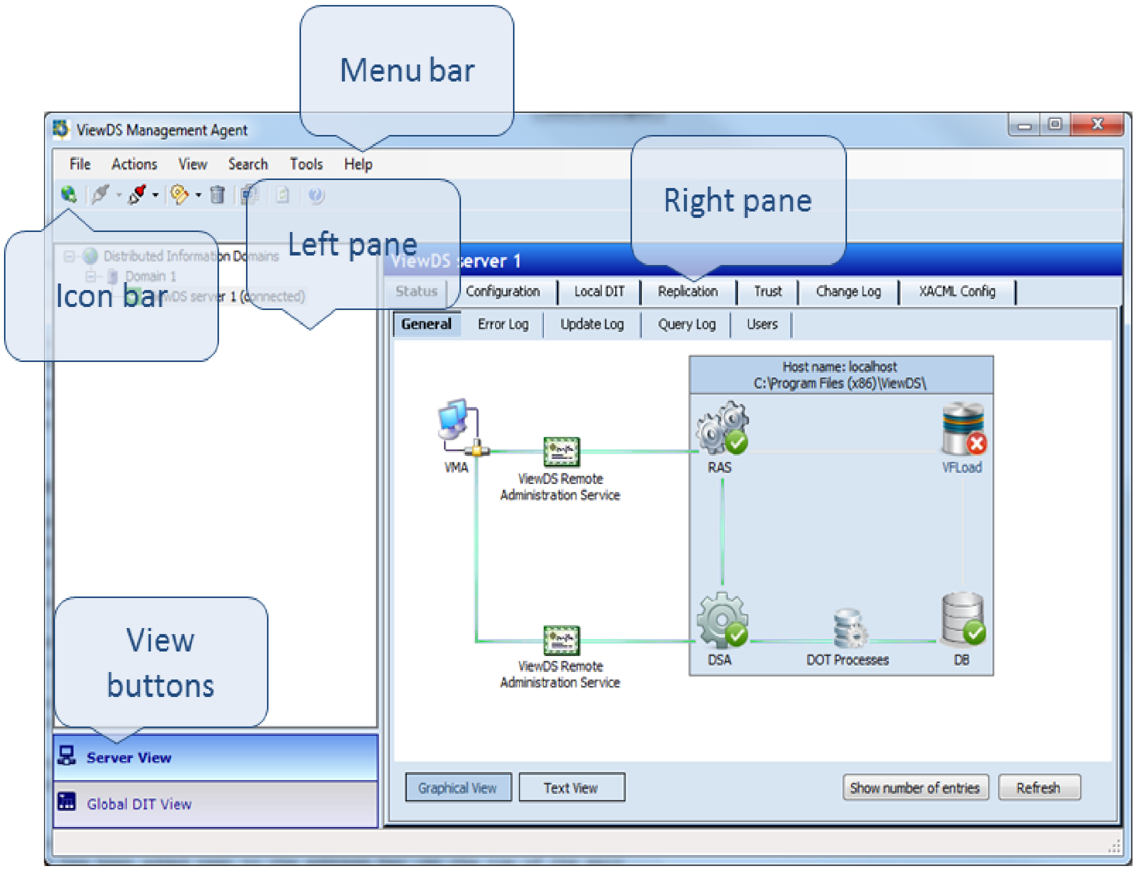
Moving a non-leaf entry

To move the Deltawing Automotive sales department to Deltawing InfoSystems:

1. Click **Tree Browsing**. The DIT for Deltawing is displayed in a new frame.
2. In the DIT, click **Deltawing InfoSystems**. Its details are displayed.
3. In the details frame, click **Set Target Object**.
4. In the DIT below Deltawing Automotive Ltd, click **Sales**.
5. In the details frame, click **Move**. A confirmation dialog box is displayed.
6. Click **OK**. The Sales unit is displayed below Deltawing InfoSystems.

Exploring through the Management Agent

This subsection introduces you to the ViewDS Management Agent (VMA), and involves exploring Deltawing through the application. The main areas of the VMA are shown below:



The main areas are:

- Menu bar
- Icons bar
- Left pane
- Right pane
- View buttons

The view buttons allow you to move between two views:

- **Server View** - This view allows you to manage one or more ViewDS servers (for example, RAS and DSA status, configuration, logs and replication agreements).
- **Global DIT View** - This view allows you to manage the data stored by a DSA (for example, entries in the Directory Information Tree, schema and access controls).

The content of the left and right panes is different according to whether the Server View or Global DIT View is displayed. (The following example shows the Global DIT View.)

If your installation is licensed for just ViewDS Access Proxy, only the Server View is available. The Server View and Global DIT View buttons are not displayed.

Viewing the Deltawing DIT

1. At the bottom of the left pane, click **Global DIT View**. The Deltawing DIT is displayed in the DIT tab on the left.
2. Open and close different branches of the DIT by clicking them.

Viewing an entry's attributes

1. In the left pane, click the **Deltawing** entry. The Directory Information Tree (DIT) is expanded.
2. Click the entry for **Deltawing Automotive Ltd** and then click the entry for **Tony Liggett**. The Attributes tab in the right pane now shows the entry's attributes and their values.

   Note the following:

   - The structural object class for the entry is stored by the attribute `objectClass`. This entry's structural object class is `organizationalPerson` (subclass of `person`).
   - The entry's RDN is the value of `commonName`, Tony Liggett. This value is displayed in the DIT.

Adding an attribute to the entry

1. Right-click anywhere in the right pane and then click **Add Attribute**. The Add an Attribute window is displayed.
2. For the **Attribute Type**, click **comment**. (Note that the available attributes for the entry's structural object class are listed by the Attribute Type drop-down box).
3. Click **Add Attribute**. The new attribute is added to the Attributes tab.

Assigning a value to the new attribute

1. In the **Attributes** tab, right-click the comment attribute and then click **Modify Attribute Value**. The cursor is displayed in the Value cell for the attribute.
2. Enter a value and then press the enter key.
3. At the bottom of the right pane, click **Submit**. The changes you have made are saved.

As `comment` is a multi-value attribute, you have the option to add another instance of the attribute to the entry by repeating this task from step 5.

Operating the Directory

This section describes how to perform operational procedures from the command line and through the ViewDS Management Agent.

It covers the following topics:

Starting and stopping the ViewDS server

This subsection describes how to start and stop the ViewDS server – Directory System Agent (DSA) and Remote Administration Service (RAS) – and how to open and close its database. The status of the DSA is started or running when its processes are running; its status is terminated when they are not running.

To be available for use, the DSA must be started and its database must be open. Normally, the database is opened when the DSA starts, but some administrative operations on the DSA are only possible when the database is closed. Opening and closing the database are administrative operations on a running DSA.

NOTE: All administration tasks require the DSA to be running.

Starting the DSA

You can start the DSA through one of the following:

- ViewDS Management Agent
- Command line
- Windows Services Manager

ViewDS Management Agent

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click a server.
3. In the right pane, click the **Status** tab.
4. In the Status tab, click **General**. The General screen has two views:

   Graphical View

   - In the right pane, right-click the DSA icon and click either **Start DSA (Database Open)** or **Start DSA (Database Closed)**. The DSA is shown as running and its connection to the VMA is restored.

   Text View

   a. For the DSA to start with the database closed, select the **Database Closed** checkbox.

   b. Click **Start DSA**. The DSA status changes to `running`.

Command line

1. Log in to the ViewDS host as the system administrator.
2. The RAS is usually configured to start the DSA as part of its own starts-up process. To start the RAS and DSA enter the following command: `ras`

Windows Services Manager

1. If ViewDS is not installed as a service, enter the following command:

   ```
   ras -i
   ```

2. Open the **Control Panel** and choose the **Services** option.
3. Select the **ViewDS Administration** service and click **Start**. (The name of the service is set by the configuration-file parameter `rasServiceName`.)

   Click **on Startup** to set other options, such as automatic startup on reboot.

Opening the database

The database normally opens when the DSA starts. If DSA is running but the database is closed, you can open it through either the ViewDS Management Agent or DSA Controller.

ViewDS Management Agent

1. If the **General** screen in the **Status** tab is currently displayed, move to step 5 in this task. Otherwise, continue to step 2.
2. At the bottom of the left pane, click **Server View**. The DSA pane is displayed.
3. In the left pane, click the server.
4. In the right pane, click the **Status** tab followed by the **General** tab, then the **Text View** button. The screen includes the status of the database.
5. Click **Open Database**. The status of the database changes to running.

Command line

- Enter the command: `dsac open`

Closing the database

Before you can modify certain configuration parameters, the database must be closed with the DSA running.

ViewDS Management Agent

1. If the **General** screen in the **Status** tab is currently displayed, move to step 5 in this task. Otherwise, continue to step 2.
2. At the bottom of the left pane, click **Server View**.
3. In the left pane, click the server.
4. In the right pane, click the **Status** tab followed by the **General** tab, then the **Text View** button. The screen includes the status of the database.
5. Click **Close Database**. The status of the database changes to closed.

Command line

- Enter the command: `dsac close`

Stopping the DSA

The DSA must be stopped before you back up its database or modify certain configuration parameters.

ViewDS Management Agent

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click a server.
3. In the right pane, click the **Status** tab.
4. In the Status tab, click **General**. The General screen is has two views:

   Graphical View

   - In the right pane, right-click the RAS icon and then click **Stop DSA**. The DSA is shown as stopped and the connection to the VMA is greyed out.

   Text View

   - Click **Start DSA**. The DSA status changes to `stopped`.

Command line

The RAS is usually configured to stop the DSA as part of its shut-down process.

- Enter the following command to stop the RAS (and the DSA): `ras stop`

Viewing status

This section includes how to test the directory's availability, examine the status of the database and dot threads, and list the current users (the bind list for DUAs and other DSAs).

Checking whether the DSA is running

There are several ways to check whether the DSA is running, which are described below.

ViewDS Management Agent

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button. The screen includes the status of the DSA.

Start DSA Controller or Stream DUA

A simple way to check whether the DSA is running is to start either the DSA Controller or Stream DUA. Both bind into the DSA when they start up, or print an error message if they cannot connect.

- To start the DSA Controller, enter `dsac`
- To start the Stream DUA, enter `sdua`

Connect through Access Presence

Alternatively, you can try to connect to the directory through Access Presence and see whether you can proceed past the login dialog (or automatic login if applicable). If the directory cannot be accessed, the following message is displayed:

```
Cannot connect to the directory, try again later
```

Linux ps command

You can also use the Linux `ps` command (the exact form depends on the operating system). There should be a single `dsa` process and a single `rasrv` process.

Viewing the directory status

The directory's status is the internal state of its processes and the values of its parameters. You can view the directory status through the DSA Status screen of the ViewDS Management Agent or through the DSA Controller's display command.

ViewDS Management Agent

To display the DSA Status screen:

1. At the bottom of the left pane, click the **Server View** button.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button. The General screen for the DSA is displayed.

The General screen includes the following status information:

- **RAS connect status** – the status of the DSA's connection to the RAS. If the DSA is not connected to the RAS, certain information and functions will be unavailable. You will not be able to start or stop the DSA, view logs, or view or modify the DSA's configuration.
- **Installation path** – the installation path for the DSA.
- **DSA status** – the status of the DSA, which is either running or stopped.
- **Database status** – the status of the DSA’s database, which is either running or closed.
- **DOT thread status** – the status of each DOT (Directory Operation Thread) thread. The DOT threads provide an interface between the DSA and its database, and also provide ViewDS’s flexible searching capabilities.
- **No. of current connections to DSA** – the number of users currently connected to the DSA. The users can include DUAs, LDAP clients, other DSAs and the RAS.
- **No. of DSA entries (master/shadow)** – the number of master entries the DSA contains, and the number of entries it shadows.

Command line

To display the current status and settings, enter the following command:

```
dsac display
```

A typical response is as follows:

```
DSA                 : available# line 1
  Database          : open# line 2
  DOT 1             : waiting for an operation# line
  Number of dots    : dots        = 1
  Max dot size (MB) : dotsize     = 20
  Max sessions      : sessions    = -1
  Max updates       : updates     = 1
  Size limit        : sizelimit   = 2000
  Time limit        : timelimit   = 30000
  Cache size (MB)   : cache       = 2
  DAP timeout       : daptimeout  = 0
  DSP timeout       : dsptimeout  = 0
  Optimistic        : optimistic  = on
  Async mode        : async       = on
  Recovery          : recovery    = on
  Enable ldap       : ldap        = off
  Query logging     : qlog        = on
  Update logging    : ulog        = onv
  Activity logging  : alog        = off
  LDAP session log  : clog        = off
  SEP size factor   : sizefactor  = 5
  Search aliases    : searchalias= on
  Safe file initial size(MB)      = 4
```

The first line, DSA, shows that the status of the DSA as one of:

- available
- not available (no DOTs)
- not available (DSA is starting DBM)
- not available (database is closed)
- not available (closing DOTs)
- not available (closing database)

The second line, Database, shows that the status of the database as one of:

- open
- open (being dumped)
- open (being emptied)
- closed

The third line, DOT, shows a status of:

- connecting to DSA
- waiting for an operation
- being closed by DSA
- processing operation
- returning operation
- abandoning operation

If you have set a value for a parameter that applies until the DSA is restarted, then the persistent value is displayed in brackets next to this temporary value.

The DSA parameters are described in the *ViewDS Directory Server: Technical Reference Guide*.

Listing the users

The procedures in this section list the users (including DUAs and other DSAs) that are currently connected to a DSA.

ViewDS Management Agent

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab and then click **Users**.

   Details of the users connected to the DSA are displayed.

Command line

To list the users currently connected to the DSA, enter the command:

```
dsac userlist
```

The response has the following format:

```
User                    Session                 Queries Updates
-----------------------------------------------------------
Mr Mark Jones            1997-05-22,16:05:34      14      0
Mr Bob Chambers          1997-05-22,15:33:31       2      0
Ms Sue McDonald          1997-05-22,09:02:12      89      9
Fred Halliday            1997-05-22,15:08:11       7      0
G.S. Brown               1997-05-22,16:09:58       3      0
Dr Thomas Castle         1997-05-22,16:15:09      11      0
Deltawing Test DSA       1997-05-22,15:33:31      177    29
```

It shows:

- the Common Name of the user or remote DSA;
- the date and time their session started (the time of login);
- the number of queries requested by the user or DSA; and
- the number of updates requested by the user or DSA.

Creating a new empty database

Creating a new database involves removing all data and schema, and then creating a new first-level entry (a context prefix) in the database.

ViewDS Management Agent

Emptying a DSA's database

This task empties a directory. It deletes the schema and removes all data from the database. It is advisable to back up the database before performing this task.

To empty a database:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button. The General screen for the DSA is displayed.
4. In the Database Recovery area, click **Empty**. All data is removed except for an empty root entry.
5. Follow the steps to add a first-level entry below.

Adding a first-level entry

This task adds a first-level entry to a directory and gives you the option to import a predefined schema and create default X.500 Basic Access Controls.

For more about predefined schema, see the *ViewDS Directory Server: Technical Reference Guide*.

To add a first-level entry:

1. In the right pane, click the **Local DIT** tab. The **Root** entry is displayed in the Master tab.
2. Right-click the **Root** entry and then click **Add First Level Entry**. The Add First Level Entry window is displayed.
3. Follow the instructions on the screen. (Press **F1** to view help for the ViewDS Management Agent.)

Command line

Creating a new database from the command line involves starting the DSA, initializing the database's safe file (see Runtime settings) and then emptying the current database.

To empty a database:

1. The RAS is usually configured to start the DSA as part of its starts-up process. To start the RAS (and therefore the DSA) enter the command: `ras`
2. Enter the following command: `dsac close init open empty`

   You can override the default size for the safe file by including a parameter after the init command, but this is rarely necessary.

You now have a database containing only an empty root entry. You need to add many operational attributes to the root entry and to the first real entries created below the root to set up schema, security, knowledge, DUA configuration information etc. This is normally done automatically when you load or reload bulk data into a database.

Bulk loading a database

This section describes how to load a new or empty database with bulk data.

Bulk data is generated when a ViewDS database is dumped, or when data is extracted from a non-ViewDS database and converted.

Bulk data must be in the format of a series of insert (or entry) commands to the Stream DUA (for more information about the Stream DUA, see the *ViewDS Directory Server: Technical Reference Guide*). The entries must be ordered so that a superior entry is inserted before its subordinates.

The following conventions apply to bulk-data files:

- file names should take the form dib.nnnnn
- nnnnn is a sequential number beginning with 00000
- bulk-data files are stored in the dump directory
- each file holds data for approximately 1000 entries

ViewDS Management Agent

Loading a database through the Management Agent involves stopping the DSA. If this is unacceptable, see Stopping ViewDS is unacceptable.

To load a database:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab.
4. In the Status tab, click the **General** tab followed by the **Text View** button.
5. In the right pane, click **Stop DSA**. The DSA status changes to stopped.
6. In the **Database Recovery** area, click **Load**. The database may take several minutes to load. When it has loaded, a confirmation window is displayed.
7. Click **Start DSA**. The DSA status changes to running.

Command line

There are two options when loading bulk data from the command line:

- If stopping ViewDS is acceptable, use the ViewDS Fast Load (vfload) utility.
- If stopping ViewDS is unacceptable, use Stream DUA.

Stopping ViewDS is acceptable

This option uses the ViewDS Fast Load utility, which offers no protection against fatal errors. If a crash or fatal error occurs you will need to rebuild the database.

For this reason, when loading large amounts of data, you may prefer to prepare a load script that breaks up the process to the following steps:

1. Load a number of dib.* files.
2. Copy the ddm.* files to a safe location.
3. Repeat until all files have been loaded.

If you encounter a fatal error, resume the load from the last checkpoint rather than restarting from the beginning.

To stop the DSA and then bulk load a database:

1. Stop the DSA by entering the following command: `ras stop`

   NOTE: If you want to preserve but disable the existing database files, you can now move the data/ddm.* files to another directory. Never move the data/ddm.* files unless the DSA has been stopped.

2. Run ViewDS Fast Load with the names of the files to be loaded: `vfload -dm ../load/*`

   The files are loaded from the load directory.

> **NOTE:** If building of the indexes was deferred by using the empty for filling command, the last command needs to be a fill command. Processing of this fill command may take a long time if the database is very large. This is normal.

3. Start the DSA by entering the following command: `ras`

   Users can now access the directory.

Stopping ViewDS is unacceptable

Before loading bulk-data files to a running database, check that none of the files contain an empty command.

There is also a Stream DUA mechanism to consider that helps minimize the impact of the load on users. The Stream DUA's sleep file allows you to specify the time period it should pause between executing operations. For more information, see the *ViewDS Directory Server: Technical Reference Guide*.

To bulk load a database that is running, enter the following command:

```
sdua -dm file1 file2 ...
```

where file1 and file2 are the files to be loaded.

Viewing logs

This subsection describes the following ViewDS logs:

- Query log
- Update log
- Error log
- VMA log

You can configure and view the logs through a text editor or the ViewDS Management Agent (VMA).

Query log

The query log contains users' queries – read, compare, search and list operations – but not the results. The contents of the query log can be replayed using the Stream DUA without any modification (a query that generates an X.500 error is logged, but is commented out with a '#' character and therefore ignored by the Stream DUA).

When query logging is on, all attempted query operations are written to the query log. This log is useful when monitoring performance, tracking problems, or building a file of typical queries.

> **NOTE**: The query log is normally off. If left on, the query log file will grow very quickly and the DSA's host will eventually run out of disk space.

ViewDS Management Agent

To work with the query log:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server whose query log you want to view.

To turn the query log on or off:

1. In the right pane, click the **Configuration** tab followed by the **Runtime Settings** tab.
2. For the **Query logging** parameter, select either **on** or **off**.
3. At the bottom of the right pane, click **Set**.

To view or modify the location of the query log:

1. In the **Configuration** tab, click **File System**. The location is set by the Query logs parameter.
2. If required, modify the Value for the Query logs parameter and then click **Set**.

To view the query log:

- In the right pane, click the **Status** tab followed by the **Query Log** tab.

Command line

The query logs are written if the DSA operational parameter qlog is on:

```
dsac setwrite qlog = on
```

The query logs are written to the file `qlog` in the directory set by the configuration-file parameter `qlogdir` (by default, `${VFHOME}/logs` or `%VFHOME%\logs`).

Update log

The update log contains all users' update operations – add, remove, modify, move or rename an entry.

The update log is critical to maintaining database integrity after a failure. After restoring a backup, replaying this log will update the database according to all committed transactions since the backup was made.

The contents of the log can be replayed using Stream DUA without any manual modification (an update that generates an X.500 error is logged, but is commented out and therefore ignored by Stream DUA). However, it must first be run through the smerge utility, which ensures that all operations in the log are in chronological order.

ViewDS Management Agent

To work with the update log:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server whose update log you want to work with.

To view the update log:

1. In the right pane, click the **Status** tab.
2. Below the Status tab, click **Update Log**. The update log is displayed.

To export the update log:

1. In the right pane, click the **Status** tab.
2. Below Status tab, click **Update Log**.
3. At the bottom of the right pane, click **Export** . The update log is exported to the `logs` directory (by default, the `logs` subdirectory in the ViewDS installation directory).

To view or modify the location of the update log:

1. In the right pane, click the **Configuration** tab.
2. In the Configuration tab, click **FileSystem**.
3. If required, modify the Value of the **Update logs** parameter.
4. At the bottom of the right pane, click **Set**.

Command line

The update log is written if the DSA operational parameter ulog is on:

```
dsac setwrite ulog = on
```

The update log is written to the file `ulog` in the directory set by the configuration-file parameter `ulogdir` (by default, `${VFHOME}/logs` or `%VFHOME%\logs`).

Error log

The error log records all errors (except X.500 operational errors) encountered by the DSA and co-resident DUA processes. Out of context, individual error messages can be rather cryptic and potentially misleading (for example, warnings and errors may be given when nothing appears to be wrong).

Each log entry has a time-stamp, a process number and error message. The error messages are often only warnings and can be ignored. For example, the log may include messages to say a time limit is too short or there have been too many aborted transactions on a modify command.

The error log cannot be switched off and grows indefinitely. However, the standard dbbackup script (Linux only) installs an empty error log after completing a weekly backup.

ViewDS Management Agent

To view or modify the location of the error log:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Configuration** tab and then click **File System**.
4. If required, modify the **Value** for the **Error log** parameter.
5. At the bottom of the screen, click **Set**.

To view the error log:

1. In the right pane, click the **Status** tab.
2. In the **Status** tab, click **Error Log**. The error log is displayed.

Command line

The error log is written to the file specified by the configuration-file parameter `errorlog` (by default, `${VFHOME}/general/error` or `%VFHOME%\general\error`).

VMA log

The VMA log is a text file that contains event and error messages generated by the ViewDS Management Agent.

The default location of the VMA log is: `/ProgramData/ViewDS/logs/AdminUI.log`

Log level

The log levels are described below. Note that these are 'minimum' log levels. This means that, for example, if the level is set to `Info`, then `Info`, `Warn`, `Error` and `Fatal` are all logged, but `Debug` and `Trace` are not. There is also the option to set the level to `Off`.

Level	Logged
`Fatal`	Events that cause the application to close.
`Error`	Errors relating to an aspect of the application failing; the application may or may not continue to run.
`Warn`	Warnings relating to an unexpected event, but the application continues to run.
`Info`	Information about normal behaviour such as, mail sent, user updated profile, etc.
`Debug`	Debugging information such as, query executed, user authenticated, session expired, etc.
`Trace`	Trace debugging information such as, begin *method X*, end *method X*, etc.

Enable the VMA log

To enable the VMA log:

1. From the **Tools** menu, click **Settings**. The ViewDS Management Agent Settings window is displayed, which includes the Logging section.
2. In the **Level** box, click the required minimum log level (see above).
3. In the **Configuration** box, click either:
   - `production-size-restricted` – the size of the log file is restricted by the storage space available.
   - `local-per-run` – the log file is cleared when the VMA is closed down.

Disable the VMA log

To disable the VMA log:

1. From the **Tools** menu, click **Settings**. The ViewDS Management Agent Settings window is displayed, which includes the Logging section.
2. In the **Level** box, click **Off**.

View the VMA log

1. From the **Tools** menu, click **Settings**. The ViewDS Management Agent Settings window is displayed, which includes the Logging section.
2. Click **Open in File Explorer**, then double-click `AdminUI.log`.

Dumping a database

When a database is dumped, a text file is generated that contains all data from a specified subtree or from the entire database. This data can be reloaded into the database later.

A dump is a single atomic operation that produces a snapshot of the database. During a dump, the database allows normal operations (including modify) but there can be only one dump process running at a time.

A dump produces a text file (`dib.*`) in the dump directory specified by the configuration-file parameter dumpdir (by default, `${VFHOME}/dump` or `%VFHOME%\dump`). The dump files usually occupy around half the disk space used by the directory database files.

You can dump a database from the ViewDS Management Agent or the command line.

ViewDS Management Agent

To dump a DSA's database:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button.
4. In the **Database Maintenance** area of the screen, click the **Dump** button.

Command line

There are several options for dumping from the command line:

- Full dump
- Subtree dump
- LDIF dump
- ELDIF dump

Full dump

To dump the entire database:

```
sdua -c dump
```

Subtree dump

To dump a specific subtree:

```
sdua -c dump name
```

Where name is the full Distinguished Name of the entry at the top of the subtree to be dumped. (The name typically extends over several lines, and precise spelling and punctuation are required.)

Alternatively, to avoid potential problems with typing the full Distinguished Name:

1. Use Stream DUA to search for the entry at the top of the subtree to be dumped. For example:

   ```
   sdua -c "search {} for ou ~= \"media\" return" > dumpcmd
   ```

   Where:

   - `media` appears in the name of the entry at the top of the subtree
   - `dumpcmd` is the text file to which the result of the search is written
2. Remove any spurious entries from the text file and replace the keyword entry with dump.
3. Run the following command: `sdua dumpcmd`

LDIF dump

To dump the entire database in the LDIF format:

```
sdua -c 'dump as ldif'
```

ELDIF dump

To dump the entire database in the ELDIF format:

```
sdua -c 'dump as eldif'
```

Selective dump

For more control over the entries and attributes dumped, use the Printing DUA described in the *ViewDS Access Presence: Technical Reference Guide*.

Backing up a database

A database can be backed up using one of the following commands: Save or Dump.

Save – saves a copy of the database binary files to the `save` directory.

The default location is the `save` directory below the ViewDS installation directory. Note that the database remains available for normal operation, including updating, while the command is being processed. Also note that it is unsafe to copy the database binary files in any other way than through the save command.

Dump – dumps the database to text files (`dib.*`) in the `dump` directory.

The default location is the `dump` directory below the ViewDS installation directory. A dump is slower than a save, but has the advantage of allowing data to be transferred across different platforms and allowing recovery from database corruption. The dumped files usually occupy around half the disk space used by the DSA's database files.

As well as backing up the database files, you should also back up the ViewDS update log on a daily basis. The log is critical for maintaining database integrity after a failure. After restoring from a backup, replaying the log will update the database according to all transactions committed since the backup was made.

Daily and weekly backups are recommended.

Daily backup (save)

ViewDS Management Agent

A daily backup involves exporting the update log and then saving the database:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and the **Text View** button.
4. In the **Database Maintenance** area of the right pane, click **Save**. The directory is saved to the `save` directory (by default, `${VFHOME}/save`). The command also creates a checkpoint of the update log. The update log's filename is `ulog`, and the checkpoint's filename is appended by a string, for example `ulog-84b70ab1510`.
5. From the command line, use the ViewDS *smerge* utility to sort the contents of the checkpoint file chronologically, and rename the file with today's date. To illustrate:

   ```
   smerge ulog-84b70ab1510 > ulog.20191211
   ```
6. Compress the renamed checkpoint file and the contents of the `save` directory.
7. Copy the compressed files to a backup device.

Command line

The steps for a daily backup are incorporated into the `dbbackup` script (see Backup scripts) are as follows:

1. Save the database:

   ```
   sdua -c save
   ```

   This command saves the database files to the `save` directory and creates a checkpoint of the update log.

   The update log's filename is `ulog`, and the checkpoint's filename is appended by a string, for example `ulog-84b70ab1510`.
2. Use the ViewDS *smerge* utility to sort the contents of the checkpoint file chronologically, and rename the file with today's date.

   To illustrate:

   ```
   smerge ulog-84b70ab1510 > ulog.20191211
   ```
3. Delete the duplicate update log (for example, ulog-84b70ab1510).
4. Compress the renamed checkpoint file and the contents of the `save` directory.
5. Copy the compressed files to a backup device.

Weekly backup (dump)

The weekly backup can be run at any time, but the recommendation is to run it when the load on the directory is minimal.

ViewDS Management Agent

A weekly backup involves exporting the update log and then dumping the database:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and the **Text View** button.
4. In the **Database Maintenance** area of the right pane, click **Dump**. The directory is dumped to the `dump` directory (by default, `${VFHOME}/dump`). The command also creates a checkpoint of the update log. The update log's filename is `ulog`, and the checkpoint's filename is appended by a string, for example `ulog-84b70ab1510`.
5. From the command line, use the ViewDS *smerge* utility to sort the contents of the checkpoint file chronologically, and rename the file with today's date. To illustrate:

   ```
   smerge ulog-84b70ab1510 > ulog.20191211
   ```
6. Compress the renamed checkpoint file and the contents of the `dump` directory. (Dump files typically compress by an approximate ratio of 10:1.)
7. Copy the compressed files to a backup device.

Command line

The steps for a weekly backup are incorporated into the `dbbackup` script (see Backup scripts) are as follows:

1. Dump the database:

   ```
   sdua dumpcmd
   ```

   Where `dumpcmd` is a file containing the dump command.

   The resulting `dib.*` files are written to the `dump` directory. They usually occupy around half the disk space used by the DSA's database files.

   The command also creates a checkpoint of the update log. The update log's filename is `ulog`, and the checkpoint's filename is appended by a string, for example `ulog-84b70ab1510`.
2. From the command line, use the ViewDS *smerge* utility to sort the contents of the checkpoint file chronologically, and rename the file with today's date. To illustrate:

   ```
   smerge ulog-84b70ab1510 > ulog.20191211
   ```
3. Compress the renamed checkpoint file and the contents of the dump directory (compressing the `dib.*` files typically gives a space reduction of around 10:1).
4. Copy the compressed files to a backup device.

Backup scripts

Linux

Daily and weekly backups can be performed using the `dbbackup` script, and fully automated using the cron and at commands in a shell script.

Windows

Daily and weekly backups can be automated by incorporating the steps described in the above subsections into a script that runs under the Windows Scheduler.

Restoring from a backup

Restoring save files

You can restore a database from save files using either the ViewDS Management Agent or command line.

ViewDS Management Agent

To restore from save files:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button.
4. Click **Stop DSA**. The DSA's status changes to `stopped`.
5. Replace the contents of the *Database directory* (by default, `${VFHOME}/data` or `%VFHOME%\data`) with the backed-up save files.
6. If any update logs were backed up after the dump files from which you have restored were created, then replay the update logs. Otherwise, click **Start DSA**. The DSA status changes to `running`.

Command line

To restore from save files:

1. The RAS is usually configured to stop the DSA as part of its own shut-down process. Enter the following command to stop the RAS (and therefore the DSA):
   ```
   ras stop
   ```
2. Replace the contents of the database directory (by default, `${VFHOME}/data` or `%VFHOME%\data`) with the backed-up save files.
3. Start the RAS (and therefore the DSA) by entering the following command:
   ```
   ras
   ```

Restoring dump files

You can restore a database from dump files using either ViewDS Management Agent or the command line.

ViewDS Management Agent

To restore from dump files:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click the server.
3. In the right pane, click the **Status** tab followed by the **General** tab, and then the **Text View** button.
4. Click the **Stop DSA** . The DSA status changes to `stopped`.
5. Replace the contents of the `dump` directory (by default, `${VFHOME}/dump`) with the backed-up dump files.
6. In the **Database Recovery** area of the tab, click **Load**. The DSA's database is emptied and reloaded with the contents of the dump files.
7. If any update logs were backed up after the dump files from which you have restored were created, then replay the update logs. Otherwise, click **Start DSA**. The DSA status changes to `running`.

Command line

To restore the database from a weekly backup that uses the dump command:

1. Load the database (see Bulk loading a database).
2. To update the database according to the transactions that have been committed since the backup was made, follow the steps below to replay the update log.

Replaying the update log

If there are any update logs that were backed up after the save or dump files from which you have restored were created, you will need to replay these logs. This will update the database with the transactions that were committed after the save or dump backup was made.

To replay the update logs:

1. Copy the appropriate update logs to the `load` directory (by default, `${VFHOME}/load`).
2. If the DSA is running, stop it.
3. From the command line, navigate to the `load` directory and enter the following:

   ```
   vfload ../load/*
   ```

   The update logs are applied to the database.
4. Start the DSA.

Key concepts: Schema

This section introduces the concepts required to work with ViewDS schema, and describes how to manage schema through the ViewDS Management Agent. It also includes high-level guidance to help you adapt ViewDS to your requirements.

It covers the following topics:

Introduction to schema

A schema is a set of rules that controls all aspects of what can be stored in a directory. Every directory has schema that defines, for example, what kind of entries appear in the Directory Information Tree (DIT), where a particular kind of entry can appear in the DIT and the entry's attributes.

The following concepts are discussed below:

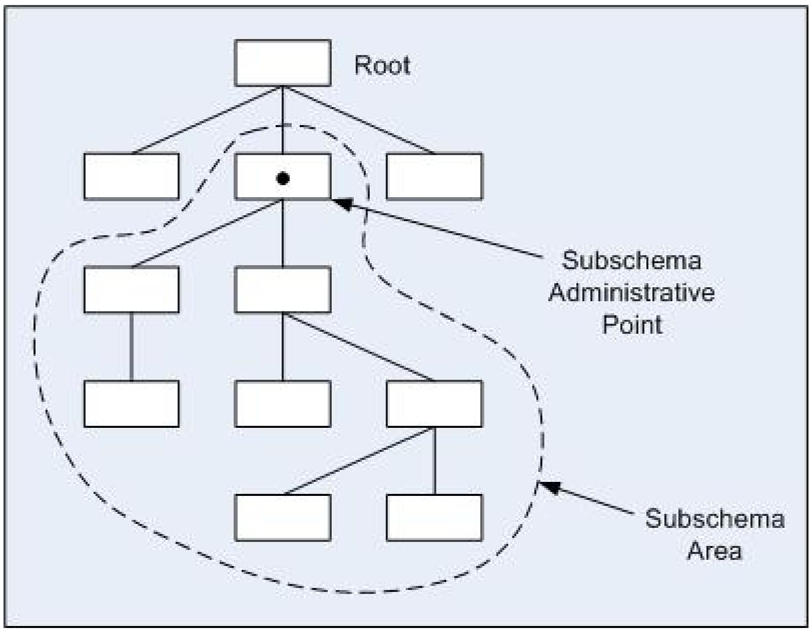
- Subschema area and subschema administrative point
- Subschema defines object classes
- Identifying schema with OIDs
- Predefined and built-in schema

Subschema area and subschema administrative point

A subschema area is the area of a DIT where a particular set of schema definitions apply. The entry at the top of the subschema area is called the subschema administrative point.

As shown below, the scope of a subschema area extends to the entries in the subtree below a subschema administrative point.

A DIT can have as many subschema areas as required, although it is usual to have just one.



Subschema defines object classes

Every entry in a DIT has a structural object class defined in a subschema area. Among other things, the definition for a structural object class includes the following:

- **Attributes**
  An object's attributes are either mandatory or optional. A valid entry must have values for its mandatory attributes, but may or may not have values for its optional attributes.
- **Content rule**
  Allows you to add attributes to a standard structural object class (in addition to those inherited from its superclass). It also allows you to add an auxiliary object class to a standard or non-standard structural object class.
- **Name form**
  Defines which attributes appear in an entry's Relative Distinguished Name (RDN). The RDN is displayed in the DIT and in search results. An entry's Distinguished Name (DN) uniquely identifies the entry and comprises its RDN plus the DN of its superior.
- **Structure rule**
  Defines where entries of the object class can appear in the DIT.

Identifying schema elements with OIDs

An object identifier (or OID) is a sequence of integers that identifies an element of schema - such as an object class, attribute, name form or matching rule.

An OID describes a tree, each node in the tree represents an object and a naming authority responsible for allocating the next level of the tree.

The ViewDS Management Agent allows you to define OID arcs. This ensures that ViewDS will automatically allocate an OID whenever you create a new schema definition.

Example object identifier

The OID that represents the attribute type `streetAddress` is `2.5.4.9`.

It is constructed as follows:

- by international agreement, the object identifier `2` has been allocated to joint ISO-ITU administration
- this administration has allocated the arc `5` to directory, which gives the OID `2.5`, and has given administration of this OID to the X.500 committee
- the X.500 committee has allocated the arc `4` to attributes, which gives `2.5.4`
- the committee has assigned the arc `9` to the attribute `streetAddress`, which gives `2.5.4.9`

Object identifier prefixes

Typically, the object identifiers used in ViewDS have one of the following six prefixes, plus any other prefixes introduced by the ViewDS system administrator.

Prefix	Definition
ds	2.5
mhs	2.6
vf	1.3.32.0.1
Ads	1.2.36.79672281.1
Xed	1.3.6.1.4.1.21472
Vds	1.3.6.1.4.1.21473 5

Creating object identifiers

To be allocated an object identifier, apply to one of the international standards bodies (ITU or ISO) or to your national standards body (for example, Standards Australia). The allocated object identifier can then be used as the root for new object identifiers when you create new schema elements.

Every Australian organization has a default object identifier, which includes their Australian Business Number (ABN):

`1.2.36.`*`abn`*

Where *abn* is the ABN.

Alternatively, your ViewDS vendor can assign object identifiers from its own arc if you do not have (or wish to use) your own. Contact your ViewDS vendor for further details.

Object identifier organization

The following table shows the recommended organization for object identifiers. It is parallel to the organization used by X.500, which involves using the arc below your organization’s arc to represent the category-of-information object (see X.501 Annex A for a full list).

Category	Arc
attribute type	4
object class	6
name form	15
matching rule	13

Predefined and built-in schema

ViewDS includes predefined schema for you to import. You can import one or more of definitions from a predefined schema - attributes, object classes, content rules, name forms, structure rules, matching rules, etc.

Because an imported definition is part of 'standard' schema, you should not modify it. However, when the ViewDS Management Agent recognizes a schema definition as being 'standard', you can modify a subset of its properties such as the description and name. These definitions that are recognized as 'standard' are referred to as 'built-in' schema.

More about schema

As already mentioned, every entry in a DIT has a structural object class defined, which includes the following:

- **Attributes** – an object's attributes are either mandatory or optional.
- Content rule – allows you to add attributes to a standard structural object class and also allows you to add an **auxiliary object class** to a structural object class.
- **Name form** – defines which attributes appear in an entry's Relative Distinguished Name.
- **Structure rule** – defines where entries of the object class can appear in the DIT.

Subschema attributes

Attributes are defined independently of the object classes that use them. An attribute definition can be optimized for searches by defining matching rules, approximate matching rules and indexes.

Other key concepts for attributes are described below.

Complex syntax and components

By supporting complex syntax, ViewDS allows you to extend a schema to include complex syntaxes (such as certificates and XML documents) so that their components can be searched. The components are the individual primitive types, which combined constitute a complex syntax.

To illustrate, consider a Human Resources department that stores employees' resumes as XML documents in a ViewDS directory. This would allow people to perform searches on specific areas (components) of the resumes. They might, for example, search the qualifications area to find the employees who have a Masters of Business Administration.

Collective attributes

You can define that an attribute is 'collective', which means it has one value for all its instances in a subtree. This might be useful, for example, for common information such as a departmental fax or phone number.

When a user modifies a collective attribute of an entry, the new value is published to all subordinates of the entry.

Attribute extensions

You can define extensions for an attribute type so that:

- its values are written to a separate file when the database is dumped, which is useful for attributes that store, for example, documents or images as you can open the files with an appropriate application.
- its values arehashed when stored in the database, which is useful for passwords.
- it tracks an entry by its DN, which means that it keeps track of an entry when it is moved in the DIT.

Auxiliary object classes

Every entry in a directory is an instance of a structural object class that describes it through a set of attributes. For example, every employee might be an instance of a structural object class called `person` whose attributes include `name`, `address` and `phoneNumber`.

As well as the structural object class, there are two other kinds of object class: abstract and auxiliary. The abstract object class is a standard concept in object-oriented theory, and is not discussed here any further.

The auxiliary object class, however, has the following characteristics:

- An instance of an auxiliary object class cannot exist on its own, it must be connected to an instance of a structural object class.
- An auxiliary object class can be used to tag information to an entry temporarily.
- The possible locations of an auxiliary object class are defined by adding it to a structural object class's content rule. (The location of a structural object class is fixed according to the hierarchy defined by its structure rules.)
- An auxiliary object class can be used to identify entries as part of a Basic Access Control policy.

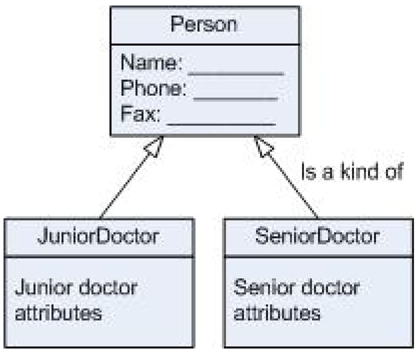
Example scenario

To illustrate how an auxiliary object class might be used, consider the requirement for a directory to have two new kinds of entry: one for junior doctors and one for senior doctors. The information they both need to store includes the usual sorts of things for people, such as email address and phone number, along with information that is specific to either a junior doctor or a senior doctor.

Two approaches to fulfilling this requirement are described below.

Using structural object classes

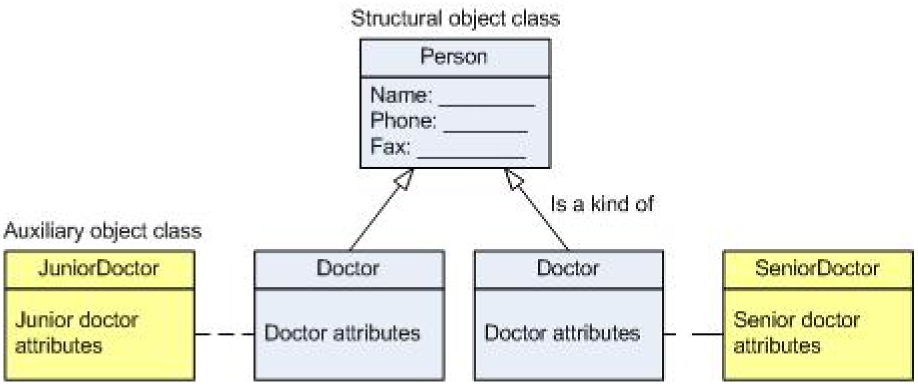
One approach is to create two structural object classes, `JuniorDoctor` and `SeniorDoctor`, each with a similar set of attributes.



However, because an entry cannot change its class, a problem would arise using this approach when a junior doctor became a senior doctor.

Using auxiliary object classes

An alternative is to create a structural object class, `Doctor`, and two auxiliary object classes, `JuniorDoctor` and `SeniorDoctor`, each containing attributes specific to either a junior or senior doctor.



The advantage of this approach becomes apparent when a junior doctor is promoted. The only action required is to change their auxiliary class from `JuniorDoctor` to `SeniorDoctor`.

Making auxiliary object classes available

An auxiliary object class is only available for an entry whose structural object class with a content rule to associate it with the auxiliary object class.

About name forms

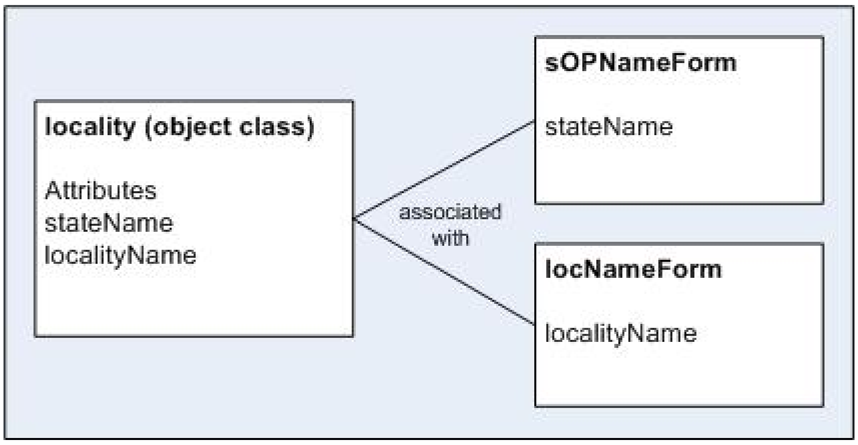
A name form defines which attributes appear in an entry's Relative Distinguished Name (RDN). The RDN is displayed in the Directory Information Tree (DIT) and in search results.

An entry’s Distinguished Name (DN) uniquely identifies the entry and comprises its RDN plus the DN of its superior.

One object class with multiple name forms

Most object classes have just one name form.

There are, however, scenarios where an object class needs more than one. To illustrate, consider the `locality` object class which has two name forms associated with it.

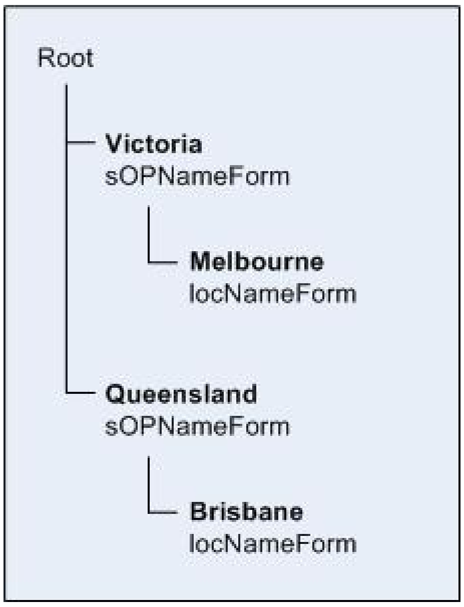


The name forms are:

- `forename` sets an entry's RDN to the value of its `state Name` attribute
- `locNameForm` sets an entry's RDN to the value of the `localityName` attribute

Each name form is used according to where the entry appears in the DIT.

Consider the following DIT, containing four entries of the `locality` object class (Victoria, Melbourne, Queensland and Brisbane).

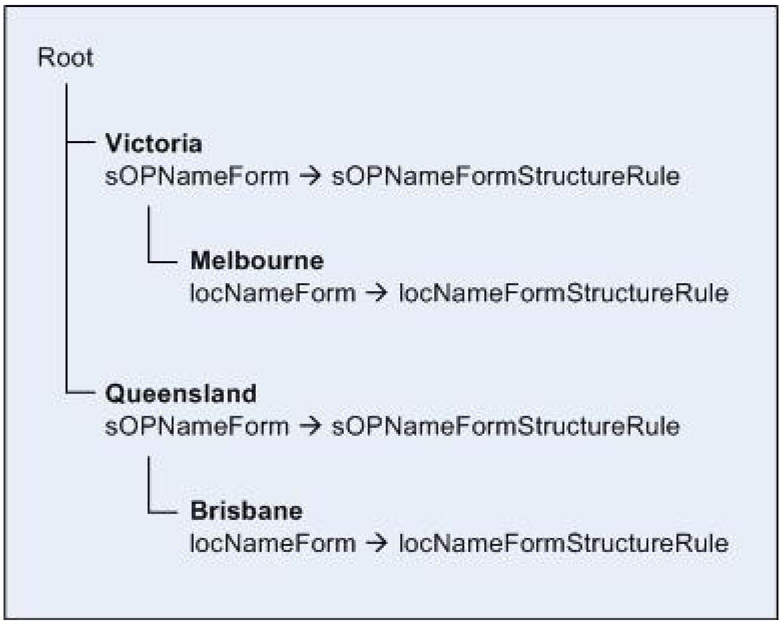


The Victoria and Queensland entries use `sOPNameForm` - the RDN is the value of the `stateName` attribute.

The Melbourne and Brisbane entries use `locNameForm` - the RDN is the value of the `localityName` attribute.

How does the schema define which name form should be used?

Every object class has at least one name form associated with it. In turn, every name form has at least one structure rule, as shown below.


Root
Victoria
sOPNameForm → sOPNameFormStructureRule
Melbourne
locNameForm → locNameFormStructureRule
Queensland
sOPNameForm → sOPNameFormStructureRule
Brisbane
locNameForm → locNameFormStructureRule

There are two structure rules in the above example:

- `sOPNameFormStructureRule` which applies to a `locality` entry if its RDN is `stateName` (Victoria and Queensland in the DIT).
- `locNameFormStructureRule` which applies to a `locality` entry if its RDN is `localityName` (Melbourne and Brisbane in the DIT).

Each structure rule has a list of 'superior structure rules':

- `sOPNameFormStructureRule` has a list containing no superior structure rules
- `locNameFormStructureRule` has a list containing one superior structure rule `sOPNameFormStructureRule`

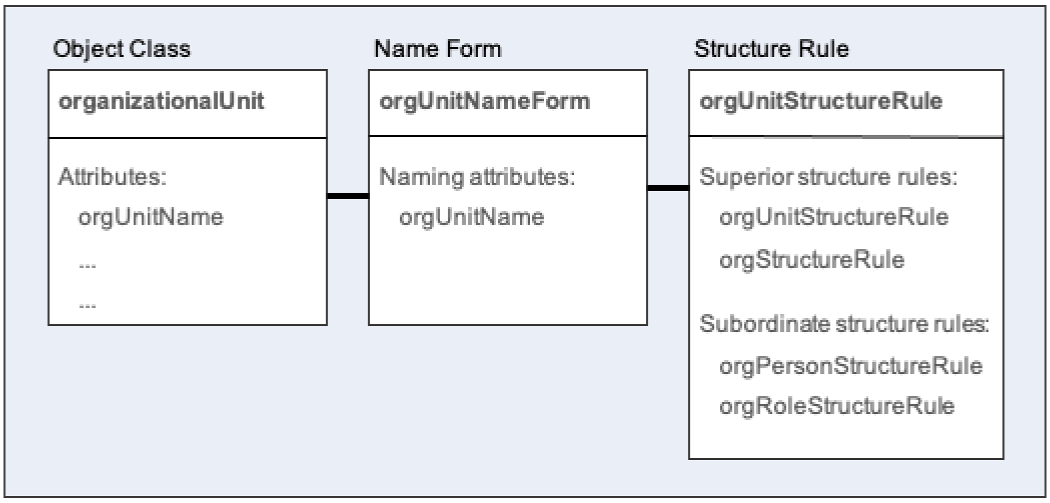
The list of superior structure rules determines where an entry can appear in the DIT hierarchy:

- Entries that use `sOPNameFormStructureRule` (Victoria and Queensland) must be subordinate to the subschema administrative point.
- Entries that use `locNameFormStructureRule` (Melbourne and Brisbane) must be subordinate to an entry that uses `sOPNameFormStructureRule` (Victoria and Queensland).

About structure rules

Every structural object class must have at least one structure rule. A structure rule defines where an entry of an object class can appear in the Directory Information Tree (DIT) according to the name form it uses.

To illustrate, consider the `organizationalUnit` class which has a structure rule called `orgUnitStructureRule`.



The structure rule includes two list: 'superior structure rules' and 'subordinate structure rules'.

Superior structure rules

The list of 'superior structure rules' identifies directory entries that can be superior to an entry using the structure rule.

For the above illustration, an `organizationalUnit` entry must be subordinate to an entry whose class uses either `orgUnitStructureRule` or `orgStructureRule`.

Also to consider:

- **Own superior**. As shown above, a structure rule can appear in its own list of 'superior structure rules'. This means that an `organizationalUnit` can be subordinate to another `organizationalUnit` in the DIT.
- **No superiors**. If the list of ‘superior structure rules’ were empty, then an `organizationalUnit` could only appear below the subschema administrative point.

Subordinate structure rules

The list of 'subordinate structure rules' identifies the directory entries that can be subordinate to an entry using the structure rule. (If a potential subordinate has 'superior structure rules' in its structure rule, then there will be other classes of entry below which it can appear.)

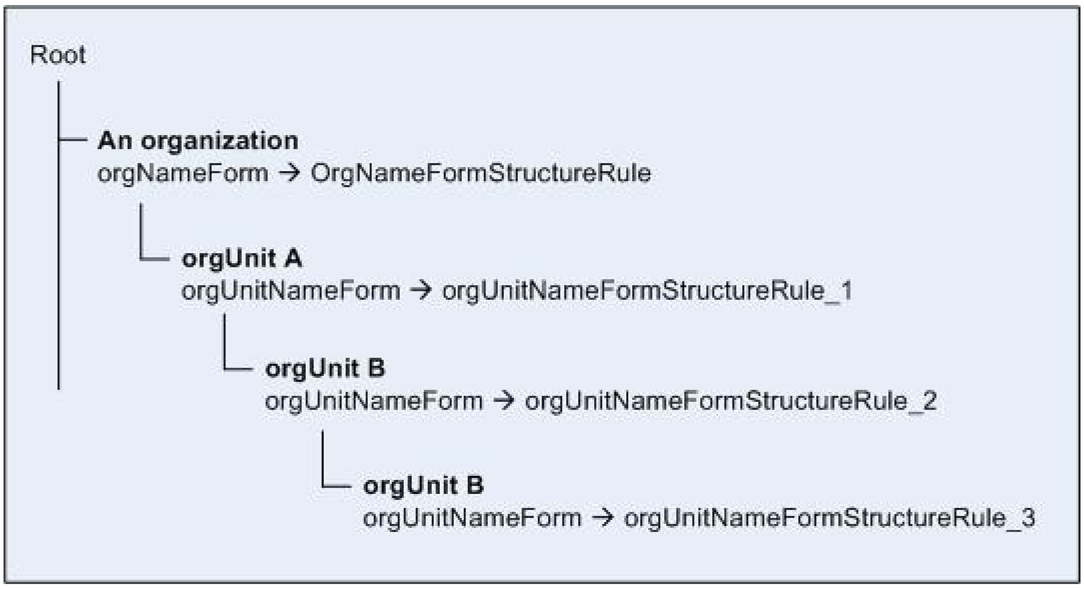
For the above illustration, an entry using `orgPersonStructureRule` or `orgRoleStructureRule` can be subordinate to an `organizationalUnit` entry.

Multiple structure rules to constrain a hierarchy

You can create a constrained hierarchy by associating multiple structure rules with the same name form.

An example of a constrained hierarchy is when there can be a maximum of three levels for a particular object class, say `orgUnit`, below another object class, `organization`.

This example is shown in the following illustration.


Root
An organization
orgNameForm → OrgNameFormStructureRule
orgUnit A
orgUnitNameForm → orgUnitNameFormStructureRule_1
orgUnit B
orgUnitNameForm → orgUnitNameFormStructureRule_2
orgUnit B
orgUnitNameForm → orgUnitNameFormStructureRule_3

`orgUnitNameForm` has three structure rules:

- `orgUnitNameFormStructureRule_1` which has one superior structure rule `organizationNameFormStructureRule` (an entry that uses this structure rule can only be below a `organization` entry)
- `orgUnitNameFormStructureRule_2` which has one superior structure rule `orgUnitNameFormStructureRule_1` (an entry that uses this structure rule can only be below an `orgUnit` entry that uses `orgUnitNameFormStructureRule_1`)
- `orgUnitNameFormStructureRule_3` which has one superior structure rule `orgUnitNameFormStructureRule_2` (an entry that uses this structure rule can only be below an `orgUnit` entry that uses `orgUnitNameFormStructureRule_2`)

Working with schema

This subsection introduces you to managing schema through the ViewDS Management Agent. It includes high-level overviews to help you plan and implement a schema according to your requirements.

Viewing schema through the VMA

1. At the bottom of the left pane, click **Global DIT View**. The DIT tab is displayed.
2. In the DIT tab, click the **Deltawing** entry at the top of the DIT.

   The following tabs are now displayed in the right pane:

   - **Attributes** – shows the attributes of the Deltawing entry.
   - **Schema** – this tab is displayed because the Deltawing entry is a subschema administrative point.
3. In the right pane, click the **Schema** tab.

   The Schema tab includes buttons to import and export schema definitions, and contains the following subordinate tabs:

   - **Attributes** – manage the attribute definitions in the subschema area.
   - **Object Classes** – manage object class definitions.
   - **Words** –manage noise words, synonyms and truncated words, which help optimize searches performed by users.
   - **Definitions** – add XML syntax, which can then be used to create an attribute definition.
   - **DUA** – define how different elements of schema are presented by Access Presence.
   - **Indexing** – manage the indexing for attribute definitions.
   - **Matching Rules** – manage the matching rules assigned to attributes in the subschema area.
   - **OID Arcs** – manage Object ID (OID) arcs for attributes, object classes, name forms and matching rules. Defining an OID arc ensures that ViewDS will automatically allocate an OID whenever you create a new schema definition.

Viewing an object class definition

4. Click the **Object Classes** tab. A list of the object classes in the subschema area is displayed.
5. In the **Object Classes** tab, double-click **organizationalPerson**. The organizationalPerson Properties window is displayed.
6. In the window, click the **Attributes** tab. The tab includes two boxes: one for the mandatory attributes, and another for the optional attributes.
7. To view the mandatory attributes, click the **Inherited Mandatory Attributes** button. The Inherited Mandatory Attributes window shows you the attributes that must have values when a new entry of this structural object class is created.

8. Click **OK** to close the Inherited Mandatory Attributes window.
9. Click the **Rules** tab. This tab has three areas: Content Rules, Name Forms and Structure Rules.

Viewing the content rule

10. In the Content Rules box, click **deltawingOrganizationalPerson** and then click the **View** button. The Content Rules window is displayed, which has the following areas:
    - Auxiliary Object Classes – the auxiliary object class specialUser is listed, which means it can be associated with an organizationalPerson entry.
    - Mandatory Attributes – the mandatory attributes in addition to those inherited from organizationalPerson's superclass.
    - Optional Attributes – the optional attributes in addition to those inherited from organizationalPerson's superclass.
    - Precluded Attributes – the attributes that cannot be added to the organizationalPerson's superclass.
11. Click **Cancel** to close the Content Rules window, and then **Cancel** again to close the organizationalPerson Properties window.

Identify schema requirements

The following considerations will help you define the requirements for a subschema:

1. What entries do you want to store? For example, people, departments, resources, etc.
2. What are the sub-categories, if any, for each of the entries? For example, people might divide into contract and permanent staff.
3. What attributes do you want to store in each entry?
4. Should each attribute have a primitive or complex syntax (for example, XML)?
5. Which attribute should be used to uniquely identify each entry?
6. What structure is your data currently stored in?
7. If you intend to use Access Presence, can data in the existing structure be displayed in a browser?
8. How will the directory data be maintained? The structure of the data is important in terms of maintenance because the ViewDS Management Agent allows you to delegate responsibility for maintaining the data in a particular area of the directory. The benefit being that there will be more attention to data management, resulting in more accurate data throughout the directory.

Adapt to your requirements

The following high-level overview will help you adapt a subschema area to your requirements:

1. If the Deltawing subschema matches, or partially matches, your requirements:
   a. Delete the entries in the DIT (see Delete a subtree in the ViewDS Management Agent help).
   b. Go to step 3.
2. If the Deltawing schema does not match your requirements:
   a. Empty the Deltawing directory (see ViewDS Management Agent help).
   b. Identify the predefined schema (see ViewDS Management Agent help) that provides the closest match to your requirements.
   c. Add a first level entry to the empty DIT (see ViewDS Management Agent help). During this task, import the schema you identified in the previous step and create the default Basic Access Controls.
3. Modify the schema according to your requirements. For example, you might decide to do one or more of the following (all of which are described in the Management Agent help):
   - Import additional schema elements that match to your requirements.
   - Add an attribute to a standard object class Modify an attribute.
   - Create an attribute.
   - Modify an object class.
   - Create an object class.
4. Optionally, bulk load data into your modified schema.

Understanding search optimization

If you expect an attribute to appear in a user’s search criteria, it should be optimized for searches.

The following can be used to optimize an attribute for searches:

- Matching rules
- Approximate-matching rules
- Word lists
- Indexing

The above can also be assigned to components of an attribute with a complex syntax (for example, an XML document or digital certificate), allowing users to search on the components.

Matching rules

ViewDS uses matching rules when it searches attribute values and when attempting an equality, ordering or substring match. Each matching rule has an assertion syntax, which defines the syntax of the information provided in a user's search criteria.

ViewDS supports matching rules defined in the X.500 (1993) and X.400 (1988) recommendations, along with the ViewDS-specific rules described in the *ViewDS Directory Server: Technical Reference Guide*.

There are three kinds of matching rule:

- equality
- ordering
- substring

Equality matching rules

ViewDS uses an attribute's equality matching rule when:

- adding a value to check whether the value already exists
- deleting a value to identify the value to be deleted
- comparing a value using the compare operation

If no equality matching rule is specified, ViewDS cannot distinguish between values of the attribute. Modify operations will work, but operations that involve matching individual values will not.

ViewDS references an attribute's noise words when equality matching.

Ordering matching rules

ViewDS uses an attribute's ordering matching rule when performing a greater-than or less-than search, and when ordering search results.

Substring matching rules

ViewDS uses an attribute's substring matching rule when searching for a substring in an attribute's value.

Approximate-matching rules

ViewDS uses approximate-matching rules when a user searches for an approximate match to their search criteria. These rules only apply to attributes, and components of complex attributes, with a string-type syntax.

The approximate-matching rules are:

- Phonetic – for example, the search criteria 'pane' would match the attribute value 'payne'
- Typing correction – 'Dircetor' would match 'Director'
- Synonym – 'Bob' would match 'Robert', and 'road' would match 'street' (there is more information about this rule below)
- Prefix – 'thom' would match '*Thom*as', '*Thom*son' and '*Thom*kins' (ViewDS references an attribute's noise words when prefix matching)
- Suffix – 'son' would match 'Thom*son*', 'Hodg*son*' and 'Peter*son*'
- Abbreviation – 'NSW' would match 'New South Wales', and 'DOF' would match 'Department of Fisheries' (there is more information about this rule below)

For each of the above, there are two approximate-matching rules:

- keyword matching, which looks at each word in a string individually
- non-keyword matching, which treats an attribute’s value as a discrete string

To illustrate, for the keyword prefix rule, a search criteria of 'thom' might return 'Thompson', 'Robert Thompson' and 'Thomas Shipman'. For the non-keyword prefix rule, the same search would only return 'Thompson'.

There are two more approximate-matching rules, which perform keyword matching only:

- Equality – 'self' would match 'Will *Self*', 'John *Self*' and '*Self*ish Alfonzo the Third'
- Phonetic Mandarin – performs phonetic matching on Mandarin pronunciation of simplified/traditional Chinese characters

Synonym approximate matching

For synonym approximate matching, you must declare sets of synonyms for each attribute. Otherwise, selecting synonym approximate matching has no effect.

ViewDS references an attribute's noise words when synonym approximate matching.

Abbreviation approximate matching

When a user's search criteria for an attribute is an abbreviation, ViewDS attempts to match the search criteria against each value of the attribute. ViewDS does this by first removing any noise words you have defined for the attribute. Then, if a value contains:

- one keyword – ViewDS does not abbreviate it
- two keywords – ViewDS abbreviates each keyword (automatically or using a truncated word you have defined) and concatenates the abbreviations
- three keywords – ViewDS abbreviates the value to the first letter of each keyword

To illustrate, if a user searches on the abbreviation 'DOF', then ViewDS might return '**D**epartment **o**f **F**inance' and '**D**epartment **o**f **F**isheries'. If you have declared 'of' as a noise word, however, the abbreviation 'departFish' would also return 'Department of Fisheries'.

The ViewDS Management Agent allows you to create a set of truncated words for an attribute. ViewDS uses these truncated words instead of automatically generating abbreviations when there are two keywords in the attribute's value.

NOTE: The above methods of abbreviating an attribute's values are also used when an Abbreviated Hierarchy Name is displayed by Access Presence.

Word lists

To improve approximate matching, ViewDS allows you to define sets of synonyms, noise words and truncated words for an attribute (or a component of an attribute) that has a string syntax.

Noise words

A noise word is a word - such as 'the' or 'and' - that is so common that it is usually of little use for searching or indexing.

ViewDS ignores the noise words when it performs the following on an attribute:

- indexing
- keyword equality matching
- prefix approximate matching
- abbreviation approximate matching

You should create noise words for an attribute that has any of the above assigned to it.

Synonyms

You can create sets of synonyms for an attribute. The words in each set are treated as equivalent when a user requests an approximate match on one of the words in the set. For example, a set of synonyms for an attribute might be ’high school’ and ’secondary college’. A search on ’high school’ would return matches on both ’high school’ and ’secondary college’.

Truncated words

The ViewDS Management Agent allows you to create sets of truncated words for an attribute.

ViewDS uses these truncated words instead of automatically generated abbreviations when there are two keywords in the attribute's value. They are also used when Access Presence displays Abbreviated Hierarchy Names.

Indexing

Indexing the attributes in a directory results in faster searches.

An attribute or component that has approximate-matching rules assigned should also have indexing assigned. Fortunately, ViewDS recommends the most appropriate indexing according to an attribute's syntax and matching rules.

ViewDS references an attribute's noise words (see above) when indexing.

For a description of the indexing options available, see the *Directory Server: Technical Reference Guide*.

Optimizing for searches

The following high-level overview will help you optimize for searches:

1. Identify the attributes that you expect users to search on.
2. For these attributes (see the ViewDS Management Agent help for each task):
   a. assign approximate-matching rules
   b. assign indexing
   c. create a set of noise words
3. For each attribute to which you assigned the synonym approximate-matching rule, create a set of synonyms (see the ViewDS Management Agent help).

Key concepts: Security

This section introduces the concepts required to work with ViewDS security, and describes how to manage security through the ViewDS Management Agent (VMA).

It covers the following topics:

- Introduction to security
- ViewDS Access Control
- X.500 Basic Access Control
- Working with X.500 Basic Access Controls

Note that ViewDS Directory also includes XACML access control, which is discussed in the chapter Key concepts: XACML.

Introduction to security

There are two stages of user access:

- authentication
- authorization

For more information about security, including password policies, see the *ViewDS Directory Server: Technical Reference Guide*.

Authentication

Authentication involves establishing a user's identity through credentials recognized by the server.

A Directory User Agent (DUA) user can connect (bind) to ViewDS using one following levels of authentication: anonymous, simple or strong authentication.

Anonymous

The user connects anonymously without a user name or password.

Simple

The user connects with a Distinguished Name (DN) and password, which is stored in their directory entry. To implement simple authentication each user must be assigned a password (see the VMA help).

Strong

The user is authenticated through a certificate-based mechanism (such as those described by X.509 and TLS).

A user's certificate should be stored as a certificate attribute in their DIT entry.

All certificates must be issued by a Certification Authority. The DSA must have a list of the Certification Authorities (CAs) that it trusts regarding the user and CA certificates they issue. The CAs in this list are the 'trust anchors'.

The trust anchors may certify other CAs (intermediate CAs) which may in turn issue certificates or certify further subordinate intermediate CAs. Each intermediate CA that has issued certificates must have a DIT entry that stores its CA certificate.

To manage the list of CAs, see the VMA help.

Authorization

After authentication, authorization controls a user's access. It is implemented using access controls to govern which areas of the directory a user is authorized to access.

The identity established during authentication is usually also used to evaluate authorization decisions. However, this is not always the case (for example, when proxy authorization is used).

ViewDS provides three access-control schemes:

- ViewDS Access Control
  A simple security scheme where each user is assigned one of four levels of access control after anonymous authentication.
- X.500 Basic Access Control
  A much more versatile option that allows you to set up any number of Access Control Items (ACIs) to match the different roles of your directory users. It also allows you to group users and assign them the same ACI.
- XACML Based Access Control (XBAC)
  Allows XACML policy that conforms to the XACML Version 3.0 standard to be applied to the directory. This capability can be extended to allow XACML policy to be applied to applications external to ViewDS Directory (an Access Sentinel licence is required).

ViewDS Access Control

This is a simple access-control scheme that applies to a ViewDS directory by default.

Each user is authenticated anonymously and assigned a level of ViewDS Access Control:

- None
  User has no access to the directory, apart from being able to view their own DN.
- Read Access
  User can read all attributes, apart from another user's password, and update their own password. Users have Read Access by default.

- Update
  User can read and update all attributes, apart from privileges and other users' passwords, in a specific subtree.
- Admin Access
  User can read and update all attributes, including all passwords, in a specific subtree.
- Super-user Access
  User can read and update all attributes, including all privileges and passwords, in the entire directory.

If a user does not have one of the above assigned, the default of Read Access applies.

ViewDS Access Control and the VMA

To implement ViewDS Access Control through the ViewDS Management Agent (VMA):

1. Set up anonymous authentication and set the default level of ViewDS Access Control – in the ViewDS Management Agent help, see the topic Configure anonymous privileges.
2. Add the privilege attribute to each user's entry – in the ViewDS Management Agent help, see the topic Configure anonymous privileges.

   NOTE: If a user's entry does not have the privilege attribute, the default level of ViewDS Access Control applies.

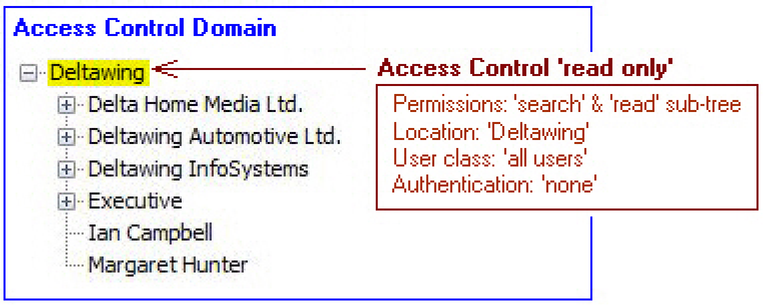
Basic Access Control

ViewDS implements a X.500 Basic Access Control scheme. It also extends the model to simplify administration and provide greater flexibility.

Basic Access Control has two main components:

- Access Control Domain
  An area of the Directory Information Tree (DIT) containing one or more Access Control Items.
- Access Control Item (ACI)
  An ACI defines permissions that grant access to entries and attributes in the DIT, and also defines to which users the permissions apply.

In the following illustration, there is one Access Control Domain that contains an ACI called 'read only'. It allows all users to search and view the entries in the Deltawing subtree.



The key concepts for Basic Access Control are described in more detail below.

Access Control Domain

An Access Control Domain is a specific area of a DIT that contains one or more ACI. Its border is the scope of the ACIs within it.

Usually, just one Access Control Domain is required. An exception, however, is when access needs to be managed autonomously for different areas of a DIT.

The entry at the top of an Access Control Domain is called the administrative point. This is the point from where you can manage the ACIs in the Access Control Domain.

Access Control Item

An Access Control Item (ACI) comprises the following:

- Permissions
- Location
- User class
- Authentication

Permissions

An ACI has a set of permissions that **grant access** to entries and their attributes. (The permissions can also deny access. However, this should be avoided because users are denied access to the directory by default. Keeping track of ACIs becomes very complicated if some grant access and others deny access to the same areas of the DIT.)

The permissions have a scope, which is either:

- individual entries
  When the scope of the permissions is individual entries, the ACI is termed an **Entry ACI**.
- subtrees
  When the scope of the permissions is subtrees, the ACI is termed a **Subtree ACI**. A Subtree ACI can also have a refinement to identify, for example, all 'people' entries in a subtree.

An Entry ACI should only be used when the location of an ACI cannot be identified by a subtree.

Location

An ACI has one or more locations in the DIT, which is **where** its permissions apply. An ACI can be located at either:

- individual entries
  An Entry ACI can be located at one or more individual entries – its permissions will apply to each entry.
- subtrees
  A Subtree ACI can be located at the top of one or more subtrees – its permissions will apply to all entries in each subtree. If the Subtree ACI has a refinement, its permissions will apply to the entries identified by the refinement.

By allowing one ACI to have many locations in a DIT, ViewDS simplifies directory management significantly.

User class

A user class is a set of directory users. It defines **who** the ACI's permissions apply to at each of its locations. The same user class can apply at all locations; or different a user class can apply at different locations.

You can specify a user class by selecting one or more of the following:

- individual user entries
- a subtree of users
- a group of users
  A group is an entry of an object class with a multivalued attribute that holds each group member. The attribute has the syntax `DistinguishedName`.
- a search filter to identify users
  You can create a role-based ACI by specifying that membership of the user class is determined by information in a user's entry. You can also specify conditions for membership based on operational attributes that appear in every entry. There are operational attributes that provide the following: date and time at the DSA, time at the DSA, day of the week at the DSA, the time zone the user is in, date and time in the user's time zone, time in the user's time zone, day of the week in the user's time zone.

An Access Control Domain frequently contains many ACIs with many user classes. When a user belongs to several user classes, ViewDS merges the ACIs that apply to the user. The result might be that several ACIs apply or that some of the ACIs no longer apply to the user.

Authentication

An ACI also has a minimum level of authentication.

For an ACI that **grants** access to an entry, a user can only access the entry if:

- they connected to the directory with at least the minimum level of authentication; and
- they are in the ACI's user class for the entry.

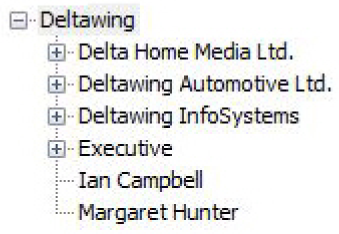
For an ACI that **denies** access to an entry, a user is denied access if they are in the ACI's user class for the entry. (Their level of authentication is irrelevant.) However, the same ACI grants access to a user if:

- they connected to the directory with at least the minimum authentication; and
- they are NOT in any of the ACI's user classes.

NOTE: It is advisable to avoid using ACIs that deny access.

Example 1: One location with one user class

Consider the example directory, Deltawing, supplied with ViewDS:



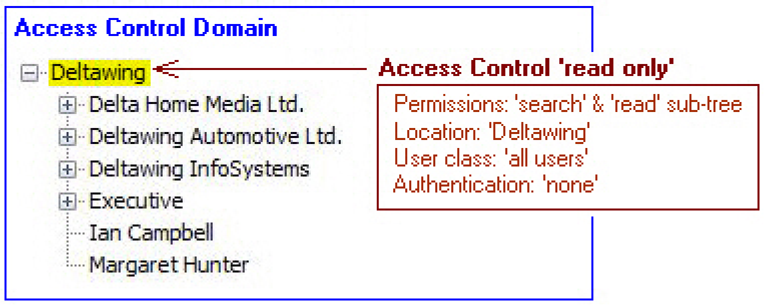
Now consider a requirement that every user should be able to search and view all entries in the directory.

Fulfilling the requirement

You can fulfill this requirement by creating an Access Control Domain at the top of the DIT, and then creating an Access Control Item (ACI) at the same location with the following properties:

- Name - the name of the ACI is 'read only' (the name of an Access Control Item should reflect its purpose by saying what it protects or who it applies to).
- Permissions - the ACI grants 'search' and 'read' permissions on the entries in the subtree where it is located. (It is a Subtree ACI.)
- Locations - the ACI has one location, Deltawing, which is the subtree where the permissions apply.
- User class - the user class includes all users in the Deltawing subtree.
- Authentication - the level of authentication required is 'none'.

To illustrate:



Example 2: Multiple locations with one user class

In this example, there is the requirement for one person to coordinate the Deltawing meeting rooms. This meeting-room coordinator should be Tony Liggett at Deltawing InfoSystems, and he should be able to 'search', 'read' and 'modify' the following entries:

- Large Conference Room (under the Avalon Factory at Deltawing Automotive)
- Large Meeting Room (under Deltawing Automotive)
- Sales Meeting Room (under Sales at Deltawing Automotive)
- Fishbowl Board Room (under Executive)

Another requirement is that the coordinator must connect to the directory with 'strong' authentication (their identity is confirmed through a security certificate).

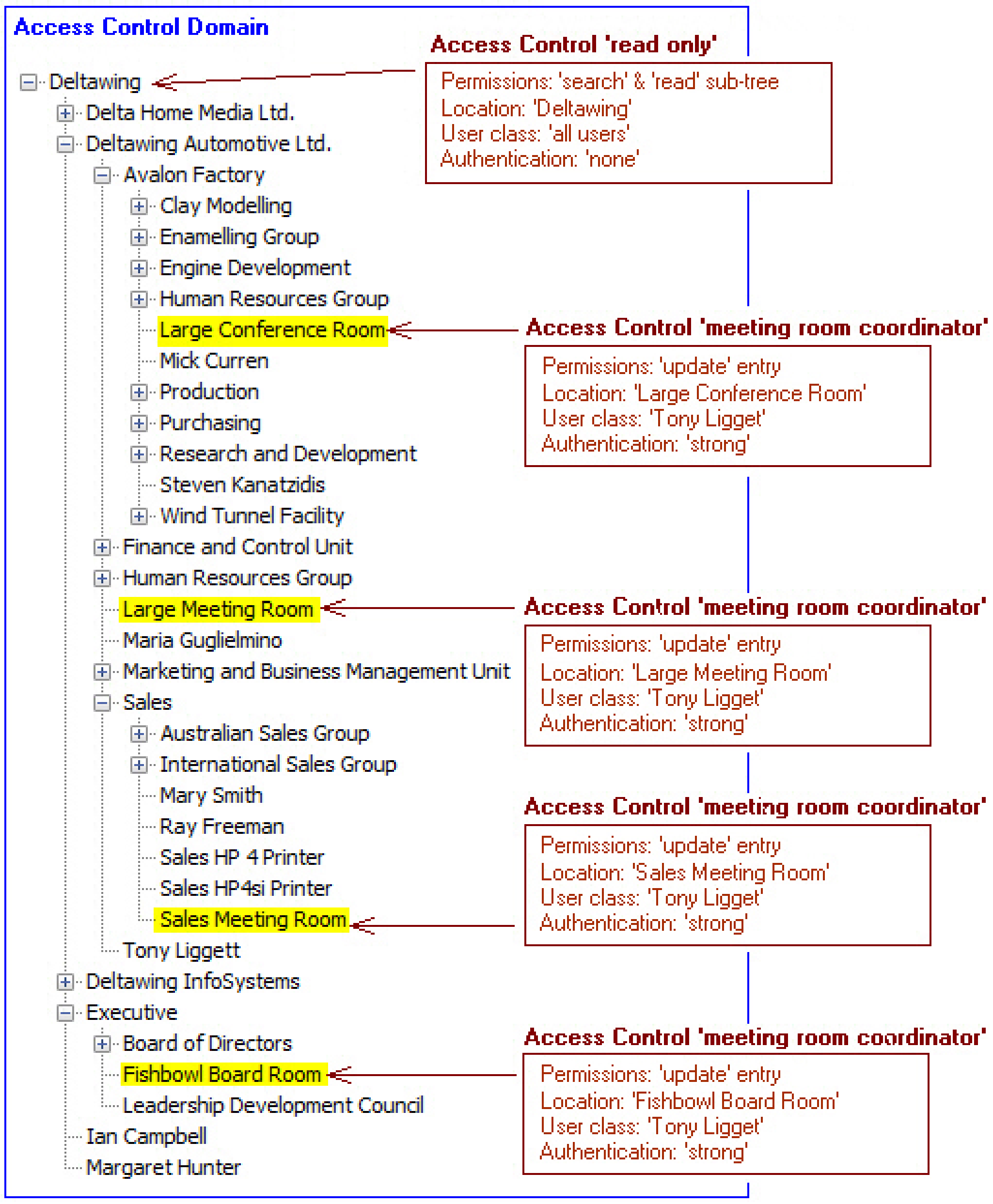
Fulfilling the requirement

This requirement can be fulfilled by adding a new Access Control Item (ACI) to the Access Control Domain created in the first example and by giving it the following properties:

- Name - the name of an ACI should be meaningful and is 'meeting room coordinator' in this example.
- Permissions - the ACI grants 'update' permission on the entries where it is located. (It is an Entry ACI.)
- Locations - the ACI has four locations, the meeting rooms, where the permissions apply. (Note that because this is an Entry ACI, any new locations will be individual entries.)
- User class - there is one person in the user class, Tony Liggett.
- Authentication - the level of authentication required is 'strong'. If a user connects with a lower level of authentication, access to the locations will be denied.

Note that the people in this ACI's user class will also have 'search' and 'read' permissions because the 'read only' ACI also applies to them.

This is illustrated below.



Example 3: Multiple locations with multiple user classes

In this example, the requirement is for each division of Deltawing to have its own administrative user who can update users' entries. The following people should have 'search', 'read' and 'modify' access to all users' entries in their division:

- Craig Hunt (Delta Home Media Ltd)
- Maria Guglielmino (Deltawing Automotive Ltd)
- Celine Joyce (Deltawing InfoSystems)

However, they must connect to the directory with 'strong' authentication (their identity is confirmed through a security certificate).

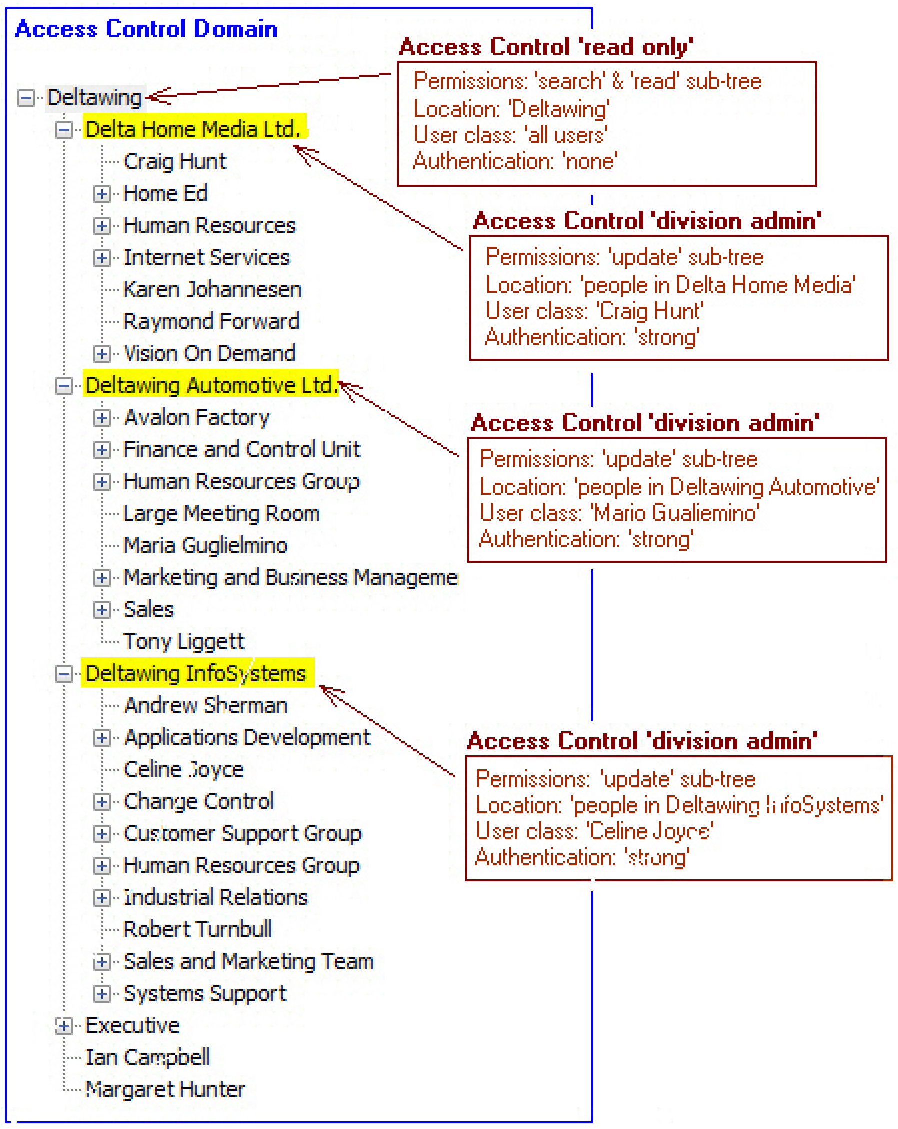
Fulfilling the requirement

This requirement can be fulfilled by adding a new Access Control Item (ACI) to the Access Control Domain created in the first example and by giving it the following properties:

- Name - the name of an ACI should be meaningful and is 'division admin' in this example.
- Protected items - the ACI grants 'update' permissions on the entries where it is located.
- Locations - the locations are the people entries in each division.
- User class - there is a different user class at each location.
- Authentication - the level of authentication is 'strong'.

Note that the people in this ACI's user class will also have 'search' and 'read' permissions because the 'read only' ACI also applies to them.

These properties are illustrated below:



Access Control Domain

Deltawing
- Delta Home Media Ltd.
  - Craig Hunt
  - Home Ed
  - Human Resources
  - Internet Services
  - Karen Johannesen
  - Raymond Forward
  - Vision On Demand
- Deltawing Automotive Ltd.
  - Avalon Factory
  - Finance and Control Unit
  - Human Resources Group
  - Large Meeting Room
  - Maria Guglielmino
  - Marketing and Business Manageme
  - Sales
  - Tony Liggett
- Deltawing InfoSystems
  - Andrew Sherman
  - Applications Development
  - Celine Joyce
  - Change Control
  - Customer Support Group
  - Human Resources Group
  - Industrial Relations
  - Robert Turnbull
  - Sales and Marketing Team
  - Systems Support
- Executive
- Ian Campbell
- Margaret Hunter

Access Control 'read only'
Permissions: 'search' & 'read' sub-tree
Location: 'Deltawing'
User class: 'all users'
Authentication: 'none'

Access Control 'division admin'
Permissions: 'update' sub-tree
Location: 'people in Delta Home Media'
User class: 'Craig Hunt'
Authentication: 'strong'

Access Control 'division admin'
Permissions: 'update' sub-tree
Location: 'people in Deltawing Automotive'
User class: 'Mario Gualiemino'
Authentication: 'strong'

Access Control 'division admin'
Permissions: 'update' sub-tree
Location: 'people in Deltawing InfoSystems'
User class: 'Celine Joyce'
Authentication: 'strong'

Example 4: Multiple Access Control Domains

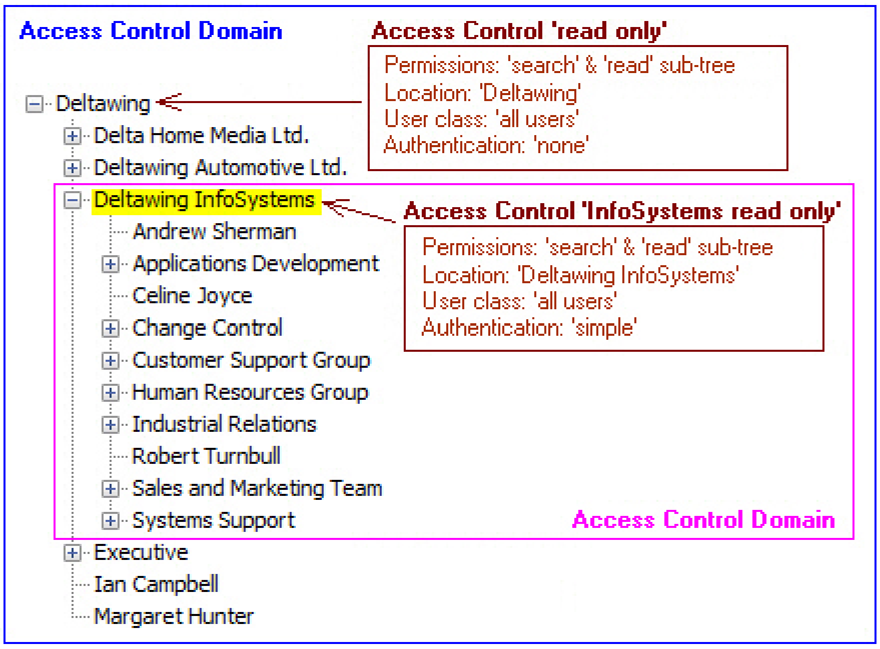
In this example, the requirement is for Deltawing InfoSystems to be completely autonomous. Only employees of Deltawing InfoSystems should be able to search and view this subtree after connecting to the directory using 'simple' authentication.

Fulfilling the requirement

This requirement can be fulfilled by creating a new Access Control Domain at the top of the Deltawing InfoSystems subtree, and then creating a new Access Control Item (ACI) within it with the following properties:

- Name - the name should be meaningful and is 'InfoSystems read only' in this example.
- Permissions - the ACI grants 'search' and 'read' permissions on the entries in the subtrees where it is located.
- Locations - the ACI has one location, Deltawing InfoSystems, where its permissions apply.
- User class - the user class is everyone in the Deltawing InfoSystems subtree.
- Authentication - the level of authentication required is 'simple'.

The following illustration shows the new Access Control Domain and the one created in the first example:



An ACI only applies within its Access Control Domain:

- the 'read only' ACI does not apply to users in the Deltawing InfoSystems subtree
- the 'InfoSystems read only' ACI only applies to users in the Deltawing InfoSystems subtree

Example 5: Default Access Control Items

Whenever you create a new Access Control Domain, you are presented with the option to automatically create the following default Access Control Items (ACIs):

- Own Entry Access
- Read Access
- Super User Access
- Update Access

Own Entry Access

Allows all users to modify their own password, and has the following properties:

- Permissions – grants 'modify' permission on the entry and password attribute where it is applied. (Granting permission to modify an attribute involves granting 'modify' twice - first, on the entry permission, and then on the specific attribute.)
- Location – all entries in the Access Control Domain.
- User class – the user class is 'this entry', which refers to the user who connects to the directory with the same RDN as the entry being accessed.
- Authentication – simple.

Read Access

Allows all users to search and view directory entries, and has the following properties:

- Permissions – grants 'search' and 'read' permissions on the entries where it is located.
- Location – all entries in the Access Control Domain.
- User class – all users in the Access Control Domain.
- Authentication – 'simple'.

Super User Access

Provides full access to all entries in the directory, and has the following properties:

- Permissions – grants all permissions on all entries and attributes (including user names, passwords and some operational attributes).
- Location – all entries in the Access Control Domain.
- User class – none assigned.
- Authentication – 'simple'.

Update Access

Allows a user to update entries in the directory, and has the following properties:

- Permissions – grants all permissions (except ‘search’ and ‘read’) on all entries and attributes (except users’ passwords), and grants access to several operational attributes.
- Location – all entries in the Access Control Domain.

- User class – none assigned.
- Authentication – 'simple'.

Working with X.500 Basic Access Controls

This subsection introduces you managing X.500 Basic Access Controls through the ViewDS Management Agent. It also includes a high-level overview to help you plan Basic Access Controls according to your requirements.

Basic Access Control and the VMA

To add the default Basic Access Controls to Deltawing:

1. If the DIT tab is not displayed, click **Global DIT View** at the bottom of the left pane.
2. Click the **Deltawing** entry at the top of the DIT. The entry’s attributes are displayed in the Attributes tab.
3. Right-click the **Deltawing** entry and then click **Add Access Control Domain**. A confirmation window is displayed.
4. In the confirmation window, select the checkbox and then click **OK**. The ACI tab is added to the right pane.
5. k the **ACI** tab.

   The default ACIs are listed in the Subtree ACI tab: Own Entry Access, Read Access and Superuser Access. A fourth default ACI, Update Access, is listed in the Entry ACI tab.
6. Double-click the first ACI in the list, **Own Entry Access**.

   The Own Entry Access window is displayed and includes the following information about the ACI:

   - The name, precedence, authentication level and scope. The scope is Subtree, which tells you that when the ACI is located at an entry in the DIT, its permissions apply to all the entry's subordinates.
   - In the Entry Permissions area, the modify permission is granted. This tells you that the users in the ACI's user class can modify the entries where the ACI is located.
   - In the User Attribute Permissions area, modify access is granted for the userPassword attribute. This tells you that the users in the ACI's user class can modify this attribute for the entries where the ACI is located.
   - In the Operational Attribute Permissions area, modify access is granted for the userConfig operational attribute. This tells you that the users in the ACI's user class can modify this attribute for the entries where the ACI is located.
   - The box on the left of the window, Locations of ACI, shows the locations of the ACI. The entries where the ACI is located are displayed in pink text. You can see that this ACI has one location, Deltawing. As its scope is 'subtree', this means that the ACI's permissions apply to the Deltawing entry and all its subordinate entries.

7. To view the user class for the ACI, right-click the **Deltawing** entry and then click **Edit User Classes**.

   The User Classes window is displayed. The 'This entry' box is selected in the window, which tells you that the user class comprises every user who connects to the directory with the same RDN as the entry being accessed. The result is that each user can modify their own entry.

8. Click **Cancel** to close the User Classes window, and then click **Cancel** again to close the Own Entry Access window.

Identifying requirements

Consider the following for a new installation:

1. Do you need more than one Access Control Domain? (Usually, just one Access Control Domain is required. An exception, however, is when access needs to be managed autonomously in different areas of a DIT.)
2. What are the different roles and groups of users that need access to the directory:
   - What are the groups of users that need different levels of access?
   - What is the basic level of access you want to allow all users?
   - How will you identify these user groups (as individual entries, a group of entries, a role identified by a refinement)?
   - What special access do you want to grant to a user to their own entry (for example, ’self service’)?
   - Are any of your requirements fulfilled by the default access controls)?

For each role or user group you have identified:

1. Which locations in the directory should they have access to?
2. Can these locations be identified as entire subtrees, refinements of subtrees, or individual entries? (The last option, individual entries, should only be used when the locations cannot be identified as subtrees or refinements. Another alternative is to use an auxiliary object class to identify the entries to which the ACI should apply.)
3. Which operational and user attributes at each location do you want to protect?
4. Should the user class be the same at each location of the ACI?
5. What are the permissions in the existing ACIs that apply to these users?
6. What permissions should apply to the protected entries and attributes? (All access is denied by default, and it is usually unnecessary to use an ACI to deny access. Access should be granted on an ’as needed’ basis.)
7. What level of authentication is required by the users?

For the steps to create an Access Control Domain and to create ACIs, see the ViewDS Management Agent help.

Key concepts: XACML

This section introduces the concepts required to work with XACML access controls, and includes how to manage XACML security through the ViewDS Management Agent.

It covers the following topics:

- Brief introduction to XACML
- XACML framework and Access Sentinel
- Introduction to XACML policy
- Attribute versus role-based access control
- Tutorial: Attribute-based access control
- Tutorial: Role-based access control

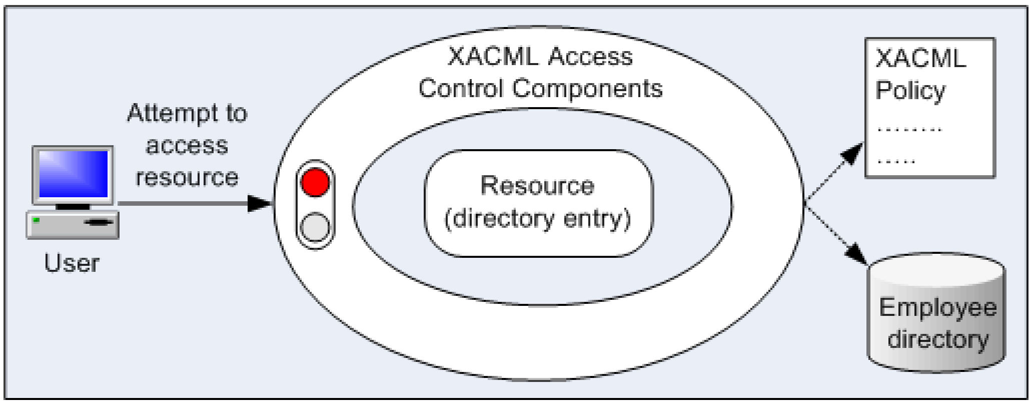
Brief introduction to XACML

XACML Version 3.0 is a standard that provides a framework for fine-grained, enterprise-wide access control. The standard describes two languages, both written in XML: an access-control *policy* language, and an access-control *decision* language.

The policy language describes access-control requirements by defining policies that describe, for example, who can access what and when. The decision language is used to form requests and responses. A request asks whether a given action by a given entity should be allowed; and a response provides the answer, which is determined according to an XACML policy.

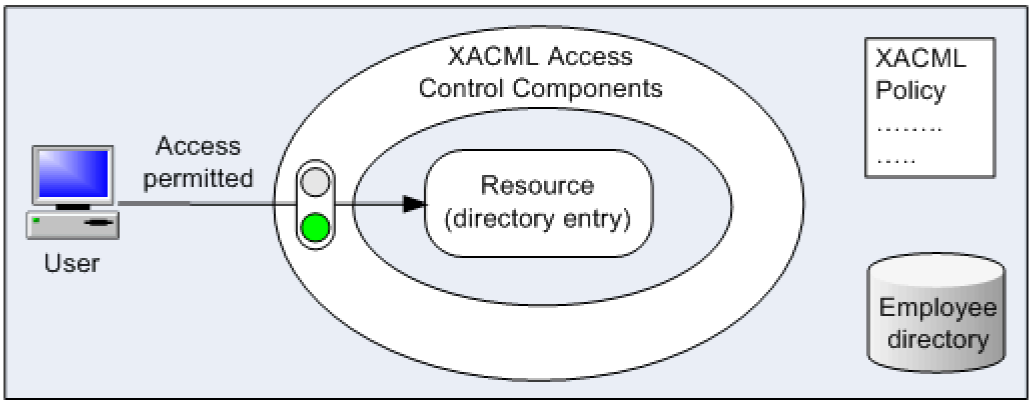
Simplified XACML implementation

The following illustrates a simplified XACML implementation.


User
Attempt to access resource
XACML Access Control Components
Resource (directory entry)
XACML Policy
....
.....
Employee directory

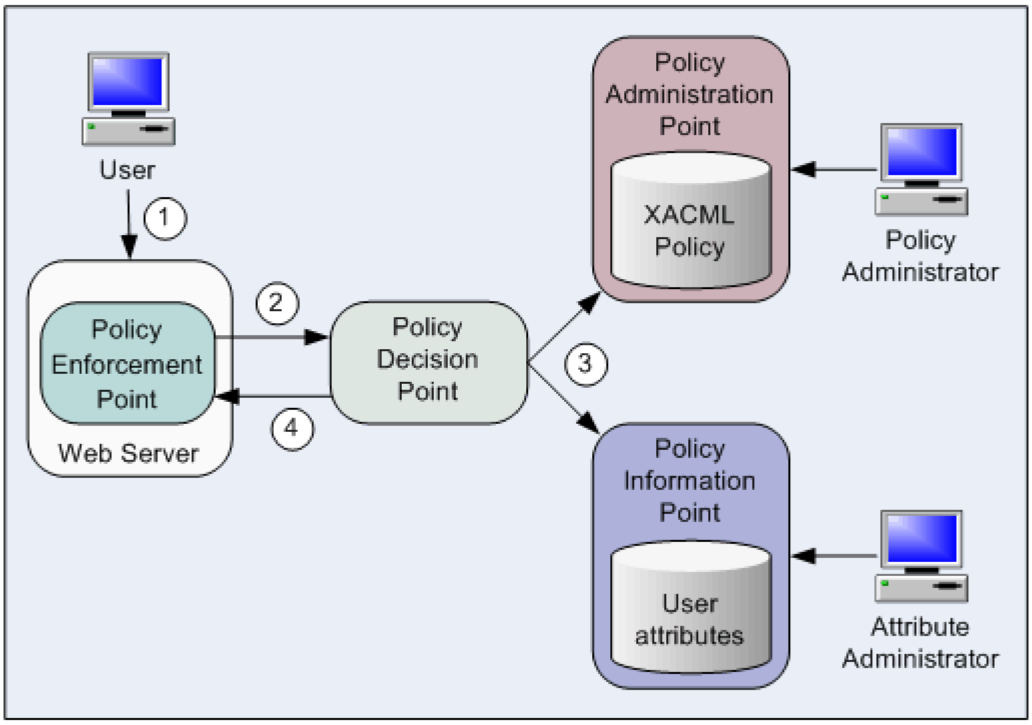
In the following illustration, a user attempts to view a document file protected by an XACML access-control implementation. The implementation determines whether the user should be permitted or denied access by interrogating the appropriate XACML policy.

The policy might include considerations such as the user's security level, department, role, position, location and the time of day. All combine to determine whether the user should be allowed access to the resource (as shown below).



XACML access control components

An implementation of XACML access control has four logical components (shown below).



The logical components are:

- Policy Enforcement Point (PEP)
  Protects a resource from unauthorized actions.
- Policy Decision Point (PDP)
  Determines whether access should be granted to a protected resource.

- Policy Administration Point (PAP)
  Allows policies to be created and stored in a repository.
- Policy Information Point (PIP)
  Stores additional information, such as user attributes, that can be used by the PDP to make access-control decisions.

In the illustrated example the resources protected by a Policy Enforcement Point (PEP) are the web pages available through a web server.

The steps shown in the illustration are as follows:

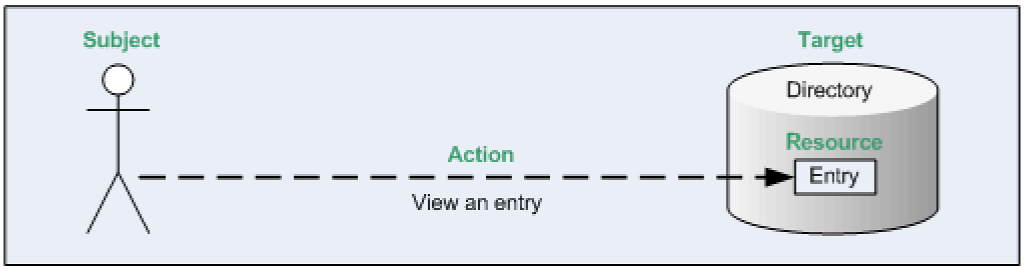
1. A user requests access to a web page.
2. The web server asks the Policy Enforcement Point (PEP) to send an 'authorization decision request' to the Policy Decision Point (PDP). The request includes *XACML attributes* that identify (among other things) the user, the resource they are attempting access, the action they are attempting to perform, and the environment (for example, date and time).
3. The Policy Decision Point (PDP) determines whether access should be permitted. It looks at the appropriate XACML policy in the Policy Administration Point (PAP), and the appropriate user attributes in the Policy Information Point (PIP). The information in the PIP allows the PDP to identify the user attempting to access the resource.
4. The PDP returns an 'authorization decision response' to the Policy Enforcement Point (PEP), which then acts on the decision to permit or deny access to the user.

XACML terms to remember

There are a couple of important XACML terms to remember:

- **Target** – the set of resources protected by the XACML policy (for example, a directory or a web site)
- **Resource** – the specific item (for example, an entry, attribute or value in the directory or a specific web page) within the target that the subject is attempting to access
- **Subject** – the user attempting to access a resource
- **Action** – the action attempted by the subject (e.g. view or modify an entry or web page)

These terms are illustrated below for XBAC where the target is the ViewDS directory and the resource is an individual directory entry.



XACML framework and Access Sentinel

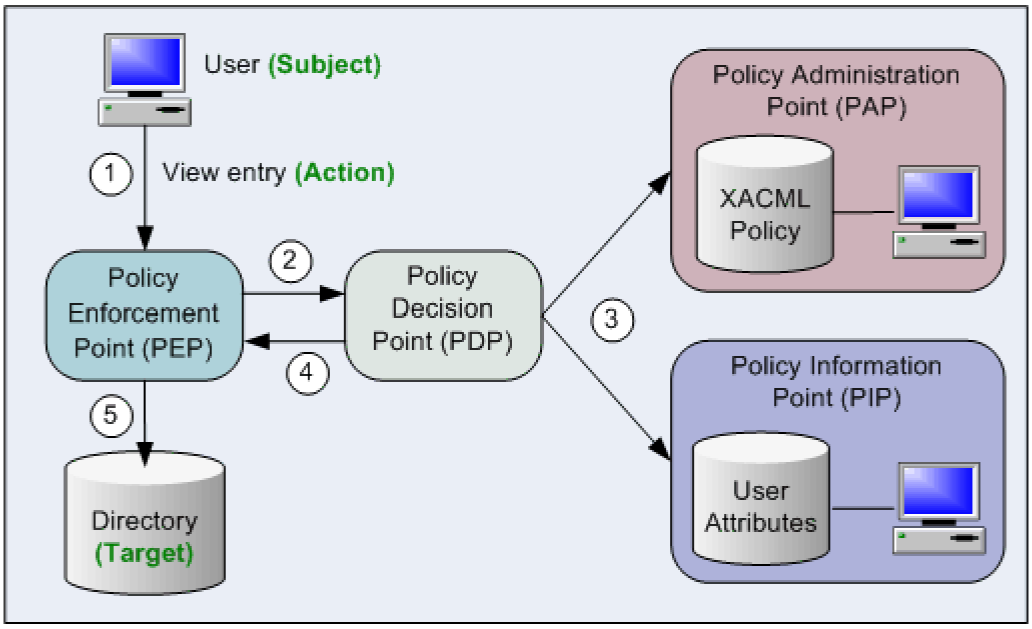
The **XACML framework** is part of ViewDS Directory. It allows you to apply the **XACML Access Control scheme** by defining XACML policy that controls access to the directory.

ViewDS **Access Sentinel** is an extension of the XACML framework that allows you to apply XACML policy to applications external to ViewDS. Access Sentinel requires additional licencing beyond that of the core ViewDS Directory.

The XACML framework and Access Sentinel conform to the XACML Version 3.0 standard.

ViewDS XACML framework

The following illustration shows the logical components of the ViewDS XACML framework. The framework is part of the core ViewDS Directory product.


User (Subject)
1
View entry (Action)
Policy Administration Point (PAP)
XACML Policy
2
Policy Enforcement Point (PEP)
Policy Decision Point (PDP)
3
4
Policy Information Point (PIP)
5
User Attributes
Directory (Target)

Aside from the user interfaces, all components of the framework are within the ViewDS Directory System Agent (DSA) and database.

The three data sets in the illustration are shown as separate components for the sake of clarity. The PAP's XACML policies, the PIP's user attributes, and the directory are all stored in the ViewDS database and can be managed through the ViewDS Management Agent.

The steps shown in the illustration are as follows:

1. A user attempts to view an entry in the ViewDS directory.
2. The PEP sends an 'authorization decision request' to the PDP. The authorization decision request includes a set of XACML attributes that identify (among other things) the user, the entry they are attempting to access, and the action they are attempting to perform .
3. The PDP determines whether access should be permitted. It looks at the appropriate XACML policy in the PAP, and the appropriate user attributes in the PIP (the user may be identified according to directory attributes in the PIP).

4. The PDP returns an 'authorization decision response' to the PEP, which enables ViewDS to act on the decision to permit or deny access to the document.
5. If access has been permitted, the user is allowed to view the entry.

Introduction to XACML policy

The ViewDS implementation of an XACML policy comprises:

- XACML Access Control Domain
- Status and version
- XACML attributes
- Rules

Each is described below.

XACML Access Control Domain

An XACML Access Control Domain is a specific area of a DIT that contains one or more XACML policies.

NOTE: In the ViewDS XACML framework, the default behaviour is to deny access to the entities within an Access Control Domain. (This does not apply to administrative users of the ViewDS Management Agent, who bypass all access controls.)

For example, when working with the ViewDS directory and the internal PEP, an XACML Access Control Domain is an area of the directory where the XACML access controls apply. The entry at the top of the domain is termed the access control administrative point. By default, ViewDS denies access to all entries within the domain.

Status and version

Every XACML policy has a status and version.

A policy can have multiple versions, each with a unique version number. A version also has a status that identifies whether it is 'locked' and 'active'.

Only one version of a policy can be 'active'. This is the version that currently applies. You can therefore test a new version of a policy and then roll-back to a previous version if necessary.

A 'locked' version cannot be modified. However, you can create a new version based on an existing locked version. This offers a level of version control.

Status: `active,locked`
Version: `1.1`

XACML attributes

XACML is based on the concept of attributes.

The PAP uses XACML attributes to identify the subject, resource, action and environment information within a rule. The PEP sends requests made up of XACML attributes to the PDP to convey information about the subject, resource, action and environment. The PDP then compares these to attribute values in a policy to make access decisions.

The XACML standard defines four categories for attributes:

- Subject – identifies the subject attempting to access a particular resource.
- Resource – identifies the resource the subject is attempting to access.
- Action – identifies the action the subject is attempting to perform on the resource (for example, read, modify).
- Environment – identifies environmental factors such as day of the week and time of day.

It is permissible within the XACML standard for any of these four categories to be sub-divided or for other new attribute categories to be added.

For details of the XACML categories and data types of the attributes provided by the PEP, see the VMA in-application help topic *XACML attributes provided by the ViewDS PEP*.

For an XACML attribute to be included in policy rules, it must first be declared in the XACML Access Control Domain. Declaring an XACML attribute involves giving it a 'user-friendly' name. This is important because XACML attributes are identified by long URIs or complex XPath expressions that are unwieldy when creating rules.

You can declare two different types of attributes: attribute designators and selectors.

Attribute Designators

An attribute designator comprises the `Category`, `AttributeId` and `DataType` URIs of a particular XACML attribute.

For some XACML attributes, the declaration also includes a mapping to a directory attribute in an entry that uniquely identifies a subject or resource.

Attribute designators allow a policy to specify an attribute value with a given category, identifier and data type. The PDP will then look for that value in the request, or elsewhere, if no matching values can be found in the request .

Attribute Selectors

In addition to attributes, XACML requests can contain XML documents for each category. For example, an XML document might describe the subject or be the actual resource being accessed.

Attribute selectors allow a policy to look for attribute values in an XML document using XPath queries. XPath is a language, based on a tree representation of XML documents, which provides the ability to navigate the tree and select nodes using a variety of criteria.

An attribute selector comprises a category, data type and an XPath expression. Together these are used to resolve a set of attribute values in the request document.

Attribute selectors can be used within XACML policy expressions in the same way as attribute descriptors. For example, consider an XACML request that contains an XML document which is the resource a user is attempting to access. An attribute selector can include an XPath expression to find elements in the document named PublicationDate. An XACML policy can then include a condition that denies access if the PublicationDate is more than five years ago.

The following are currently supported:

- the definition of attribute selectors within the Authorization Policy Manager (and the ViewDS Management Agent)
- the ability to use and evaluate attribute selectors within XACML policies

NOTE: Attribute selectors are not applicable to the ViewDS XACML framework or HTTP PEPS. This is because they do not make use of XML documents within authorization decision requests.

Rules

Every XACML policy includes a rule.

A rule allows the Policy Decision Point (PDP) to determine whether a subject should be permitted or denied access to a resource. Each has a target, scope, an effect (permit or deny access) and a condition.

The *target* identifies the resources protected by the policy. The scope is used when defining policy for hierarchical resources, such as directory entries. It determines whether the policy applies to a single target resource, or to a target resource and all its subordinates.

The condition incorporates XACML attributes which the PDP uses to identify the resource and subject. It determines whether the rule's effect should be applied.

A simple example rule is shown below.

Rule
Target: Documents
Scope: subtree
Effect: Permit access (if the following condition is true)
Condition:
resource has attribute `webpage` = '`index.html`' AND
subject has attribute `role` = `Board Member` AND
action = READ

The *condition* is true if the subject is a Board Member attempting to view the resource 'index.html'.

Also to consider is that there are two kinds of rules: attribute- and role-based access control (ABAC and RBAC).

Attribute- versus role-based access control

The ViewDS XACML framework supports both attribute-based access control (ABAC) and role-based access control (RBAC) rules.

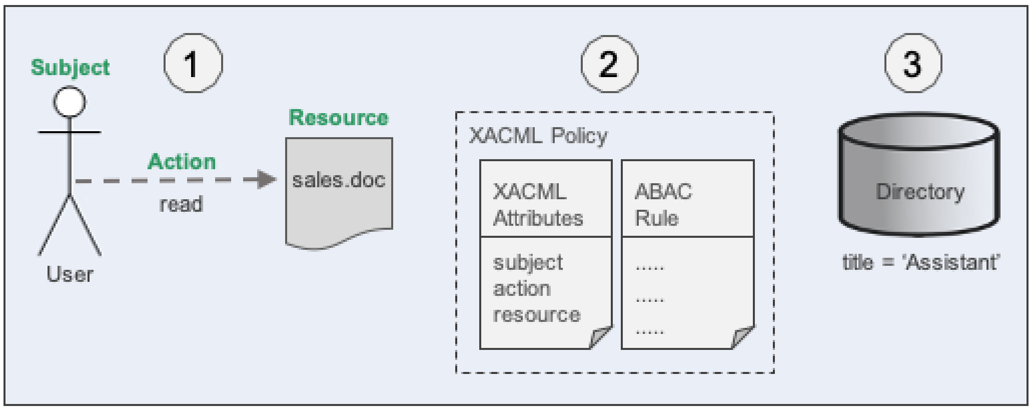
Attribute-based access control

With ABAC, the attributes associated with the subject, action, resource or environment are used to construct conditions. These conditions compare attributes to static values, or to one another (relation-based access control), to determine whether access should be permitted or denied.

To illustrate, here is an example ABAC rule:

```
Permit access if:
   action = read AND
   resource = sales.doc AND
   subject’s title = ‘Assistant’
```

The following illustration includes the example rule.



The steps in the illustration are as follows:

1. A user attempts to read the `sales.doc` file.
2. ViewDS evaluates the XACML policy. The XACML Attributes definition tells ViewDS to obtain the 'subject's title' from the value of the `title` attribute in the user's directory entry.
3. As the user's `title` attribute is set to 'Assistant', all the clauses in the ABAC rule are true and ViewDS permits access to the document.

Role-based access control

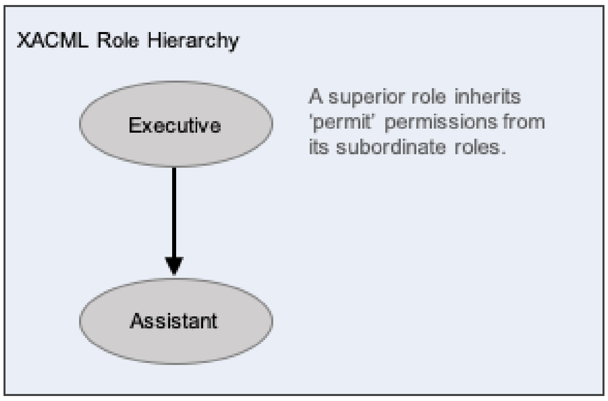
RBAC also uses attributes to construct conditions. However, unlike an ABAC rule, an RBAC rule includes a role condition that is evaluated against a role hierarchy.

For example, here is an example RBAC rule:

```
Permit access if:
 action = read AND
 resource = sales.doc AND
 subject’s role = Assistant
```

Aside from the 'role' clause, this is identical to the ABAC example. However, another important distinguishing feature of RBAC is that it includes a role hierarchy.

A role hierarchy defines permission inheritance. It allows ViewDS to evaluate a user's access rights according to their role's position in the hierarchy. However, while each role inherits access rights from its subordinates, this only applies to the 'permit' permissions. In the following example, the role 'Executive' inherits permissions from the role 'Assistant'.


XACML Role Hierarchy
Executive
Assistant
A superior role inherits 'permit' permissions from its subordinate roles.

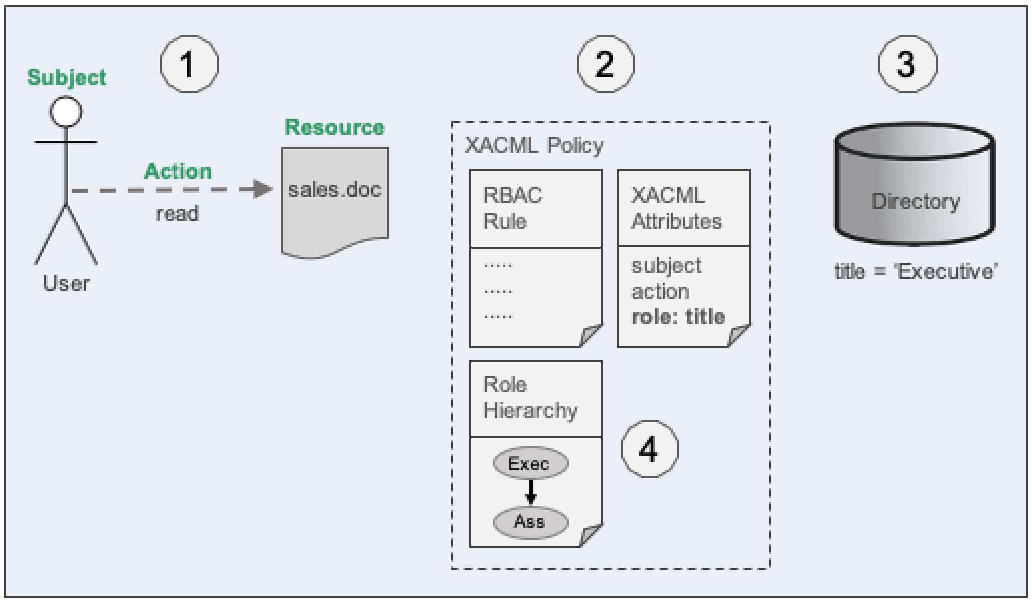
When ViewDS evaluates the previous rule that permits access to an 'Assistant', it also evaluates the role hierarchy which defines that an 'Executive' also has this permission. However, before ViewDS can evaluate the role hierarchy, it first needs to map the user (the subject) to a role. This role mapping can be either:

- **Static** - static roles are obtained from a directory entry or the XACML request.
- **Dynamic** - dynamic roles are allocated at run-time by evaluating **role enablement rules**.

Both options are described below.

Static roles

With static roles, the policy's role definition identifies an attribute in the ViewDS directory.



The above illustration shows an XACML policy that includes a static role, which is mapped to the `title` attribute in a user's directory entry.

The steps in the illustration are as follows:

1. A user attempts to read the `sales.doc` file.
2. ViewDS evaluates the XACML policy. The definition of `role` tells ViewDS that the subject's role is the value of their `title` attribute in the directory.
3. From the value of the user's 'title' attribute, ViewDS determines that user's role is 'Executive'.
4. From the policy's role hierarchy, ViewDS identifies that a 'Executive' inherits permissions from an 'Assistant'. ViewDS therefore grants the user access to the document.

With static roles, the names of the roles in the role hierarchy must exactly match values in the corresponding directory attribute.

As you can probably see, the only real difference between RBAC with static roles and ABAC is the permission inheritance defined by a role hierarchy.

Role enablement

Role enablement provides greater flexibility to RBAC by allowing ViewDS to assign roles dynamically. It does this by evaluating a policy's *role enablement rules*.

To illustrate, consider these example role enablement rules:

Assign the role of **Executive** if:
subject's `title` attribute contains 'Director' OR 'Chief Executive Officer' OR 'Board Member'

Assign role of **Assistant** if:
subject's `title` attribute contains 'Executive Assistant' OR 'Personal Assistant' OR 'Administrative Assistant'

When ViewDS assesses the first rule, it will assign the role of 'Executive' to the subject (user) if their `title` attribute contains either 'Director', 'Chief Executive Officer' or 'Board Member'.

Finally, another example that illustrates the flexibility provided by role enablement:

Assign the role of **Contractor** if:
subject's `emailAddress` attribute contains '@third-party-contractor'

With this example, ViewDS will assign the role of 'Contractor' to the user if the `emailAddress` stored in their directory entry contains the specified string.

Tutorial: Attribute-based access control

This tutorial takes you through how to define an XACML policy that includes attribute-based access control (ABAC). The policy will apply to an area of the Deltawing directory that is provided with the ViewDS Directory.

The tutorial includes the following stages:

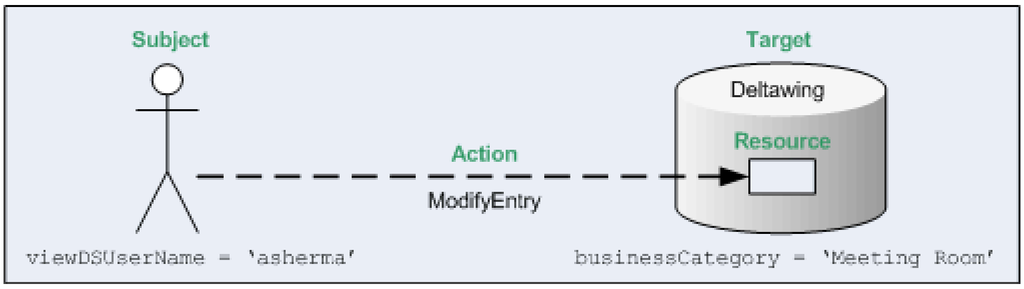
1. Requirements
2. Create an XACML Access Control Domain
3. Create an XACML policy
4. Declare XACML attributes
5. Define the first rule
6. Define the first rule's condition
7. Define the second rule
8. Define the second rule's condition
9. Activate the policy
10. Test the policy
11. Lock the policy

Requirements

This tutorial's requirement is for a policy that gives one user, Andrew Sherman, the privileges to modify meeting room entries in the Deltawing directory.

Both Andrew Sherman and a meeting room can be identified in the Deltawing directory by their entries' directory attributes:

- Andrew can be identified by his entry's `viewDSUserName` attribute which is set to 'asherma'.
- a meeting room entry can be identified by its `businessCategory` attribute which is set to 'Meeting Room'.



When a directory user (subject) attempts to modify an entry (resource), the Policy Enforcement Point (PEP) will send an authorization decision request to the Policy Decision Point. The request includes the values of directory attributes in the subject and resource entries, plus a value to identify the attempted action. These values are held in XACML attributes.

XACML attributes

Before an XACML attribute can be used by the PAP, it must first be declared in the XACML Access Control Domain.

Each declaration has a 'Label' that will appear in a rule's condition, an XACML category, and may also require a mapping to a directory attribute.

In this tutorial, the following declarations are required.

Label	XACML attribute category	XACML attribute identifier	XACML data type
User Name	urn:oasis:names:tc:xacml:1.0:subject-category:access-subject	urn:oasis:names:tc:xacml:1.0:subject:subject-id	string
Action	urn:oasis:names:tc:xacml:3.0:attribute-category:action	urn:oasis:names:tc:xacml:1.0:action:action-id	string
Business Category	urn:oasis:names:tc:xacml:3.0:attribute-category:resource	businessCategory (urn:oid:2.5.4.15)	string

Note that an XACML attribute's category corresponds to its purpose, as shown in the illustration above.

Also note that `Business Category` is mapped to the directory attribute `businessCategory` through its XACML Attribute Identifier. However, `User Name` does not need to be mapped to a directory attribute because it is one of three values the PEP provides to identify the subject.

Rules

Two rules are required. The first will permit Andrew Sherman to modify meeting room entries in the directory. The second will permit all users to search and view entries in the directory. This is necessary because the default behaviour is to deny access within an Access Control Domain, unless explicitly permitted.

Rule 1

The first rule's target, scope, effect and condition are shown below.

Rule 1:
Target: `Deltawing`
Scope: `subtree`
Effect: `Permit` (if the condition is true)
Condition:
resource has attribute `Business Category` = 'Meeting Room' AND
subject has attribute `User Name` = 'asherma' AND
(Action = 'ModifyEntry' OR Action = 'AddType' OR
Action = 'RemoveType' OR Action = 'AddValue' OR Action = 'RemoveValue')

The rule's target will be the entry at the root of the Deltawing directory; its scope will be the entire subtree below the root entry; and its effect will be to permit access if the condition is true.

The condition will be true when the user with the `User Name` 'asherma' (subject) attempts one of the actions on a meeting room entry (resource).

Note that omitting the resource clause would make the rule more general so that it applied it to all entries in the directory.

Rule 2

The second rule’s target, scope, effect and condition are shown below.

Rule 2:
Target: `Deltawing`
Scope: `subtree`
Effect: `Permit` (if the condition is true)
Condition:
Action = 'ReadEntry’ OR Action = 'BrowseEntry' OR
Action = 'ReturnDN' OR Action = 'ReadType' OR
Action = 'FilterMatchType' OR Action = 'ReadValue' OR Action = 'FilterMatchValue

It has the same target and scope as the first rule. It also permits access if the condition is true.

The condition will be true when any user (subject) attempts one of the search or read actions on any directory entry (resource).

Create an XACML Access Control Domain

An XACML Access Control Domain is a specific area of a DIT that contains one or more XACML policies. The entry at the top of the domain is termed the *access control administrative point*.

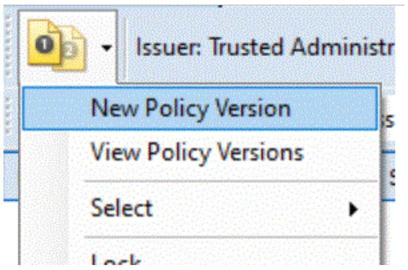
To create an XACML Access Control Domain below the `Deltawing` entry in the Deltawing directory:

1. At the bottom of the left pane, click **Server View**.
2. In the left pane, click your ViewDS server. The Status tab displays the status of your ViewDS server.
3. Ensure that the ViewDS Management Agent is connected to your ViewDS server, and that your server is running.
4. At the bottom of the left pane, click **Global DIT View**.
5. In the left pane, expand the **Deltawing** entry.
6. Right-click the **Deltawing** entry. A submenu is displayed.
7. From the submenu, click **Add XACML Access Control Domain**. The XACML AC tab is added to the right pane.

Create an XACML policy

To create the policy:

1. In the right pane, click the **XACML AC** tab and then the **Policy Versions** tab.
2. In the right pane, click the **Version Management** button followed by **New Policy Version**. The XACML Policy Version window is displayed.



3. Accept the default values by clicking **Save**. The new policy's version and status are displayed next to the Version Management button.

NOTE: The policy is marked as open, which indicates that it can be modified. Once a policy has been locked it cannot be modified. You can, however, create a new policy based on it.

Declare XACML attributes

To declare the XACML attributes for the tutorial's policy:

1. In the right pane, click the **Attributes** tab.
2. At the bottom of the right pane, click **New**. The XACML Attribute window is displayed.
3. In the **Label** box, enter '`Action`'. This is the name that will appear in the rule.
4. In the **Category** box, click **urn:oasis:names:tc:xacml:3.0: attribute-category:action**. The content of the Identifier box defaults to `urn:oasis:names:tc:xacml:1.0:action:action-id`, and the Data Type box defaults to string.
5. In the **Permitted Values** area, click the **Add** button. The XACML Value (String) window is displayed.
6. In the text box, enter '`ReadEntry`' and click **OK**. The value is added to the Permitted Values box.
7. Repeat steps 5 and 6 to define the following as permitted values: ModifyEntry, BrowseEntry, RemoveType, AddType, AddValue, RemoveValue, ReturnDN, ReadType, FilterMatchType, ReadValue, FilterMatchValue, DiscloseValueOnError, DiscloseTypeOnError, DiscloseEntryOnError.
8. Click **Save**. The XACML attribute is added to the Attributes tab.

9. Repeat steps 2 to 4 to declare the XACML attributes in the following table.
   Note that the XACML attribute `Business Category` is mapped to the directory attribute `businessCategory`.

Label	Category	Identifier	Data Type
User Name	urn:oasis:names:tc:xacml:1.0:subject-category:access-subject	urn:oasis:names:tc:xacml:1.0:subject:subject-id	string
Business Category	urn:oasis:names:tc:xacml:3.0:attribute-category:resource	businessCategory(urn:oid:2.5.4.15)	string

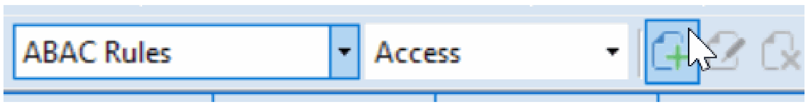
For information about the XACML attributes provided by the ViewDS PEP, see the *ViewDS Access Sentinel: Installation and Reference Guide*.

NOTE: Every attribute in an XACML domain must have a unique combination of Category, Identifier and Data Type.

Define the first rule

To create the first rule in the policy:

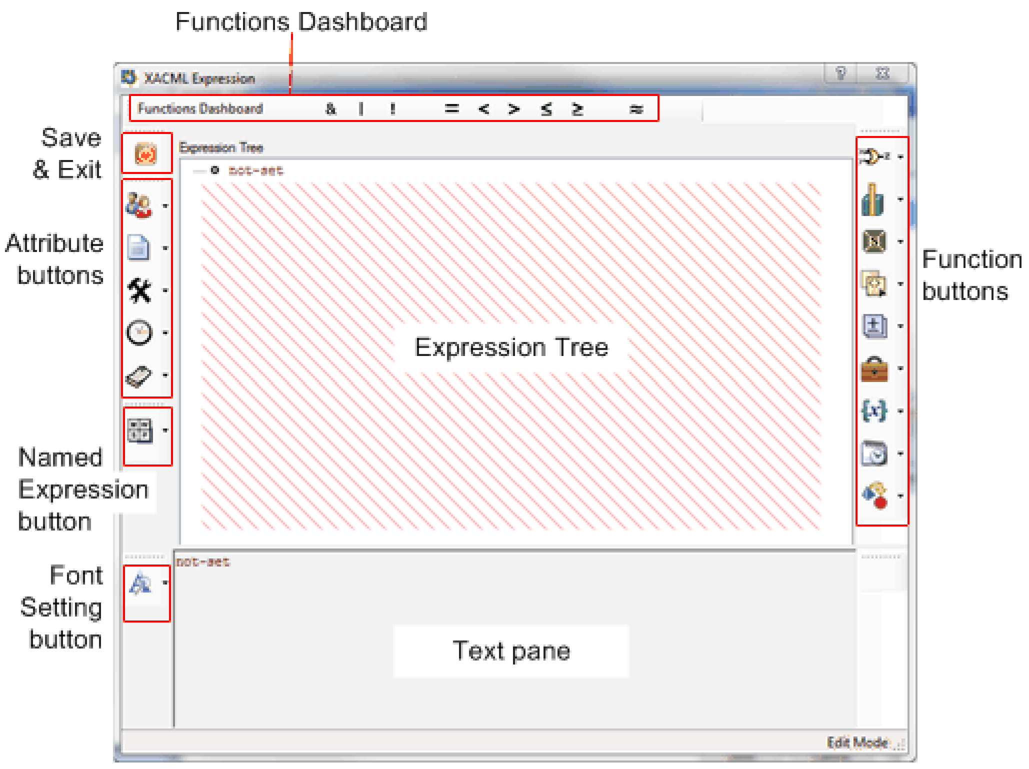
1. In the right pane, click the **Policy Versions** tab.
2. With 'ABAC Rules' and 'Access' selected in the filter boxes, click the **New** button. The XACML Rule window is displayed. It allows you to define a rule for the current policy.



3. In the **Label** box, enter `Andrew Sherman meeting room access`.
4. Enter a short **Description** of the rule, such as `ABAC that permits Andrew Sherman full access to meeting room entries`. Note that the Target is set to Deltawing and its Scope is subtree. Hence, the target is all subtrees and entries subordinate to Deltawing. Also note that the Effect is set to the default, permit.
5. Click the **Edit** button. The XACML Expression window is displayed (described below).

XACML Expression window

This window allows you to define the expressions that constitute a rule’s condition.



The window has two areas:

- Expression Tree
  This is the window's main work area and allows you to build expressions in a tree format.
- Text pane
  This area shows the contents of the Expression Tree in a plain text format.

There are five sets of buttons:

- Functions Dashboard
  These buttons allow you to add one of the frequently used functions to the Expression Tree. The functions are also available through the function buttons.
- Save and Exit button
  This button allows you save the Expression Tree and exit the Expression Builder window.
- Attribute buttons
  These buttons allow you to add XACML attributes to the Expression Tree. Only the XACML attributes declared in the current Access Control Domain are available. There is a button for each category of XACML attribute: subject, resource, action, environment and other attributes.
- Font Setting button
  This button allows you to change the font for the attributes, values, functions and named expressions displayed in the text pane.
- Named Expression button
  This button allows you to add a named expressions to the Expression Tree. Only the named expressions defined in the current Access Control Domain are available.

- Function buttons
  These buttons allow you to add a function to the Expression Tree. There are nine function categories: Boolean, Relational, XPath, String, Arithmetic, Bag, Set, Date and Time, and Conversion.

Define the first rule's condition

Each rule has a condition comprising a set of expressions. The condition for the rule in this tutorial is as follows:

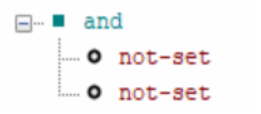
> resource has attribute `Business Category` = 'Meeting Room' AND
> subject has attribute `User Name` = 'asherma' AND
> (`Action` = 'ModifyEntry' OR `Action` = 'AddType' OR
> `Action` = 'RemoveType' OR `Action` = 'AddValue' OR `Action` = 'RemoveValue')

Each line in the condition is an expression. The three expressions are combined by Boolean 'And' functions.

Start with a Boolean 'And'

To apply a Boolean 'And' function:

- In the XACML Expression window, drag and drop the **&** from the Functions Dashboard to the node at the top of the expression tree. The function is displayed in the expression tree with two empty nodes below it.

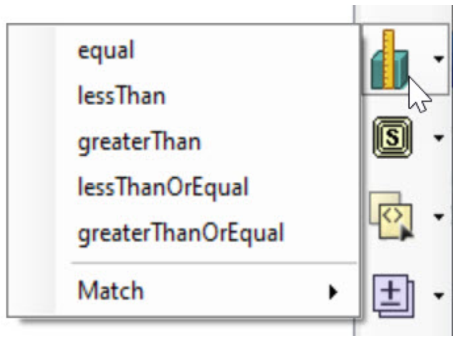


Note: To replace a function, drag and drop another function on top of it.

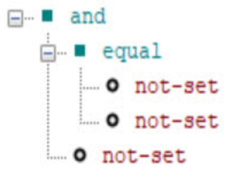
Defining the first expression

You can now define the first expression in the condition:

1. Click the **Relational Functions** button. A list of functions is displayed.

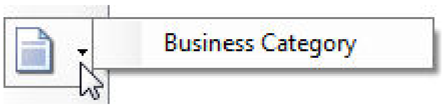


2. Drag and drop **equal** onto the first empty node in the expression tree. The equal function is added to the tree with two new empty nodes below it.

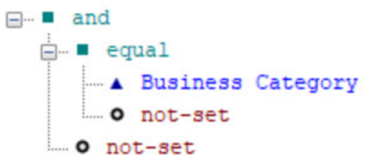


```
and
  equal
    not-set
    not-set
  not-set
```

3. On the left of the window, click the **Resource Attributes** button.

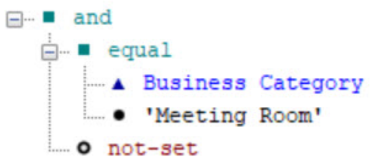


4. Drag and drop **Business Category** onto the fist **not-set** node below the equal function.



```
and
  equal
    Business Category
    not-set
  not-set
```

5. Double-click the **not-set** node below Business Category. The XACML Value (String) window is displayed.
6. In the **Value** box, enter `Meeting Room` and then click **OK**. The string is added to the expression.



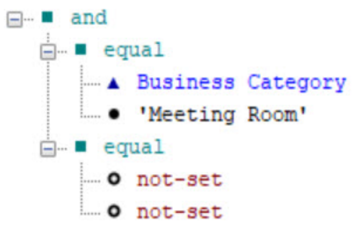
```
and
  equal
    Business Category
    'Meeting Room'
  not-set
```

Now define the second expression in the condition.

Defining the second expression

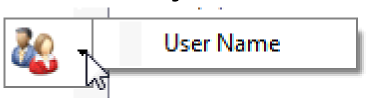
To define the second expression in the condition:

1. From the Functions Dashboard, drag and drop the **=** function onto the remaining **not-set** node. The equal function is added to the tree with two new empty nodes below it.



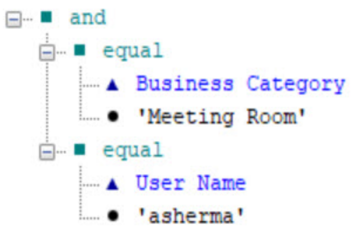
```
and
  equal
    Business Category
    'Meeting Room'
  equal
    not-set
    not-set
```

2. Click the **Subject Attributes** button.



3. Drag and drop **User Name** onto the fist node below the equal function.

4. Double-click the **not-set** node below User Name. The XACML Value (String) window is displayed.
5. Enter `asherma` and then click **OK**. The string is added to the expression.


```
and
  equal
    Business Category
    'Meeting Room'
  equal
    User Name
    'asherma'
```

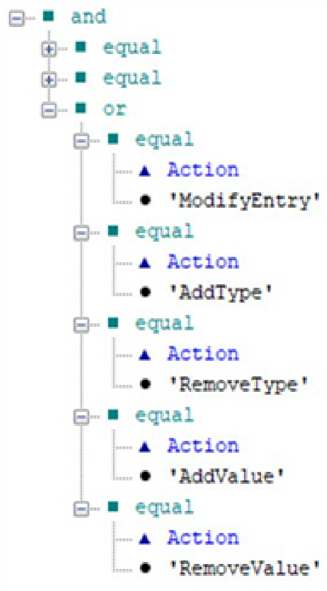
Now define the remaining expression.

Defining the third expression

To define the remaining expression:

1. Right-click the **and** function at the top of the Expression Tree, then click **Add New Argument**. A new 'not-set' node is added to the bottom of the Expression Tree.
2. From the Functions Dashboard, drag and drop the **|** function onto the new **not-set** node. The 'or' function is displayed with two new 'not-set' nodes below it.
3. In the Expression Tree, right-click the **or** function, and then click **Add New Argument**. A third 'not-set' node is displayed below the 'or' function.
4. Repeat step 3 until there is a total of five 'not-set' nodes below the **or** function in the Expression Tree.
5. From the Functions Dashboard, drag and drop the **=** function onto the first **not-set** node below the 'or' function. The equal function is added to the tree with two new 'not-set' nodes below it.
6. Click the **Actions** button. The XACML attribute 'Action' is displayed.
7. Drag and drop **Action** onto the fist **not-set** node below the equal function.
8. Double-click the **not-set** node below 'Action'. The XACML Value (Enumerated) window is displayed.
9. Choose **ModifyEntry** from the dropdown list of values and click **OK**.

10. Repeat steps 5 through 9 to add the following actions to the Expression Tree: AddType, RemoveType, AddValue, RemoveValue.



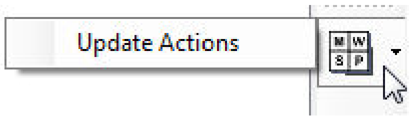
```
and
  equal
  equal
  or
    equal
      Action
      'ModifyEntry'
    equal
      Action
      'AddType'
    equal
      Action
      'RemoveType'
    equal
      Action
      'AddValue'
    equal
      Action
      'RemoveValue'
```

Working with named expressions

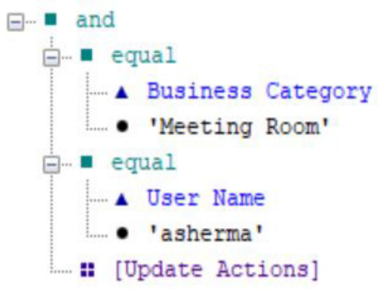
A named expression is an expression that is saved and can then be reused in different rules. If you modify a named expression, the change will affect every rule it appears in.

These steps are not required to define the first rule, but are included in this tutorial to familiarize you with named expressions:

1. In the Expression Tree, right-click the **or** function.
2. Click **Save as a Named Expression**. A window is displayed.
3. Enter `Update Actions` and then click **OK**.
4. Right-click the **or** function, then click **Delete**. The node is deleted from the tree.
5. Right-click the **and** function at the top of the Expression Tree, then click **Add New Argument**. A new 'not-set' node is added.
6. Click the **Named Expressions** button. The named expression 'Update Actions' you just created is displayed.



7. Drag and drop **Update Actions** onto the **not-set** node in the Expression Tree.



```
and
    equal
        Business Category
        'Meeting Room'
    equal
        User Name
        'asherma'
    [Update Actions]
```

   You can view the text version of the named expression by hovering your mouse over it.

8. Click the **Save and Exit** button. The XACML Expression window closes and the condition is displayed in the Condition box of the XACML Rule window.
9. Click **Save**. The rule is added to the Rules area of the Policy Versions tab.
10. To view the named expression:
    a. In the right pane, click the **Policy Versions** tab.
    b. In the first filter box, click **Named Expressions**. The named expressions are listed in the summary area of the tab.
    c. Click the named expression and then click the **Open** button. The XACML Named Expression window is displayed.
    d. Click the **Edit** button. The named expression is displayed in the XACML Expression window.

Define the second rule

To create the second rule in the policy:

1. With **ABAC Rules** and **Access** selected in the filter boxes, click the **New** button. The XACML Rule window is displayed.
2. In the Label box, enter `Search & Read access control.`
3. Enter a short **Description** of the rule, such as `ABAC that permits all users search and read access to all entries.`
4. Click the **Edit** button.The XACML Expression window is displayed.

Define the second rule's condition

The second permits all users to search and view directory entries. It is required because the default behaviour within an Access Control Domain is to deny access, unless explicitly permitted.

The second rule's condition is as follows:

```
Action = 'ReadEntry' OR Action = 'BrowseEntry' OR
Action = 'ReturnDN' OR Action = 'ReadType' OR
Action = 'FilterMatchType' OR Action = 'ReadValue' OR
Action = 'FilterMatchValue' OR Action = 'DiscloseEntryOnError’ OR
Action = 'DiscloseTypeOnError’ OR Action = 'DiscloseValueOnError’
```

To define these expressions:

1. From the Functions Dashboard, drag and drop the **|** function onto the **not-set** node at the top of the Expression Tree. The 'or' function is added to the Expression Tree with two empty nodes below it.
2. In the Expression Tree, right-click the **or** function, then click **Add New Argument**. A 'not-set' node is added to the tree.
3. Repeat the above step until there are ten 'not-set' nodes.
4. From the Functions Dashboard, drag and drop the **=** function onto the first **not-set** node below the **or** function. The 'equal' function is added to the tree with two new empty nodes below it.
5. Click the **Action Attributes** button. The XACML attribute Action is displayed.
6. Drag and drop **Action** onto the fist **not-set** node below the 'equal' function.
7. Double-click the **not-set** node below **Action**. The XACML Value (Enumerated) window is displayed.
8. Choose **ReadEntry** from the dropdown list and click **OK**.
9. Repeat steps 4 and 8 in order to add the following to the remaining not-set nodes:
   - Action = BrowseEntry
   - Action = ReturnDN
   - Action = ReadType
   - Action = ReadValue
   - Action = FilterMatchValue
   - Action = FilterMatchType
   - Action = DiscloseEntryOnError
   - Action = DiscloseTypeOnError
   - Action = DiscloseValueOnError
10. Click the **Save and Exit** button, followed by **Save**.

Activate the policy

For a policy to take effect it must be activated. Only one version of a policy can be active at any time. This ensures that after writing a new version of a policy, you can activate it at an appropriate time and also have the option to roll back by activating the previous version if necessary.

To activate the policy:

1. In the **Policy Versions** tab, click **Version Management** followed by **Activate**. A warning is displayed.
2. Click **Yes**. The policy's Status is now `Active, Open`. This signifies that the rule is in use (active) but can still be modified (open).

Test the policy

You can test the policy by attempting to modify a meeting room entry through Access Presence, first as Andrew Sherma and then as another user. (For the instructions to configure for Access Presence, see Configuring for Access Presence .)

To test the policy:

1. Open the URL: `http://host:8090/directoryservices/viewds/webdua.cgi`
2. Log on with the user name `asherma` and password `testpass`.
3. In the drop-down box, click **Function Search** and then click **Access**. The Advanced Search page is displayed.
4. In the function box, enter `meeting room` and press the return key. A list of meeting rooms is displayed.
5. Click the third meeting room in the list. The entry for the Sales Meeting Room is displayed.
6. Click **Modify**. The Modify page is displayed.
7. Modify the contents of the **Description** box and then click **Save**.
8. Log off by closing the browser session.
9. Repeat this task from step 1, logging on with the user name `rturnbu` and password `testpass`. This user will not be able to modify any entries.

Lock the policy

Once you lock a policy you cannot delete or modify it. You can, however, create a new policy based on an existing policy by clicking the New button in the Policy Versions tab.

To lock the policy:

1. In the **Policy Versions** tab, click the **Version Management** button followed by **Lock**. A warning is displayed.
2. Click **OK**. The policy's Status is now `Active, Locked`.

Tutorial: Role-based access control

This tutorial takes you through how to define an XACML policy that includes role-based access control (RBAC). The policy will apply to an area of the Deltawing directory that is provided with the ViewDS Directory.

It involves adding an RBAC rule to the XACML Access Control Domain that was defined during the previous tutorial.

NOTE: Before starting this tutorial, you should have read Attribute versus role-based access control and completed the Tutorial: Attribute-based access controls.

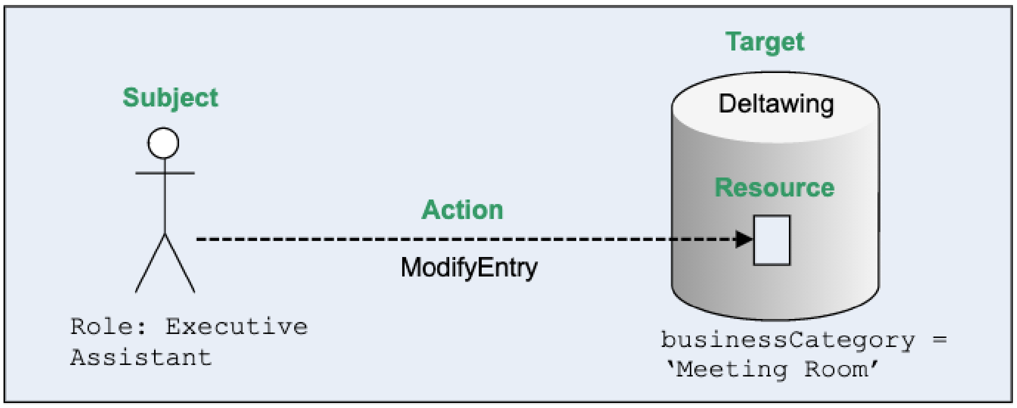
This tutorial has the following stages:

1. Requirements
2. Create a new XACML policy version
3. Define the role hierarchy
4. Define a role attribute
5. Define the RBAC rule
6. Activate the policy
7. Test the policy
8. Lock the policy

Requirements

This tutorial involves creating a new version of the XACML policy that was defined in the Tutorial: Attribute-based access controls.

The requirements include an RBAC rule that permits the role of 'Executive Assistant' to modify meeting room entries in the Deltawing directory.



The policy also requires a role hierarchy that defines that a 'Chief Executive Officer' inherits permissions from an 'Executive Assistant'.

XACML attributes

An XACML attribute must be declared before it can be used in an XACML rule.

Each declaration includes a label that will appear in a rule, plus an XACML category, identifier, and data type. An XACML attribute can also be mapped to a directory attribute.

This tutorial's rule will use XACML attributes declared in the Tutorial: Attribute-based access controls (Action and Business Category) plus an XACML attribute with the following definition:

Label	XACML attribute category	XACML attribute identifier	XACML data type	Directory attribute
Role	urn:oasis:names:tc:xacml:1.0:subject-category:access-subject	urn:oasis:names:tc:xacml:2.0:subject:subject:role	string	title

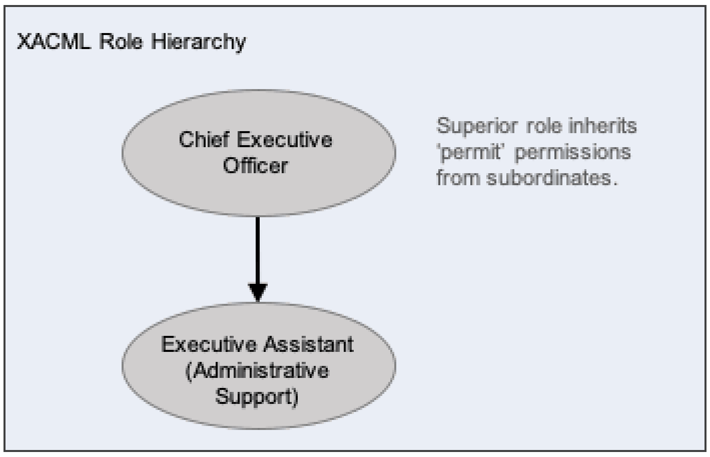
The definition maps the XACML attribute to a directory attribute called `title`.

The value of `title` in a user's directory entry identifies their role. This means that if a user's `title` is 'Executive Assistant', for example, then their role is also 'Executive Assistant'.

Role hierarchy

A role hierarchy is required that includes the role 'Chief Executive Officer' which inherits permissions from the role 'Executive Assistant'.

The 'Executive Assistant' role should also have an alternative name declared. The effect will be that all users with the `title` 'Administrative Support' or 'Executive Assistant' will have the same role and will be granted the same permissions.


XACML Role Hierarchy
Chief Executive Officer
Superior role inherits 'permit' permissions from subordinates.
Executive Assistant (Administrative Support)

RBAC rule

One RBAC rule is required. Its target, scope, effect, role and condition are shown below.

RBAC Rule:

Target: `Deltawing`

Scope: `subtree`

Effect: `Permit`

Role: `Executive Assistant`

Condition:

resource has attribute `Business Category` = 'Meeting Room' AND

```
(Action = 'ModifyEntry' OR Action = 'AddType' OR
Action = 'RemoveType' OR Action = 'AddValue' OR Action = 'RemoveValue')
```

The rule's target will be the entry at the root of the Deltawing directory; its scope will be the entire subtree below the root entry; and its effect will be to permit access if the user's role is 'Executive Assistant' and the condition is true.

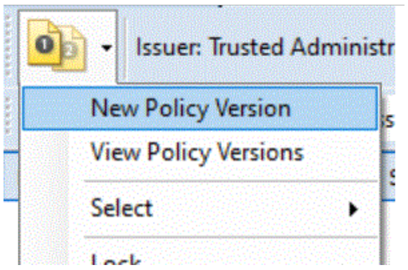
The condition will be true when an 'Executive Assistant' attempts one of the actions on a meeting room entry.

NOTE: A prerequisite of this tutorial is to have completed the Tutorial: Attribute-based access controls. This is because it includes an ABAC rule that permits all users to search and view all directory entries.

Create a new XACML policy version

To create a new version of the policy that was declared and then locked during the previous tutorial:

1. In the left pane, click the **Deltawing** entry at the top of the DIT. The XACML AC tab is displayed in the right pane. This is because the entry is the administrative point for the XACML Access Control Domain created in the previous tutorial.
2. In the right pane, click the **XACML AC** tab and then the **Policy Versions** tab.
3. Click the **Version Management** button followed by **New Policy Version**. The XACML Policy Version window is displayed.

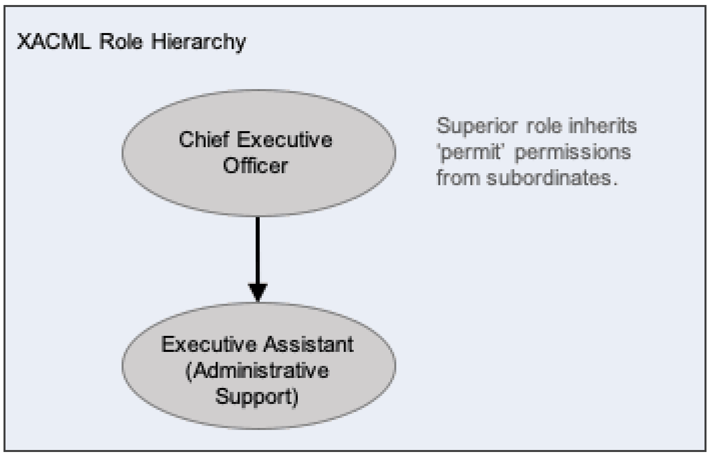


4. Accept the default values by clicking **Save**. A new policy version with the status of open is displayed next to the Version Management button.

Define the role hierarchy

Defining a role hierarchy involves two broad steps. First, declare the roles, and then declare the hierarchical relationship between them.

The roles required for this tutorial are 'Chief Executive Officer' are 'Executive Assistant'. However, you will also declare an alternative name of 'Administrative Support' for 'Executive Assistant'. This will mean that the same role and permissions will be granted to any user whose `title` is 'Administrative Support' or 'Executive Assistant'.


XACML Role Hierarchy
Chief Executive Officer
Executive Assistant (Administrative Support)
Superior role inherits 'permit' permissions from subordinates.

To declare the 'Chief Executive Officer' role:

1. In the right pane, click the **Roles** tab.
2. At the bottom of the Roles tab, click **New**. The New XACML Role window is displayed.
3. Click **Add**. The XACML Role Name window is displayed.
4. Enter the role name `Chief Executive Officer`, then click **OK** followed by **Save**. The role is added to the Roles tab.

To declare the second role:

1. Click **New**. The New XACML Role window is displayed.
2. Click **Add**. The XACML Role Name window is displayed.
3. Enter the role name `Executive Assistant`, then click **OK**.
4. Repeat steps 2 and 3 to add another name for the same role, `Administrative Support`.
5. Click **Save**. Two names for the single role are added to the Roles tab.

To declare that the 'Chief Executive Officer' should inherit permissions:

1. Right-click **Chief Executive officer** and then click **Open**. The XACML Role window is displayed.
2. Below the **Directly inherits permissions from** box, click **Edit**. The XACML Role Selection window is displayed.
3. Click either role name followed by the right-arrow button. Both role names are moved to the box on the right.
4. Click **OK** followed by **Save**.

Declare a role attribute

To declare the XACML attribute for `Role`:

1. In the right pane, click the **Attributes** tab.
2. At the bottom of the right pane, click **New**. The XACML Attribute window is displayed.
3. In the **Label** box, enter `Role`. This is the name that will appear in the rule.

4. In the **Category** box, click **urn:oasis:names:tc:xacml:1.0:subject-category:access-subject**.
5. In the **Identifier** box, click **urn:oasis:names:tc:xacml:2.0:subject:role**.
6. In the **Directory Attribute** box, click **title**.
7. Click **Save**. The Attributes tab displays the new attribute.

For information about the XACML attributes provided by the ViewDS PEP, see the *ViewDS Access Sentinel: Installation and Reference Guide*.

NOTE: Every attribute in an XACML domain must have a unique combination of Category, Identifier and Data Type.

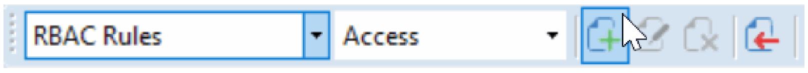
Define the RBAC rule

The required RBAC rule is as follows:

RBAC Rule:
Target: `Deltawing`
Scope: `subtree`
Effect: `Permit`
Role: `Executive Assistant`
Condition:
resource has attribute `Business Category` = 'Meeting Room' AND
(Action = 'ModifyEntry' OR Action = 'AddType' OR
Action = 'RemoveType' OR Action = 'AddValue' OR Action = 'RemoveValue')

To create an RBAC rule:

1. In the right pane, click the **Policy Versions** tab.
2. With **RBAC Rules** and **Access** selected in the filter boxes, click the **New** button. The XACML Rule window is displayed.



3. In the **Label** box, enter `Exec Assistant & Admin Support meeting room access`.
4. Enter a short **Description** of the rule, such as `RBAC rule that permits Exec Assistant & Admin Support full access to meeting room entries.` Note that the Target is Deltawing and its Scope is subtree. Hence, the target is all subtrees and entries subordinate to Deltawing. Also note that the Effect is set to the default, permit.

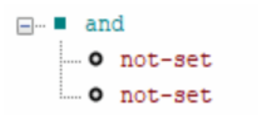
To define the role to which the rule applies:

1. Click the **Edit** button next to the **Role Expression** box, The XACML Role Expression window is displayed. The names of the roles you declared earlier are listed on the left.

2. Click **Executive Assistant** followed by the right arrow. The role name is moved to the box on the right.
3. Click **OK**. The role is added to the Role Expression box. Note that this name applies to a single role, which also has the name 'Administrative Support' assigned to it.

The remaining expressions are defined through the XACML Expression window:

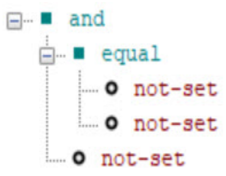
1. Click the **Edit** button next to the **Condition** box. The XACML Expression window is displayed.
2. Drag and drop the **&** from the Functions Dashboard to the node at the top of the expression tree. The function is displayed in the expression tree with two empty nodes below it.



```
■ and
  ○ not-set
  ○ not-set
```

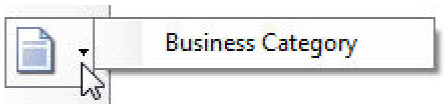
Note: To replace a function, drag and drop another function on top of it.

3. Drag and drop the **=** from the Functions Dashboard onto the first **not-set** node in the expression tree.

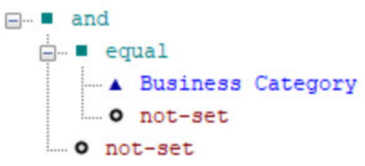


```
■ and
  ■ equal
    ○ not-set
    ○ not-set
  ○ not-set
```

4. On the left of the window, click the **Resource Attributes** button.

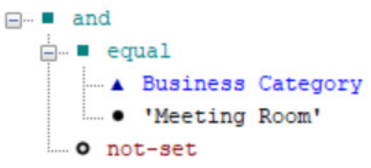


5. Drag and drop **Business Category** onto the fist **not-set** node below the equal function.



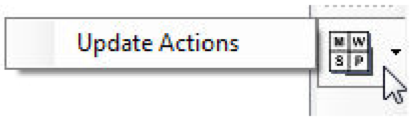
```
■ and
  ■ equal
    ▲ Business Category
    ○ not-set
  ○ not-set
```

6. Double-click the **not-set** node below Business Category. The XACML Value (String) window is displayed.
7. In the **Value** box, enter `Meeting Room` and then click **OK**. The string is added to the expression.

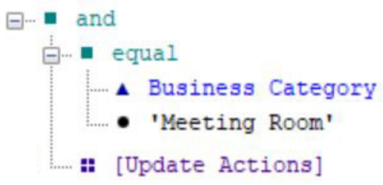


```
■ and
  ■ equal
    ▲ Business Category
    ● 'Meeting Room'
  ○ not-set
```

8. Click the **Named Expressions** button. The named expression 'Update Actions' that was created during the previous tutorial is displayed.



9. Drag and drop **Update Actions** onto the **not-set** node in the Expression Tree.



```
and
  equal
    Business Category
    'Meeting Room'
  [Update Actions]
```

   You can view the text version of the named expression by hovering your mouse over it.

10. Click the **Save and Exit** button. The XACML Expression window closes and the condition is displayed in the Condition box of the XACML Rule window.

11. Click **Save**. The rule is added to the Rules area of the Policy Versions tab.

Activate the policy

For a policy to take effect it must be activated. Only one version of a policy can be active at any time. This ensures that after writing a new version of a policy, you can activate it at an appropriate time and also have the option to roll back by activating the previous version if necessary.

To activate the policy:

1. In the **Policy Versions** tab, click **Version Management** followed by **Activate**. A warning is displayed.
2. Click **Yes**. The policy's Status is now `Active, Open`. This signifies that the rule is in use (active) but can still be modified (open).

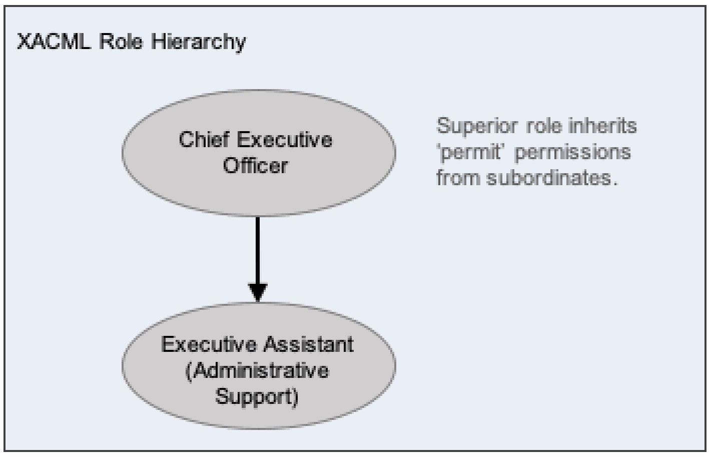
Test the policy

You can test the policy by attempting to modify a meeting room entry through Access Presence. (For the instructions to configure for Access Presence, see Configuring for Access Presence .)

Testing involves logging onto Access Presence as the following users:

Deltawing user	Role (title)	Username	Password
Maria Guglielmino	Administrative Support	mguglie	testpass
Robert Turnbull	Executive Assistant	rturnbu	testpass
Margaret Hunter	Chief Executive Officer	mhunter	testpass
Karen Johannesen	Director	kjohann	testpass

All the above users, except Karen Johannesen, will be able to modify a meeting room entry. Note that Margaret Hunter has the role 'Chief Executive Officer', which inherits permissions through the role hierarchy.


XACML Role Hierarchy
Chief Executive Officer
Superior role inherits 'permit' permissions from subordinates.
Executive Assistant (Administrative Support)

To test the policy:

1. Open the URL: `http://host:8090/directoryservices/viewds/webdua.cgi`
2. Enter a username and password `testpass`.
3. In the drop-down box, click **Function Search** and then click **Access**. The Advanced Search page is displayed.
4. In the function box, enter `meeting room` and press the return key. A list of meeting rooms is displayed.
5. Click a **meeting room** entry. The entry's details are displayed. If the user has permission to modify the entry, then the Modify button is displayed.
6. Log off by closing the browser session.
7. Repeat this task for users with different roles.

Lock the policy

Once you lock a policy you cannot delete or modify it. You can, however, create a new policy based on an existing policy by clicking the **New** button in the Policy Versions tab.

To lock the policy:

1. In the **Policy Versions** tab, click **Version Management** followed by **Lock**. A warning is displayed.
2. Click **OK**. The policy’s Status is now `Active, Locked`.

Key concepts: Distribution and replication

This section introduces you to directory distribution and replication, and includes high-level guidance to help you adapt ViewDS to your requirements.

It covers the two topics:

- Understanding distribution and replication
- Distributing or replicating a DIT

Understanding replication and distribution

A Directory Information Tree (DIT) can be replicated and/or distributed.

Some or all of the data in a Directory System Agent's (DSA's) DIT can be **replicated** to another DSA to provide, for example, fail-over, load balancing or public access to part of a DIT.

A DIT might be **distributed** so that different organizations can manage their own data while still allowing access by another associated organization. This might be desirable when, for example, a company is spread across different countries with a different directory in each.

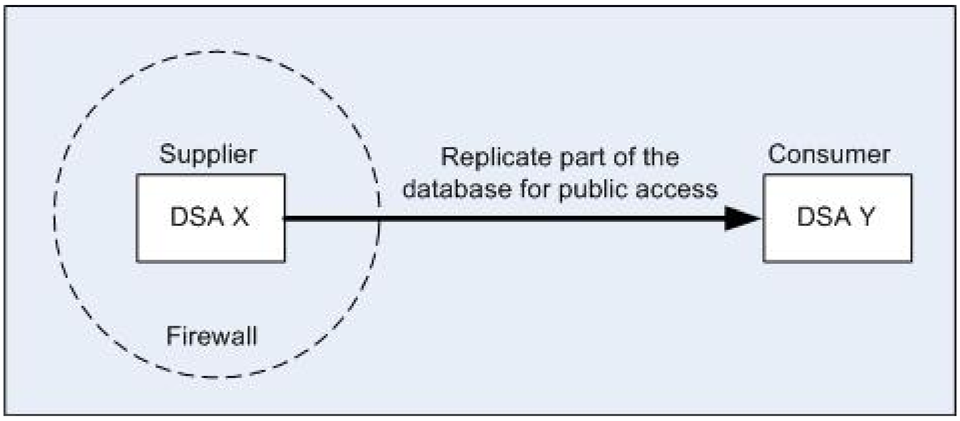
The key concepts are:

- peer trust
- replicating
- distributing
- chaining operations to a superior DSA

Peer trust

Before a DIT, or part of a DIT, can be replicated or distributed between DSAs, both DSAs need peer information about each other. The peer information tells each DSA which other DSAs it can trust, and the incoming and outgoing levels of trust.

For example, consider an implementation with two DSAs. The first DSA, *DSA X*, is within a firewall, and part of its DIT is replicated on *DSA Y*. This second DSA is outside the firewall and provides public access to the replicated area of the DIT.



In this scenario, *DSA X* does not trust *DSA Y* to authenticate access to its data. *DSA Y*, however, does trust *DSA X* to authenticate access to its copy of the data.

Replicating

A replication agreement comprises a supplier DSA and a consumer DSA. The supplier DSA provides a shadow copy of its data to the consumer DSA. In the above illustration, *DSA X* sends a shadow copy of its data to *DSA Y*, the consumer.

DSA Y can service queries on its copy of the data, but cannot modify it directly. When a user modifies data through *DSA Y*:

1. *DSA Y* sends the modifications to *DSA X*, which updates its database.
2. *DSA X* sends the modifications back to *DSA Y* to be replicated in its database.

ViewDS implements the X.500 Directory Information Shadowing Protocol (DISP).

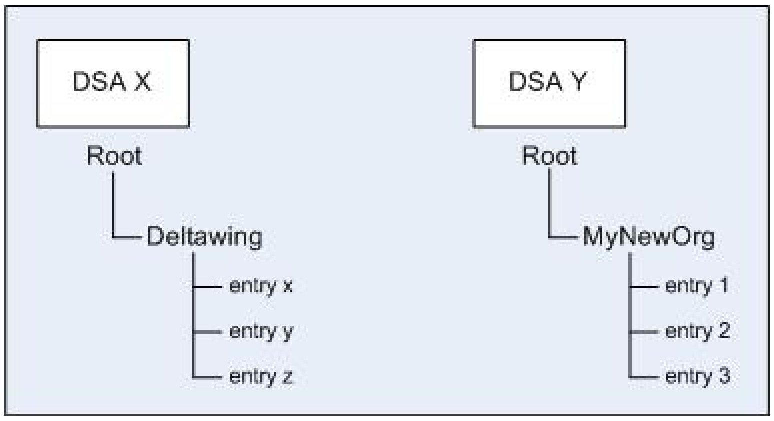
Distributing

There are two types of distribution: *collaboration* and *delegation*.

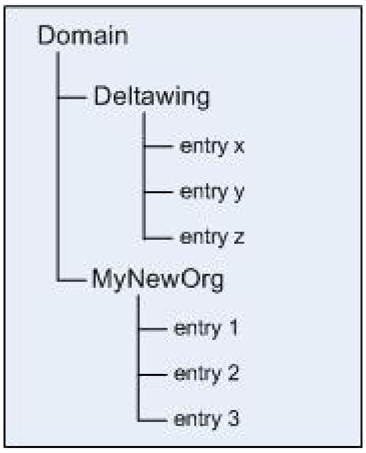
The DSAs in a distributed environment must have *knowledge* of each other. Knowledge is the information a DSA needs to locate an entry in another DSA.

Collaboration

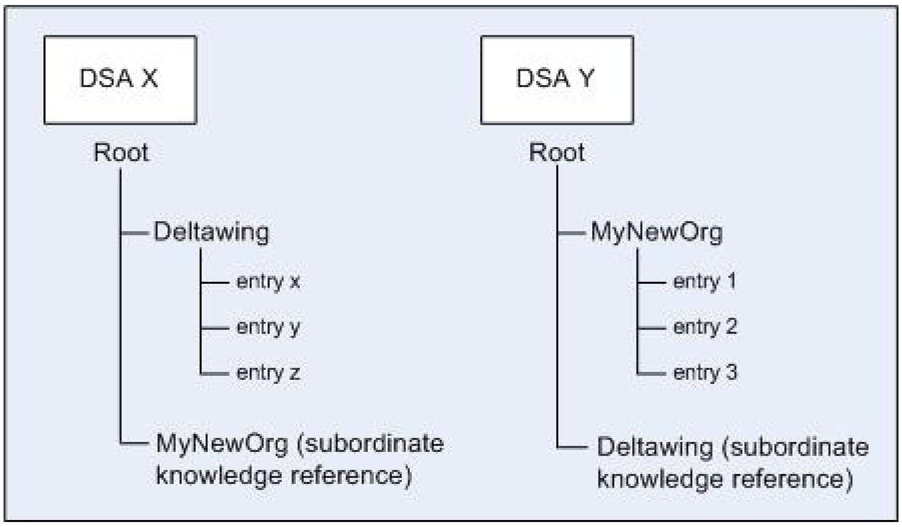
Consider two DSAs, *DSA X* and *DSA Y*:



With a collaboration, users would see the following DIT:

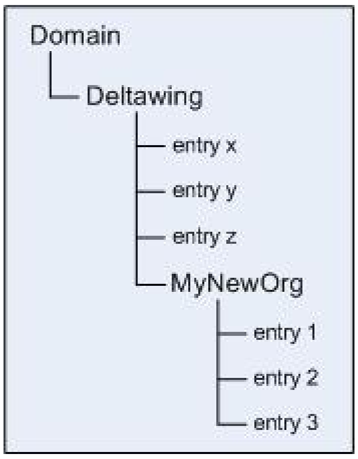


Setting this up involves adding an entry to each DIT that contains a **knowledge reference**. Each knowledge reference points to the other DSA.


DSA X
Root
Deltawing
entry x
entry y
entry z
MyNewOrg (subordinate knowledge reference)
DSA Y
Root
MyNewOrg
entry 1
entry 2
entry 3
Deltawing (subordinate knowledge reference)

Delegation

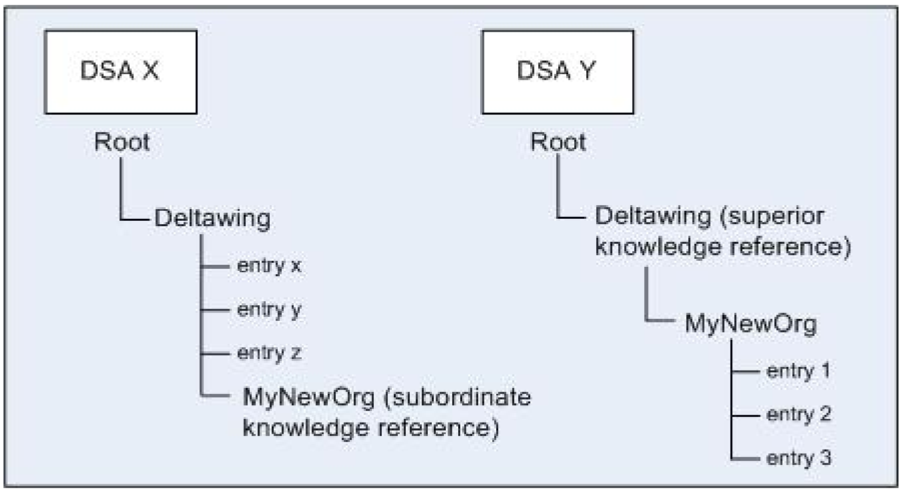
Using the same example DSAs, a delegation provides the following DIT to users:



Setting this up involves making *DSA X* the **superior DSA** by adding an entry to its DIT that contains a **subordinate knowledge reference**. When you create this entry, the ViewDS Management Agent automatically configures the rest for you as follows:

- it makes *DSA Y* the **subordinate DSA** by adding an entry to its DIT that contains a **superior knowledge reference**; and
- positions the knowledge reference in the DIT so that the hierarchy reflects that of the DIT of the superior DSA.

This is illustrated below. Each knowledge reference points to the other DSA's DIT.


DSA X
Root
Deltawing
entry x
entry y
entry z
MyNewOrg (subordinate knowledge reference)
DSA Y
Root
Deltawing (superior knowledge reference)
MyNewOrg
entry 1
entry 2
entry 3

Chaining operations to another DSA

Every DAP and LDAP operation includes a base-entry name. Before a DSA can process such an operation, it *resolves* the base-entry name to an entry in its DIT. The DSA then applies the operation to this entry. When a DSA's DIT is distributed, it chains operations that it cannot resolve to another DSA with better knowledge of the location of the entry.

In the above illustration, for example, if either DSA receives a search operation that it cannot resolve, then it will pass the operation to the other DSA. This DSA will then attempt to resolve the operation (and may potentially pass the operation to yet another DSA).

Distributing or replicating a DIT

For the steps to distribute a DIT, see the topic Distribute a DIT in the ViewDS Management Agent help. In brief:

1. Configure the DSAs for distribution.
2. Define peer trust between the DSAs.
3. Determine which type of distribution you will be using, collaboration or delegation.
4. Create knowledge references for each DSA.

For the steps to replicate a DIT, see the topic Replicate a DIT in the ViewDS Management Agent help. In brief:

1. Configure the DSAs for replication.
2. Define peer trust between the DSAs.
3. Create a replication agreement between the DSAs.