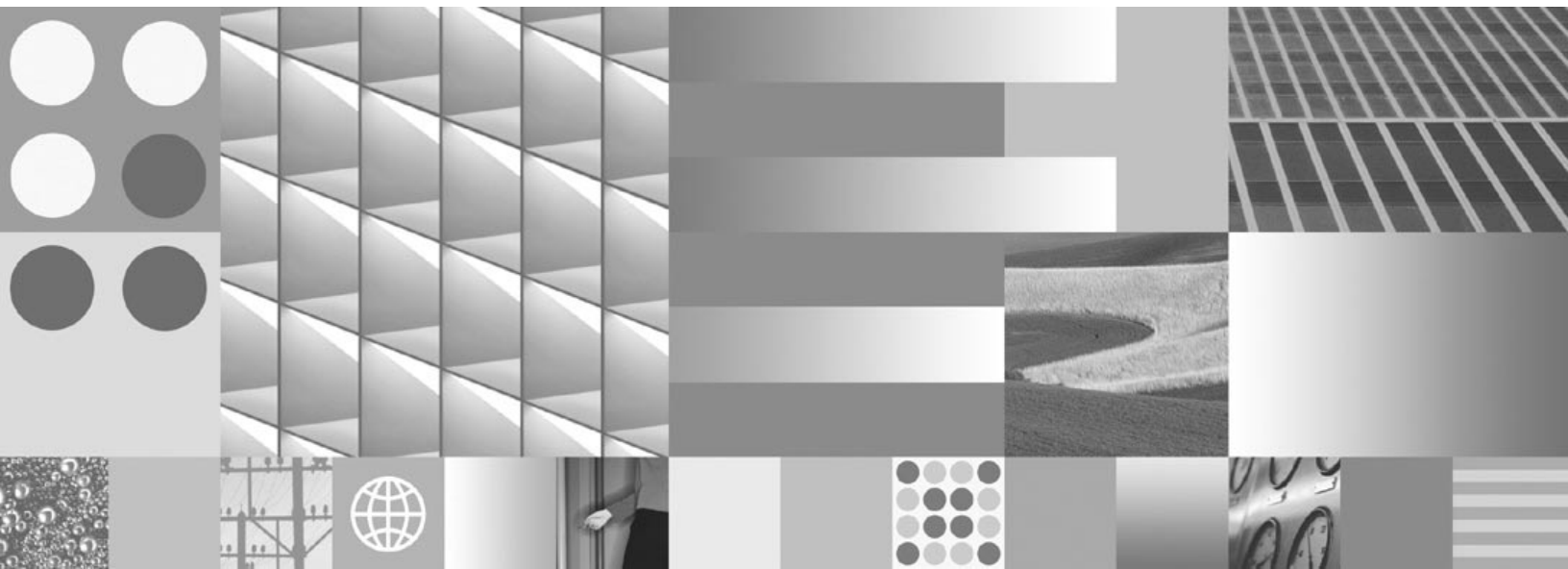


Security Help



Security Help

Note

Before using this information and the product it supports, read the information in "Notices" on page 267.

This edition applies to version 4.0.2 of IBM FileNet Business Process Manager (product number 5724-R76), version 4.0.1 of IBM FileNet Content Manager (product number 5724-R81), version 4.0.1 of IBM FileNet eForms for P8 (product number 5724-R85), version 4.0.0 of IBM FileNet Records Manager (product number 5724-S19), and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2006, 2007. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Revision Log	13
FileNet P8 security	14
Security overview	15
How does FileNet P8 secure its objects?	15
FileNet P8 domain	15
How Access Control Works	15
Independently and dependently securable objects	15
ACL and ACE model.....	15
Importance of installation.....	16
Markings	17
Modification access required	17
Access to content - querying and exporting.....	17
How is security applied?.....	17
User authentication.....	17
Configuring authentication.....	18
Security cache	19
Security administrators and security tools.....	19
Authentication	21
Supported authentication standards.....	21
JAAS overview.....	21
JAAS trust mechanisms.....	22
WS-Security overview.....	23
FileNet P8 Platform server authentication architecture.....	24
Content Engine authentication architecture.....	24
Process Engine authentication architecture	25
JAVA-based client authentication (JAAS).....	26
Browser-based clients of J2EE application servers	26
Application-managed authentication.....	26
Container-managed authentication.....	26
J2EE thick Java client.....	28
Support for Content Engine 3.5 Java API clients	29
Support for Java applets.....	29
Applets and reverse proxy servers	29
Single sign-on integrations via JAAS	31
CA/Netegrity SiteMinder Web Agent.....	31
WebSeal from IBM Tivoli Access Manager (TAM)	33
Web-Services-based client authentication via Ws-Security.....	35
Username token credentials.....	35
Kerberos Credentials.....	36
Kerberos overview	36
Microsoft's use of Kerberos	37
Sample Kerberos token	37
Web Service extensible authentication framework	38
SAML credentials	39
Java-based clients over the web service transport	39
Process Engine authentication	40
Authenticating the Process Engine Java API client using Content Engine EJB Transport.....	40
Authenticating the Process Engine Java API client using the Content Engine web service transport ...	41
The Process Engine Web Service API client	42
Application Engine authentication.....	44
Workplace XT application.....	44
Workplace application	44
Application-managed authentication.....	44

Container-managed authentication.....	44
Workplace applets.....	45
Application Integration clients.....	46
FileNet P8 Portlets clients.....	46
WebDAV clients.....	46
Application-managed authentication.....	47
Container-managed authentication.....	47
Kerberos for Content Engine.....	48
Introduction.....	48
Prerequisites.....	48
System prerequisites.....	48
Domain / account prerequisites.....	48
Stand-alone .NET client prerequisites.....	49
CE Java server prerequisites.....	49
Middleware, web application servers.....	49
Creating the Kerberos Service Principal Name (SPN) identity.....	50
Some Background.....	50
Enabling Kerberos on the application server.....	52
WebLogic.....	52
WebSphere.....	53
JBoss.....	54
KrbServiceLoginModule options.....	54
Using Kerberos on the client.....	55
Using Kerberos with a cluster of CE servers.....	55
Cross-Domain authentication.....	56
Solving Kerberos problems.....	57
.NET client or EM problems.....	57
CE server problems.....	59
Authorization.....	62
About access rights.....	63
ACE name.....	64
ACE source: Default, Direct, Inherited, Template.....	64
ACE security levels.....	64
Allow/Deny and order of evaluation.....	64
Default security.....	66
Object store administrative groups.....	66
End users.....	66
Security for integrated components and third-party products.....	67
Browsers.....	68
Database security.....	69
Security for Autonomy K2 server.....	70
Security for fixed content providers.....	71
FileNet Content Federation Services Server for Image Services.....	71
EMC Centra.....	71
SnapLock.....	71
Security for FileNet P8 eForms.....	72
Security for Records Manager.....	73
Mapping security levels to individual access rights.....	74
FileNet P8 domain root security levels.....	75
Object store security levels.....	76
Class definition security levels.....	77
Folder security levels.....	78
Document security levels.....	79
Custom object security levels.....	80
Other object security levels.....	81
Markings.....	82
Overview.....	82

Limitations	83
Marking security: Add, Remove, Use	83
Add marking and Remove marking	83
Use Marked Objects	83
Effect of Copy to reservation	84
Constraint Mask.....	84
Allow, Deny permissions	85
Allow.....	85
Deny.....	85
Hierarchical and Non-hierarchical	86
FileNet P8 Domain resource	87
Marking Set TTL (caching)	87
Active markings	87
Marking administration (create, modify, remove)	87
Not available with choice lists.....	88
Object access rights and security	89
Securable objects.....	90
Object store access rights.....	91
Relationship of object store permissions to permissions on objects contained by the object store.....	91
Workplace and Workplace XT	91
Class access rights	92
Class security tabs	92
Classes receive default security from their parent class	92
New objects acquire initial security from a class	92
Folder access rights	93
Folder access rights	93
Default security	93
Folder behavior using the Workplace or Workplace XT applications.....	93
Document access rights.....	94
Access rights defined	94
Initial security acquired from the document class	95
Content element security.....	95
Browsing vs. searching.....	95
Compound document security	96
Required access rights for parent and child components	96
Component Relationship Actions and Parent Component Requirements	96
Requirements for Child Component Actions	97
Administration.....	97
Document lifecycle policies and lifecycle actions access rights	98
Document lifecycle policy access rights.....	98
Document lifecycle action access rights.....	98
Annotation access rights.....	100
Annotation access rights defined	100
Custom object access rights	101
Access rights defined	101
Workflow definition access rights	102
Acquisition	102
Inheritance.....	102
Workflow attachments	102
Event and subscription access rights.....	103
Access rights defined	103
Script access rights	104
Publishing access rights.....	105
A publish template acquires security from the Publish Template class	105
Access rights defined	105
A publication document has the security specified in the publish template	105
Republishing.....	105

Deleting the source.....	106
Style template.....	106
Search access rights.....	107
Acquisition and inheritance.....	107
Access rights defined.....	107
Using searches in Workplace and Workplace XT.....	107
Document entry template access rights.....	108
Workflow rosters and queues.....	109
Workflow security.....	109
Important tips regarding security.....	110
Object ownership.....	111
Object ownership.....	111
Default Instance Security and Ownership.....	111
NULL Owner.....	112
Changing Ownership.....	112
Creator-Owner substitution.....	112
Property modification access.....	113
About MARs.....	113
Normal property security (no modification access).....	113
Property security if modification access has been configured.....	114
Exceptions to the normal access requirements.....	115
Required minimum access rights by operation.....	116
Create minor version.....	116
Create major version.....	117
Checkout minor version.....	117
Checkout major version.....	117
Cancel checkout.....	117
Check in minor version.....	118
Check in major version.....	118
Promote or demote.....	118
Delete minor or major version.....	118
Security policies.....	120
About security policies.....	120
About templates.....	120
Assigning security policies.....	121
Preserve Direct ACEs.....	121
Effects of changes.....	121
Rules of association.....	122
Support in Workplace and Workplace XT.....	122
Importing and exporting security policies.....	123
Custom objects and folders use application security templates only.....	123
Storage area security.....	124
General.....	124
File storage area security.....	124
Content Engine operating system account.....	124
Shared root directory - Windows and UNIX.....	125
Inherited Security - Windows file system.....	125
Sharing the root folder - Windows.....	126
Inherited Security - UNIX file system.....	126
Content Engine on UNIX, file storage on Windows.....	126
Content cache area security.....	127
Target access required.....	128
Understanding security inheritance.....	129
General description.....	129
Inheritable depth (Apply to).....	129
Security Parent, Security Folder, Security Proxy Type.....	130
Configuring inheritance.....	130

Subclassed permissions are copied, not inherited.....	131
Summary of security sources	131
Directory service providers.....	133
Introduction.....	133
Terminology and basic concepts.....	133
Configuration Overview	135
Directory Configuration for Authentication.....	135
Directory Configuration for Authorization.....	136
Sun Java System Directory Server	137
General.....	137
Support matrix	137
Directory Configuration Properties	139
Realm Configuration (WebLogic)	141
Authentication Provider Configuration, using WebLogic	141
GCD Configuration.....	143
Configuration Result	145
Operation.....	145
Get User or Group by Short Name	145
Get User or Group by DN	145
Get User or Group by UPN.....	145
Get User or Group by SID	145
Search Users or Groups in a Given Realm	146
Novell eDirectory.....	147
General.....	147
Support matrix	147
Directory Configuration Properties	148
Realm Configuration (WebLogic)	150
Authentication Provider Configuration	150
GCD Configuration.....	152
Configuration Result	153
Operation.....	154
Get User or Group by Short Name	154
Get User or Group by DN	154
Get User or Group by UPN.....	154
Get User or Group by SID	154
Search Users or Groups in a Given Realm	154
IBM Tivoli Directory Server	155
General.....	155
Support matrix	155
Directory Configuration Properties	156
Realm Configuration (WebLogic)	157
Authentication Provider Configuration, using WebLogic	159
GCD Configuration.....	160
Configuration Result	161
Operation.....	162
Get User or Group by Short Name	162
Get User or Group by DN	162
Get User or Group by UPN.....	162
Get User or Group by SID	162
Search Users or Groups in a Given Realm	162
Windows Active Directory	163
Support matrix	163
Directory Configuration Properties	164
Realm Configuration (WebLogic)	166
Single-Realm Configuration.....	166
Multi-Realm Configuration	168
Entire Forest Configuration.....	169

Partial Forest Configuration	171
Operation.....	177
Get User or Group by Short Name	177
Get User or Group by DN	178
Get User or Group by UPN.....	178
Get User or Group by SID	178
Search Users or Groups in a Given Realm	178
Windows Active Directory Application Mode (ADAM).....	179
Support matrix	180
Directory Configuration Properties	182
Realm Configuration.....	184
Operation.....	188
Get User or Group by Short Name	188
Get User or Group by DN	188
Get User or Group by UPN.....	188
Get User or Group by SID	189
Search Users or Groups in a Given Realm	189
Directory service performance.....	190
Sun Java System Directory Server	190
Glossary definitions.....	190
Additional information	190
Configuring sorting for Novell eDirectory.....	195
Users and groups required by FileNet P8 Platform	196
Required for installation - summary.....	196
Required for database creation and connectivity - summary	196
Required for administration - summary	197
Required for Workplace and Workplace XT applications - summary.....	198
Required for internal operations - summary.....	198
Required for installation - details	198
Content Engine system user.....	198
WebLogic administrator	199
WebSphere administrator	200
Install Content Engine.....	200
Publishing user account.....	201
Logon requirements for running Process Engine Setup (all platforms).....	201
Accounts required by Process Engine Setup (Windows)	202
Accounts required by Process Engine Setup (UNIX)	203
Install Application Engine.....	203
Required for database creation and connectivity - details	204
Content Engine SQL Server account.....	204
Create Oracle tablespaces for Content Engine	204
Content Engine DB2 account	205
Rendition Engine SQL Server account.....	205
Required for administration - details.....	206
GCD administrator role	206
Object store administrator role.....	207
Directory service user (Content Engine).....	209
Operating system account (Content Engine).....	210
K2 security group account (Content Engine/Autonomy K2)	210
K2 security user account (Content Engine/Autonomy K2)	211
K2 operating system user (Autonomy K2).....	211
Service user (Process Engine)	212
Process Engine administrator group.....	212
Process Engine configuration group.....	212
Application Engine administrator role	213
CFS for IS User.....	213
Required for Workplace and Workplace XT applications - details	213

PSConsole	213
PSDesigner	214
PWAdministrator	214
PWConfiguration	215
PWDesigner	215
Required for internal operations - details	215
#AUTHENTICATED-USERS	215
#CREATOR-OWNER	216
Accounts required by Process Engine to access the workflow database	217
Accounts required by Process Engine for internal use (created by Process Engine Setup).....	218
Security tools and procedures	219
CE Bootstrap properties.....	220
Sample CEMPBoot.properties file.....	220
Edit CE Bootstrap properties with the Bootstrap Configuration Utility	221
Usage.....	221
Example	221
Digital signing.....	223
Enterprise Manager's security editor.....	224
Enterprise Manager's Select Users and Groups dialog box	226
How Workplace applications display security	227
What Workplace and Workplace XT users see.....	227
Permissions	227
Site and user preferences	229
Logging on and using Enterprise Manager	230
Network security.....	231
Content Engine Encryption	232
Content Engine credential encryption	232
Encrypting credentials with the Master Key	232
Procedure for resetting keys	233
Symmetric encryption of sensitive data at rest in the GCD.....	233
Asymmetric encryption of sensitive data sent over the wire	233
Content encryption	234
SSL and data privacy	235
How to.....	236
Add users and groups to a class.....	237
Add users and groups to a single object.....	238
Add or remove a GCD administrator.....	239
Add or remove an object store administrator	240
Allow or disallow security inheritance	241
Allow users to add items to a folder	242
Configure a folder's security inheritance	243
Use Enterprise Manager to configure a security folder	243
Configure security inheritance	244
Use Enterprise Manager to designate a folder as a security parent, using Security Parent.....	244
Use Enterprise Manager to designate a folder as a security folder, using Security Folder	245
Use Enterprise Manager to configure security inheritance using a custom object-valued property	245
Configure inheritable depth (Apply to)	248
Configure multiple authenticating attributes.....	249
Configure multiple realms	251
Deny an object store administrator access to a document	253
Modify an object's security	255
Restrict access to the root folder	256
Set security on workflow queues and rosters	257
Take or change ownership.....	258
Update object store with new users and groups	259
To update an object store with new users or groups.....	259
Security example.....	260

Analysis	261
Describe the work	261
Determine the access rights required for the work	261
List the data objects	262
Setup	263
Create users and groups in the configured authentication provider's directory service.	263
Define document classes in FileNet Enterprise Manager.....	263
Define the workflow using Process Designer.	264
Define the publish template using Publishing Designer.	264
Create folders using FileNet Enterprise Manager.	264
Frequently asked questions about FileNet P8 Platform security	266
Questions	266
Answers	266
Notices	267
Trademarks	268

Revision Log

Date	Revision
June 26, 2007	FileNet P8 4.0.1a
September 17, 2007	FileNet documentation release 4.0.2

FileNet P8 security

NOTE This document is a stand-alone rendition of the Security topics that are otherwise built into the html-based Enterprise-wide Administrative help. Because it is intended only for printing, the hyperlinks that are in the html version are not live in this PDF.

This Help section is written for system and security administrators. It explains the security features, behavior, and tools that provide the security context for your FileNet® P8 installation. This context includes a combination of security features that are inherent to FileNet P8 and to the application servers and directory services that it supports.

The security of your FileNet P8 installation is provided by several important components. How these components relate to each other is explained in the Security overview topic.

The procedures for installing and initially configuring for authentication and authorization are explained in the FileNet P8 Platform Installation and Upgrade Guide . This “FileNet P8 security” section, especially the Directory service provider integration topic, provides reference information for the installation process. In addition, the Users and groups required by FileNet P8 Platform topic provides a detailed reference for all the user and group accounts required by the installation process. Once your system is installed, you must design and implement a security model that fills the needs of your site and any customized applications. To do this, you should have a good understanding of how users are authenticated to access FileNet P8 and how the product goes about securing stored objects (for example, documents, folders, and events).

There are security-related topics in other FileNet P8 Help sections, especially:

Security Concepts in the *Content Engine API Developer's Guide*.

About Application Engine security in *Application Engine Security*

Work with security in *Help for Workplace*

Security in *Help for Workflow*

Security editor (for Enterprise Manager) in *Content Engine Administration*

Security overview

This topic describes FileNet P8 security architecture in general terms. It focuses primarily on inherent FileNet P8 features, rather than attempting to explain JAAS, Web Services, or the associated capabilities of the application servers and directory services on which FileNet P8 relies for authenticating users.

How does FileNet P8 secure its objects?

In FileNet P8, you secure data by specifying the directory service user context that controls who logs on to the system, setting access rights for those users, creating and applying security policies, and setting site preferences for user actions. Optionally, you can use technologies such as firewalls and proxy servers to control access to your network and Secure Sockets Layer (SSL) to provide data privacy.

FileNet P8 domain

Similar to other domain architectures, the FileNet P8 domain is a defined security context: only those users, groups, machine accounts (services, etc.), applications, processes, and scripts, that have been explicitly granted access to the FileNet P8 domain can access its resources (objects, documents, workflows, etc.).

How Access Control Works

After a user has been *authenticated*, but before he or she is allowed to perform any operations in an object store, a security token is generated specifically for that user's account that serves as a sort of internal passport. Like a passport, the security token is presented to an object at the time access is requested and it is used by the object to determine who the user is and whether or not the user is *authorized* for the type of access that is requested.

The security token contains the security identifier (SID) of the user, and the SIDs of all groups that the user belongs to, based on user and group information in the directory server.

Each securable Content Engine object has an associated security descriptor, part of which is the Access Control List (ACL). The ACL consists of a set of Access Control Entries (ACEs), also called permissions. Each permission specifies one security principal (user or group, via a SID), and an access mask for that SID. The access mask defines the specific operations that the grantee identified by the SID is allowed to perform. Each bit in the mask corresponds to a specific operation. If the bit is set, the security principal is authorized to perform that operation.

The above describes "normal" authorization, where markings have not been implemented. Markings provide another layer of authorization, described below.

Independently and dependently securable objects

Most Content Engine object classes are independently securable, meaning they are secured by their own ACL. Some are dependently securable, meaning that their security depends on some other object that they are associated with. A good example is the Content Element object, which can only exist in relationship to a Document object. The Document object is independently securable, while its various Content Elements take their security from the Document they are assigned to.

ACL and ACE model

Securable objects are secured by ACLs comprised of ACEs. Understanding the ACL/ACE model, and how security principals are determined and permissions granted is central to understanding how to design a secure FileNet P8 system.

First comes authentication, then comes authorization. Users are first authenticated to the FileNet P8 domain, which corresponds to a particular user context in the associated authentication provider. This determines that users are who they say they are, by validating their login user name and password. Thereafter, authenticated users are authorized to do certain things by the security of individual objects.

For example, once the authentication process determines that you are in fact the user Bob who is known to the authentication provider, then the ACL on each securable object will determine what actions Bob can perform on the object (including nothing at all!)

Users and groups that are authenticated to access a FileNet P8 system arrive into the FileNet P8 security context with no inherent permissions of any sort. It is up to the object store administrator to assign FileNet P8-specific permissions to these accounts. For example, a user who is a Windows® domain administrator is just like any end user to FileNet P8. The end user could be granted FileNet P8 object store administrative permissions while the Windows domain administrator could be granted nothing more than view or read-content permissions. In other words, the security context of the directory service or other software applications has no applicability to the FileNet P8 security model (and vice versa). The only relationship may be that a directory configuration (a set of authenticated user accounts) for a FileNet P8 domain may be defined as equal to a specific user context in a directory service).

Some other security features applied to ACEs and secured objects are:

Allow and deny Each ACE that is present on an object's ACL is either allowed or denied the right to do certain things. For example, a particular class of documents could allow Alice to delete a document but deny Bob the same right. Following standard practice, deny always takes precedence over allow, which means you must set up ACLs carefully. If Alice is allowed access to a document as a user but belongs to a group that is denied access, then she will not have access to the object.

Inheritable depth Certain rights are inheritable. FileNet P8 lets you configure whether they should not be inherited, or inherited only by objects that are immediate children (first generation only), or by all children.

Ownership Most objects have an owner, who is usually the user who created the object. Similar to the Windows "built in" accounts, FileNet P8 automatically applies an internal "special" user account called the Creator-Owner, obviously suggesting that the creator of an object is its owner. System administrators can take ownership when necessary to change the object's security.

Authenticated users The other "special" account, internally maintained by Content Engine, is the logical group #AUTHENTICATED-USERS. All users and groups who can potentially log in to FileNet P8 are automatically considered members of #AUTHENTICATED-USERS. This makes it the FileNet P8 equivalent of the Windows "Everyone" group, and makes it easy to allow or deny permissions to everyone who can log in, irrespective of exactly who those users are at any given time.

Importance of installation

As you would expect, important security decisions are made during installation. Among these are:

- Which authentication provider will be used, where it is located, and how FileNet P8 will integrate with it. During installation you will select your directory service or authentication provider and then provide values for integration parameters. For example, for non-Windows Active Directory authentication you would instruct FileNet P8 whether or not to chase any referrals that FileNet P8 might find. (A referral is an attribute that specifies the URL that should be returned for an entry not belonging to the local tree.) You would provide this value based on your knowledge of your directory service topology and your specific authentication requirements.
- Whether you will plug in an SSO solution to your application server's authentication provider.
- Which users/groups in your directory service should be FileNet P8 administrators (there are several different administrative roles required by FileNet P8) and which should be FileNet P8 end users. Both types receive initial, default permissions (documented in Object Access) which you can later modify if necessary.
- How and where will SSL be used (if at all) between FileNet P8 components, associated applications, and supported platforms. FileNet strongly recommends SSL in production environments.

Markings

Markings provide an additional, optional layer of security that is primarily designed for the records management marketplace, but which can also be applied by non-records management applications.

Modification access required

If your installation requires finer-grained security, you can configure a modification access mask that specifies access rights needed to view or modify custom properties. These access rights take precedence over those applied to the object itself. Like markings, this security feature is designed for records management applications, but can be used by any application.

Access to content - querying and exporting

All files that represent document content and that have been stored in a FileNet P8 storage area (file storage area, fixed storage area, or database storage area) are completely protected by the FileNet P8 security model. Only those users who have been granted access to the content by a FileNet P8 application will have access, whether through content-based retrieval (CBR) queries, direct SQL queries, navigation of the network file structure containing a file storage area, etc.

How is security applied?

Normally, an object's security is controlled or determined in four ways. (Markings, if they are used, would be a fifth way.) The following are brief descriptions.

Default instance security As an integral part of the class and instance design, objects such as documents, folders, and events are instances of their class. The class includes, among other things, a property containing the default security permissions that will be applied to all instances of the class. This is the simplest method of applying security: the security design sets up the default security that all instances of a class should have, and then all objects based on that class will have the same default security.

Security parent and inheritance Permissions can also be inherited from a parent object. Inheritance can take place between a class and its subclass, and between a folder and its containable objects (documents, custom objects, and other folders).

Security policies and security templates Security policies contain security templates which let you automatically apply security to documents, folders, and custom objects. In the case of documents, security templates can be associated with one of the several versioning states that documents pass through (Released, Superseded, In Process, or Reservation). This powerful feature provides efficient application of fine-tuned security across many objects.

Directly applied security Users who have sufficient permission can edit an object's security by directly adding or removing security principals, or by changing the existing permissions already granted.

User authentication

All logins to FileNet P8 are done through the Java™ Authentication and Authorization Service (JAAS). Authentication therefore is a process that occurs between a J2EE client application (for example, Workplace), a J2EE application server (either WebLogic, WebSphere®, or JBoss, hosting an instance of Content Engine), and one or more JAAS login modules. This process does not involve any FileNet code. Content Engine's ability to leverage JAAS for authentication means that if a single sign-on (SSO) provider writes a JAAS Login Module for a supported application server, then clients of FileNet P8 applications hosted in that application server can leverage that SSO solution. See the Authentication section for full information on FileNet P8 authentication architecture.

As a result, and unlike earlier releases of FileNet P8, the Content Engine installation process configures authentication and authorization separately even though these two configurations will often use the same information. Authorization takes place by means of a direct connection between Content Engine and one of the supported directory services; these are Windows Active Directory, Sun Java System Directory Server, Novell eDirectory, and IBM® Tivoli®. See the Directory service providers section for details. The Hardware and Software requirements document contains the supported versions of these third-party products.

Process Engine now delegates authentication tasks to Content Engine. A client of the Process Engine authenticates with the Content Engine server, and then sends the credentials obtained from the Content Engine to the Process Engine server with each request. See Process Engine Authentication for details.

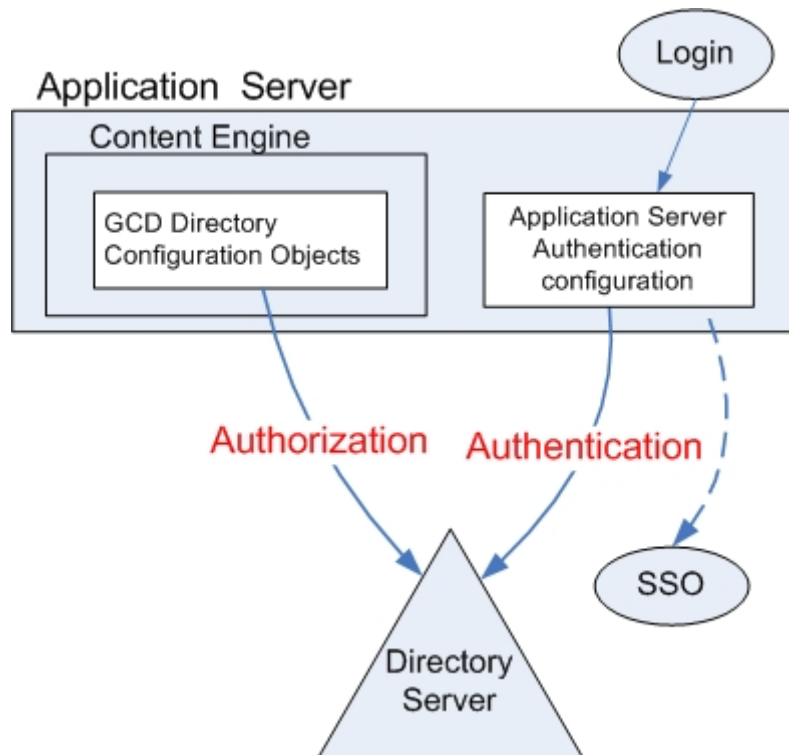
FileNet P8 authentication has been designed and extensively tested and optimized for directory servers with very large numbers of users. Optimization has particularly focused on FileNet P8's querying for accounts using pattern matching (starts with, exact match) and whether the search is optimized for short name or display name.

Configuring authentication

Two objects are required to map a directory service's naming context or "namespace" (a set of names accessible at a given node in the directory server's tree of accounts), to a FileNet P8 realm:

- On the J2EE application server, you setup **authentication** by configuring the application server's LDAP/authentication parameters that point to a naming context in one of the supported directory servers. (These could optionally point to an SSO solution.)
- Using FileNet Enterprise Manager's Directory Configuration Wizard, you then configure **authorization** by creating a directory configuration object that points to the authentication provider.

The following graphic shows the different configurations for authorization and authentication, for a single FileNet P8 domain:



FileNet P8 supports multi-realm authentication provided the application server supports it. See Configure multiple realms for instructions on how to add additional realms following initial Content Engine installation.

Security cache

For performance reasons, both Process Engine and Content Engine cache information on users, groups, and realms returned by the directory service. Cached security information may become stale as a result of changes made to the directory server. Each collection of users, groups and realms is subject to a "time to live" (TTL). From the time that a collection is first retrieved until the TTL interval has elapsed, requests for the same collection will return the cached information. Once the TTL has elapsed, the cached information will be discarded and fresh data will be obtained from the directory server.

Cache staleness may have the following effects:

- Recently added users, groups, or realms will not be visible in the relevant collections.
- Recently deleted users, groups, or realms will appear to be still present until the credentials cache refreshes.
- Updates to user or group information (for example, changed display name) will not be visible.

Staleness does not affect the ability to use a recently added user or group name in programmatic calls, nor to log in as a recently added user. Also, staleness does not compromise the security of objects. The default TTL for Process Engine security cache is four hours. Content Engine has several default TTL settings. For information on how to set the Content Engine server caches, see FileNet P8 domain properties (Server Cache tab). For information on managing the Process Engine security cache, see Manage the Process Engine user cache.

Security administrators and security tools

FileNet P8 provides several important security roles:

- GCD administrators create FileNet P8 domain resources like object stores and fixed content devices.
- Object store administrators create file storage areas, classes, folders, security policies and other object store resources. The GCD administrator is not automatically also an object store administrator, although GCD administrators can grant themselves this additional role.
- Application Engine administrator sets up Application Engine site preferences.
- Process Engine administration and configuration groups perform a variety of workflow configuration tasks.

The Users and groups topic provides full reference information.

FileNet P8 provides the following tools for configuring security:

Enterprise Manager This is the tool that system administrators will use in their daily work. Like similarly named tools provided by Microsoft® SQL Server and Oracle, Enterprise Manager gives system administrators easy access to most of the administrative and security features needed for Content Engine security configuration tasks.

Process Engine Using the Process Engine Configuration Console, the Process Engine administrator can assign access rights to workflow rosters, work queues, and user queues. You will use another Process Engine tool, the Process Task Manager, to configure how the Process Engine integrates with the directory service.

Workplace and related Application Engine applications (Application Integration, Records Manager, eForms) The security context for applications is defined and maintained by a combination of Content Engine, Process Engine, and Workplace. Consider the following examples:

- Object security (documents, folders, custom objects, events) is maintained by Content Engine.
- Workplace maintains its own configuration settings in its Site Preferences, which can be considered a security feature because they determine such things as whether a user can see certain types of documents. Workplace's site preference for Default Access Roles determines the members of such roles as Application Engine Administrators, PSConsole, PSDesigner, PWAdministrator, PWConfiguration and PWDesigner.
- Workplace also makes use of rosters, queues, etc. that are created and maintained as a part of Process Engine security.
- The Process Configuration Console is a Workplace tool.

Some security tasks can be completed by duly authorized users logged on to these applications, while others, including most of the more advanced tasks, are completed by an object store administrator running Enterprise Manager. Consult documentation provided for Workplace and the other out-of-the-box applications to see how much security-related functionality can be carried out by Application Engine administrators and advanced users.

Authentication

Authentication is the act of verifying users' identities based on credentials that those users present. This section describes FileNet P8 authentication, which operates within J2EE-established standards.

Supported authentication standards

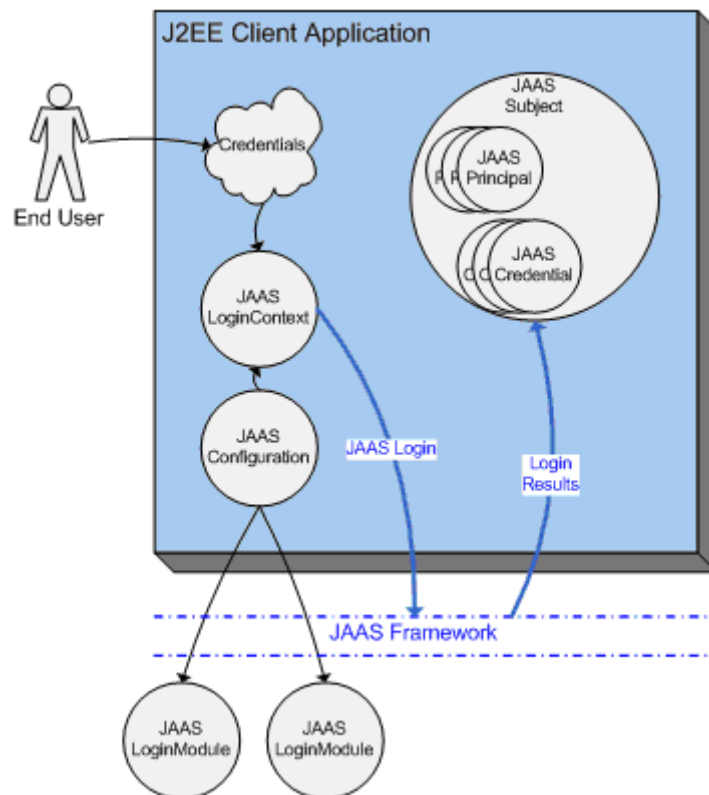
The two standards at the core of the FileNet P8 authentication process are the Java Authentication and Authorization Service (JAAS) standard and the Web Services Security standard (WS-Security). The JAAS standard forms the framework for security interoperability in a J2EE environment, while the WS-Security standard forms the framework for security interoperability in the heterogeneous environment of clients and servers that communicate through web services interfaces.

JAAS overview

Content Engine leverages JAAS for authentication (but not for authorization and it does not support the Java Security Manager). See the Directory Service Provider section for information about how Content Engine handles authorization.

JAAS provides a policy-based framework for reliably and securely determining who is invoking a Java application. Using this pluggable framework, applications like Content Engine can remain independent of the underlying authentication technologies. Likewise, third-party application server vendors, authentication providers, and single sign-on (SSO) providers can package solutions that can be leveraged by all J2EE applications and clients. Customers can plug in new or updated SSO solutions without modifying already deployed client and server applications.

The Content Engine Enterprise Java Bean (EJB) resides within the J2EE Application Server's EJB container. The Content Engine EJB is therefore accessible only by authenticated callers, those who can pass any authorization checks that the administrator has placed on the EJB.



JAAS authentication

The client application obtains a JAAS Subject prior to calling the Content Engine Java API. Interactions with the Content Engine Java API then result in the Content Engine EJB being invoked. The client's JAAS subject is transparently sent to the J2EE application server with each EJB call. The application server validates the JAAS subject, and confirms the caller's identity before executing any code in the Content Engine EJB.

NOTE For information about the authentication process used for a web service-based client, see *Web-Services-Based Client Authentication Via Ws-Security*.

JAAS configurations

To use a J2EE-based application, a client must first perform a JAAS login. To do this, the client must specify a JAAS configuration (typically through a configuration file). The JAAS configuration specifies the authentication technologies (Login Modules) that will be used to verify the client's credentials.

A JAAS configuration file lists one entry for each configured application. Within an application's entry is a list of Login Modules for that application. Based on the contents of the configuration file, the JAAS framework dynamically determines which set of authentication technologies to invoke when a client application attempts to authenticate.

Each entry in a JAAS configuration is marked as Required, Requisite, Sufficient, or Optional. The authentication process for the client succeeds only if all Login Modules marked either Required or Requisite succeed. If no Required or Requisite Login Modules succeed, then at least one Sufficient or Optional Login Module must succeed.

JAAS login contexts

A J2EE client application must specify a JAAS configuration and a mechanism through which credentials can be obtained from a user at runtime. These two items constitute a JAAS Login Context. J2EE client applications use the Login Context to interact with JAAS and to authenticate themselves to a J2EE server.

JAAS login modules

JAAS-compliant Login Modules are implemented by authentication technology providers. J2EE application server vendors, such as BEA, IBM, and JBOSS, typically provide Login Modules for standard types of credentials, such as username/password or X.509 certificate. SSO solution providers, such as CA Netegrity's SiteMinder, IBM's Tivoli Access Manager, and Oracle's CoreID, provide login modules for most of the market-leading J2EE application servers.

The JAAS Login Module is also the extensibility mechanism through which custom authentication solutions can be integrated with the FileNet P8 platform. A customer with a custom, non-standard authentication framework can integrate with FileNet P8 platform, by providing JAAS-compliant Login Modules that work with the J2EE application server that hosts the Content Engine application.

JAAS subjects

When a JAAS login has successfully completed, a JAAS Subject is returned to the caller. The Subject is a key J2EE class, which is transparently sent between J2EE clients and J2EE servers any time an EJB is invoked.

A JAAS Subject contains the set of JAAS principals (authenticated identities) and JAAS credentials (authentication data such as cryptographic keys) that result from the login process. Each Login Module that successfully authenticates the user updates the Subject with information about the user. A J2EE server application can examine the principals in the Subject to establish the caller's identity.

JAAS trust mechanisms

Trust mechanisms vary significantly from J2EE application server to J2EE application server and from Login Module to Login Module.

The JAAS Subject must either be signed by a party whom the application server trusts, or whom a configured server-side Login Module trusts (and a mechanism for configuring and verifying this trust

relationship must exist), or credentials must be included in the JAAS Subject that a server-side Login Module can verify.

Each J2EE application server vendor has invented their own proprietary trust mechanisms. This is one factor that prevents interoperability of Login Modules between different J2EE implementations.

JAAS login module interoperability

JAAS Login Modules are generally not interoperable between different J2EE application server vendors. For this reason, FileNet P8 platform supports only homogeneous application server configurations: all Content Engine servers, as well as all Application Engine servers, or other J2EE presentation tier servers, must use the same J2EE application server as a host.

Using JAAS from a stand-alone Java client

For applications that reside in a J2EE container, the containers provide a set of out-of-the-box JAAS integration capabilities that ease some of the interoperability issues discussed above.

For thick-client Java applications, however, the application must create a JAAS CallbackHandler to pass its credentials into a JAAS LoginContext. If the application relies on a username/password paradigm, then the mechanism for doing this is well defined, and interoperability options are well known. For any other type of credential, the standard is not clear on what type of callbacks the application must use, nor how those callbacks should respond. In all cases, a stand-alone Java client must use Login Modules compatible with the J2EE application server it wishes to call.

Once a JAAS Subject has been obtained by the application, there are differences in how it indicates to the Java environment that the obtained Subject should be used when executing a given set of code, such as invoking an EJB. This is an area where the J2EE application server vendors had to improvise, and, as a result, different runtime calls are required for accessing different J2EE application servers.

WS-Security overview

A web service is an XML-based interface to a system that is in conformance with the key web services standards. Through standards compliance, web services enable the development of service-oriented architectures, and allow heterogeneous systems to discover one another and interact. FileNet has developed web service interfaces to the FileNet P8 Process Engine and Content Engine services, and supports the use of Business Process Execution Language (BPEL)-compliant web services within a business process.

One of the key standards that defines a web service is the WS-Security standard, developed by the OASIS standards body. WS-Security defines three main security mechanisms for web services: security token propagation, message integrity, and message confidentiality. In this topic, we are only concerned with the first of these: security token propagation. WS-Security provides profiles that define how different types of security credentials are formatted and inserted into a web service message.

Like JAAS, WS-Security is an extensible standard which supports multiple security token formats. The WS-Security specification describes how to encode a set of standard tokens as well as defining a general mechanism for encoding any binary token. However, the specifics of the standard tokens (the actual XML elements and attributes) are not defined in the WS-Security specification. They are defined in separate profiles, including the Username Token Profile, the X.509 Certificate Token Profile, the Security Assertion Markup Language (SAML) profile, the Kerberos profile, etc. It is these profiles that ensure interoperability between different implementations using the same token type. Later subtopics will describe how these profile types are used, or might be used, in the context of FileNet P8.

Unlike JAAS, WS-Security does not provide an execution environment that defines how to configure authentication on the client and server, or how a client can become authenticated, or how a security provider can implement and package a standards-compliant Login Module. WS-Security defines only the encoding of security tokens within the SOAP XML header. The sender of the token can perform an actual authentication, and send some proof of identity with a reference to a security authority who vouches for that identity (as in the Kerberos and X.509 certificate cases), or the sender can send raw credentials,

which must be verified by the server (as in the username token case). The receiver of the token can process it in whatever way it sees fit.

When the Content Engine server receives a web service request, the Content Engine web service listener extracts the WS-Security header and performs a JAAS login based on its contents. If this JAAS login is successful, then the web service listener passes the request onto the Content Engine EJB layer within the EJB container.

FileNet P8 Platform server authentication architecture

There are two primary servers in the FileNet P8 Platform:

- Content Engine, which provides Content Management capabilities
- Process Engine, which provides Business Process Management capabilities.

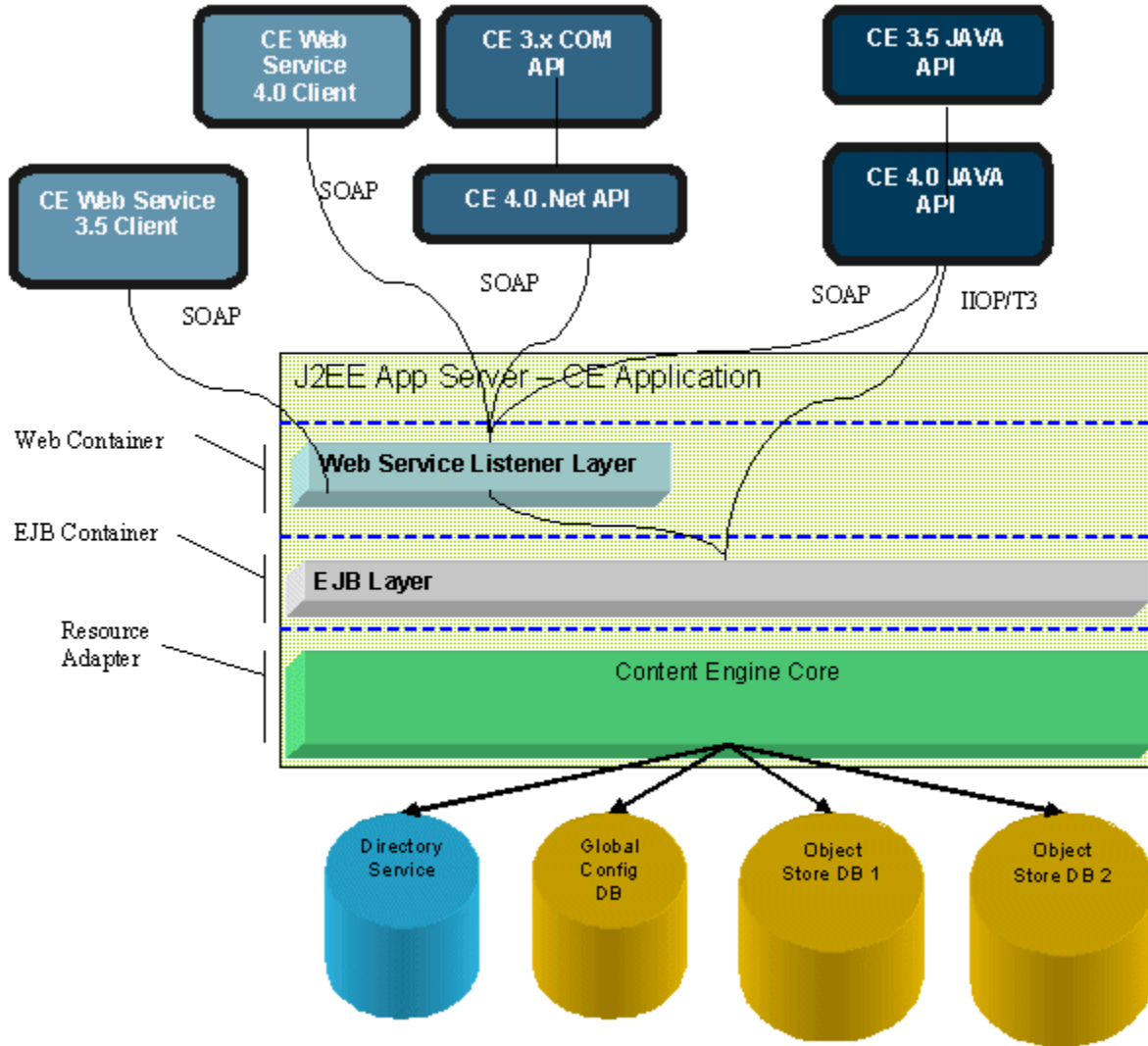
The following subtopics describe how these servers perform authentication.

Content Engine authentication architecture

The next figure shows a high-level view of a FileNet P8 Content Engine server and some of the types of client applications that access it. Content Engine is packaged as a J2EE application, deployed on one or more J2EE application server instances. The key components of this application are:

- The Content Engine Web Service listener. This listener is packaged as a servlet-based application that resides in the web container of the application server. The listener implements the Content Engine web service and supports FileNet P8 4.0 web service clients, as well as legacy FileNet P8 3.5 web service clients. The listener exposes the full functionality of the Content Engine server through a standard web services API. Requests that arrive at this web service are authenticated based on the credentials in their WS-Security headers, and then passed on to the Content Engine EJB layer.
- The Content Engine EJBs. These J2EE session beans reside in the EJB tier of the application server and implement the same web services API as the CE Web Service, exposing them through an Enterprise Java Bean interface, rather than a web services interface. All clients of this EJB layer must perform a JAAS login prior to sending a request to one of the EJBs.
- The core content management logic resides in the resource adapter tier of the application server.

The Web Service Listener and EJB layers are referred to as the two transport layers of Content Engine. All client requests enter CE through one of these two transport layers.



CE and some types of client applications.

Process Engine authentication architecture

PE delegates authentication operations to CE. This arrangement is discussed in further detail in Process Engine Authentication.

JAVA-based client authentication (JAAS)

Content Engine accepts requests either through a standard J2EE EJB or through the Content Engine (CE) web service. Clients of the CE Java API have the option of choosing to use either of these transport layers.

One of the advantages inherent to the EJB transport layer is the ability to leverage JAAS-based authentication. The JAAS standard provides a mechanism that allows third-party authentication or single sign-on (SSO) solutions to be integrated with any J2EE-compliant application such as Content Engine. If an SSO provider writes a JAAS Login Module for a given application server, then clients of applications hosted in that application server can leverage that SSO solution.

The way that JAAS-based authentication is used can differ in different client and server scenarios. The sections below illustrate several such scenarios.

Browser-based clients of J2EE application servers

Browser-based clients of J2EE-based application servers interact with servlets and Java Server Pages (JSPs). To access a FileNet P8 server, J2EE servlet-based applications must obtain a JAAS Subject that is valid in the J2EE EJB container that hosts the Content Engine EJB. There are two paradigms that may be used to obtain this JAAS Subject: application-managed authentication and container-managed authentication.

Application-managed authentication

A servlet may make JAAS calls to perform its own JAAS login programmatically (see JAAS Overview). This approach may involve application-server-specific idiosyncrasies and configuration issues, but it is fairly standardized for the common username/password case. To use application-managed authentication, each servlet deployed in the container must include logic that determines whether the user is authenticated. If the caller is authenticated, then its identity is determined by examining information in the user's session. Each servlet must perform some action (such as redirecting an unauthenticated user to a logon page) that collects and verifies the user's credentials, handles errors, and manages the encoding of the authenticated identity into the user's session.

Container-managed authentication

One of a standard set of servlet authentication options may be specified in the web application's deployment descriptor. In this case, the servlet container performs the JAAS login, based on the credentials that are supplied, relieving the application of this burden.

Each web application deployed in a servlet container can specify one of the following options, as defined in the servlet specification, that the container should use to authenticate users:

- **HTTP Basic Authentication (BASIC):** The container requests the client browser to prompt the user for a username and password, using a browser-specific dialog box. This option is defined by the HTTP specification. The user-supplied credentials are sent to the server for authentication. A secure transport must be used, as credentials are unencrypted. Few applications use this method, since the application's look and feel are not preserved, and the login prompt cannot be integrated into the application in a cohesive presentation.
- **HTTP Digest Authentication (DIGEST):** Like HTTP Basic, but a digest (a one-way hash) of the password is sent instead of the password. This option is somewhat more secure, as the actual password is not compromised, but requires that the authentication mechanism that manages identities accept the password hash in a particular form, rather than the actual password. This is not possible in most real-world cases.
- **Forms-Based Authentication (FORM):** Instead of asking the client browser to prompt for username/password, the caller is redirected to an application-specific form to provide this information. This option allows customization of the look and feel of the login page and any error

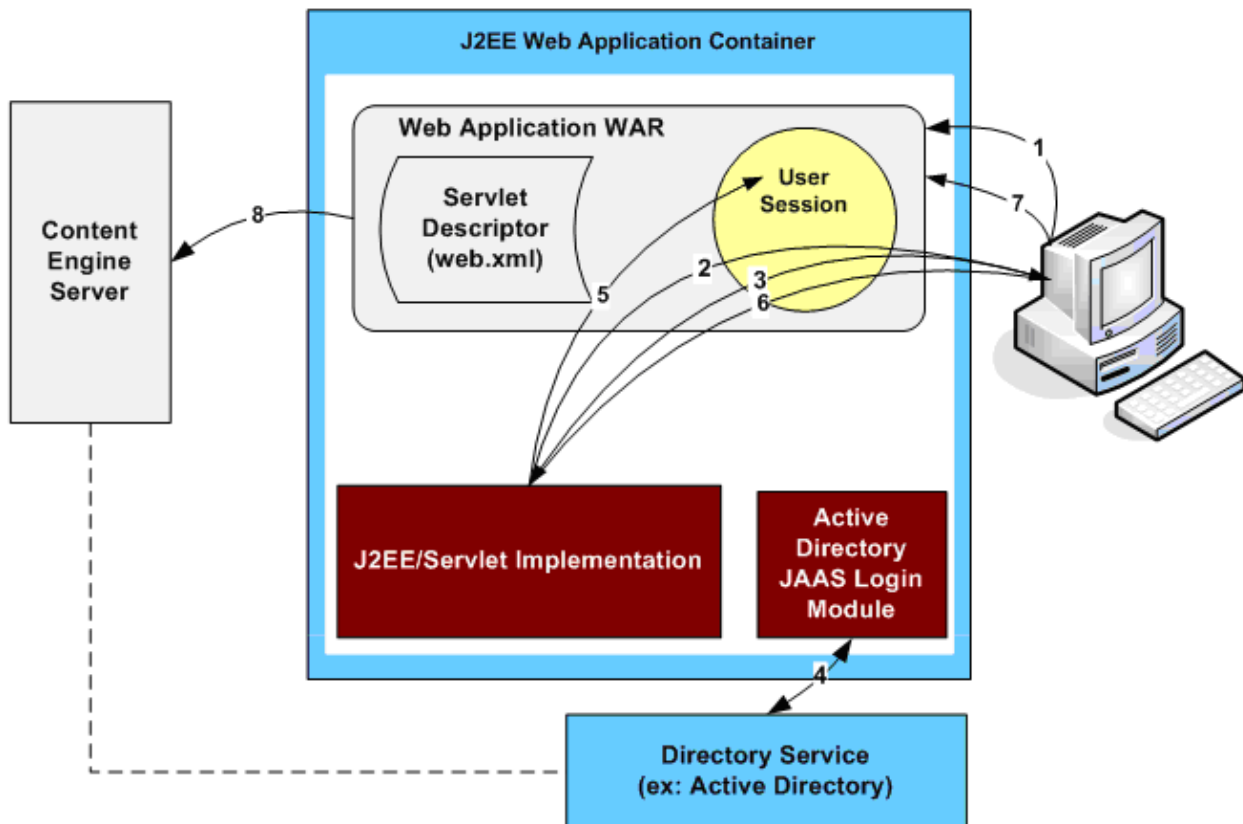
pages. It also requires a secure transport to protect the password. This is Application Engine's out of the box default (that is, for Workplace sign-in).

- **HTTPS Client Authentication (CLIENT-CERT):** This option requires each user to have a unique Public Key Certificate (PKC), and requires the use of an HTTPS (SSL) connection between the client and the server.

Note that while all four of the options above may be executed over an HTTPS connection (and, in fact that is a recommended best practice), only CLIENT-CERT actually requires an HTTPS connection. SSL is engaged through the configuration in the servlet descriptor of <transport-guarantee> as CONFIDENTIAL or INTEGRAL.

All of these technologies are forms of container-managed authentication, where the J2EE servlet container performs the JAAS authentication based on credentials obtained by a standard mechanism. The specification of one of these authentication mechanisms is a standard part of a servlet deployment descriptor. The specification and configuration of how the J2EE application server validates these credentials, however, is application-server-dependent. In an enterprise environment, an authentication mechanism must be provided to validate credentials against the enterprise identity management solution (either a directory service or SSO solution).

Once a caller has been authenticated by a J2EE servlet container, if the servlet subsequently calls an EJB, the Servlet Container is required to propagate the caller's identity (JAAS Subject) to the EJB. The diagram below illustrates the container-managed authentication case, using forms-based authentication to authenticate the caller against Active Directory:



Container-managed authentication, using a forms-based authentication option.

The following steps occur in this graphic:

1. A user attempts to access a servlet-based application.
2. The J2EE application server redirects the user to a page that challenges for credentials.

3. The user enters credentials and submits them to the server.
4. The J2EE server validates the user's credentials via JAAS.
5. The J2EE server creates JAAS Principal and Subject objects using the Active Directory JAAS Login Module, and places them in the user's session.
6. The J2EE server redirects the user back to the application page that was originally requested.
7. The servlet container looks for a user principal value available on the incoming request.
8. Once invoked, the servlet makes a call to Content Engine, and the user's JAAS Subject is propagated to Content Engine's EJB container.

Perimeter authentication

In perimeter mode, the authentication process occurs outside of the web container. An entity outside of the application server collects the users' credentials, validates them through proprietary mechanisms, and sends them onto the server in the form of an HTTP cookie. This is the mechanism used by SSO solutions to integrate with a J2EE web application. Several examples of this mode are discussed in Single sign-on via JAAS.

The basic pattern is that a third-party proxy server intercepts the web server requests and authenticates them using proprietary technology. A proprietary HTTP header is then added to the request (an SM-Session token in the Netegrity case, or an LTPA token in the Tivoli Access Manager case). When the request arrives at the web server, the servlet container intercepts it, detects that it contains an SSO cookie, extracts the cookie, and invokes the SSO provider logic to perform a JAAS login, using SSO specific Login Modules, converting the contents of the cookie into a valid JAAS subject.

Perimeter authentication is considered as a form of container-managed authentication, even though authentication occurs outside of the container. Perimeter authentication is configured by selecting CLIENT-CERT as the container-managed authentication mechanism, and then performing some additional SSO provider-specific steps. The web container extracts whatever data was present in the CLIENT-CERT cookie of the incoming request and passes it into the JAAS Login Modules that are configured for the web application. A JAAS Login Module for a third-party SSO vendor can pass a proprietary token in this CLIENT-CERT field and then process that token in a Login Module the vendor provides.

Integration with Kerberos-based authentication environments via the Simple and Protected GSS API Negotiation Mechanism (SPNEGO) standard is one type of perimeter authentication used by many J2EE web container implementations.

J2EE thick Java client

As discussed in Using JAAS from a stand-alone Java client, additional challenges exist when using JAAS in a stand-alone Java client environment. In particular, integration options with third-party SSO providers may be limited or unavailable. In many cases, third-party JAAS Login Modules may be provided that work well when executing within a J2EE container, but cease to work when running in a stand-alone Java environment. The stand-alone Java environment may not support the application-server-specific trust mechanisms needed to produce valid JAAS subjects for the target application server.

Login Modules that allow the use of username/password credentials from a stand-alone client are available from the individual application server vendors. In some cases, a Login Module that allows the use of PKI certificates with two-way SSL may also be available. Support for other authentication options in a thick Java client environment will most likely require a custom integration. Clients of the FileNet P8 Content Engine Java API may use a JAAS Subject that they have obtained themselves, or pass in username/password credentials to the API, which will then attempt to obtain a JAAS Subject for them, using Login Modules specified in the operative JAAS configuration.

Support for Content Engine 3.5 Java API clients

Clients using the Java Compatibility Layer may continue to provide username/password credentials as for the Content Engine 3.5.x Java API. In this case, the Java Compatibility Layer will perform a JAAS login on each user's behalf, using the Login Modules specified in the operative JAAS configuration. Alternatively, a client may perform its own JAAS Login prior to calling the Java Compatibility Layer. New configuration options are provided in the Java Compatibility Layer configuration file for controlling these authentication options.

Support for Java applets

A Java applet is a utility that conforms to the interfaces defined in the `java.applet` package and is designed to run as an embedded part of another application, typically a web browser. An applet will be launched in a separate window, and may execute independently from the parent application that launched it. Applets are typically created to host graphically complex interactive user interfaces that would be difficult to implement in a browser.

In the common case where an authenticated client running in a browser takes some action that causes an applet to be launched, the applet may be able to inherit the parent's credentials. If the applet communicates via HTTP to servlets in the same J2EE web container that the parent browser was communicating with, then authentication credentials that are stored in HTTP headers received by the parent should be accessible to the applet, and the applet should not require further authentication.

If, however, the applet uses other APIs to connect to the server, or if the applet communicates with different servers than the parent application, then propagating the caller's identity may require custom development, or a second authentication may be required for the applet.

Applets and reverse proxy servers

The Perimeter authentication section above discussed the use of reverse proxy servers in perimeter authentication scenarios. The reverse proxy server acts as an intermediary between the browser-based client and the HTTP server. It appears to the browser that the reverse proxy is the server. The browser does not know that the reverse proxy is actually translating the requests and forwarding them on to another server, and then translating the responses that are sent back.

To make it appear that the reverse proxy is the actual server, the reverse proxy must translate the responses from the server so that any references to the target server are converted to references to the reverse proxy server. This primarily affects embedded URL's. Web pages often contain many links to other pages hosted in the same web site. The reverse proxy must translate these references so that they appear to reference the same resource, when in fact they are hosted on the reverse proxy server rather than the target server.

Reverse proxy servers are designed to handle traffic from browser-based clients, and typically translate any references that appear in HTML data correctly. There are a number of special considerations that come into play when applet traffic is sent through a reverse proxy server:

- **Handling Cookies:** Cookies are data sent between a client and a server via HTTP headers. In a reverse proxy scenario, the reverse proxy may interject its own cookies into the data sent to the server, as well as adding to and translating the set of cookies that are returned from the server to the client. The applet must not cache cookies in memory; it must obtain them from the HTTP connection every time it needs to access them so that it can obtain the latest set of translated cookies sent by the reverse proxy.
- **Handling Redirects:** If a reverse proxy detects stale or invalid authentication tokens in the cookies sent from a client, it may use an HTTP re-direct to cause the client to refresh its cookies. The applet must handle HTTP re-directs properly.
- **Translating URL's embedded in XML or other non-HTML data:** Applets typically send non-HTML data back and forth with the server. If the data returned from the server contains embedded URL's, then the reverse proxy server must translate these. Causing this to happen may require

changes in the proxy server, to have it translate data in fields which it would not normally examine.

Single sign-on integrations via JAAS

The implementation of the FileNet P8 Content Engine (CE) server as a J2EE application allows it to take advantage of integrations between the J2EE application-server vendors (such as IBM and BEA) and the leading single sign-on (SSO) solution providers such as IBM's Tivoli Access Manager and CA/Netegrity's SiteMinder. If an SSO vendor has provided JAAS Login Modules that work with a given application server, FileNet P8 can support that single sign-on solution for Java-based applications.

Note the following limitations to integrating with SSO providers:

- JAAS-based SSO integration is available only to J2EE-based clients (clients who are able to perform a JAAS login). This type of client includes most of those that are browser-based and work with a J2EE-based presentation-tier server.
- JAAS-based SSO integration generally does not extend to .NET-based clients or pure web-services-based clients.
- JAAS-based SSO integration is generally available only if the client's J2EE environment is supplied by the same application-server vendor hosting the Content Engine application.
- The configuration of SSO solutions generally requires a high level of product-specific expertise. Before installing a FileNet P8 solution, customers must work with their SSO and J2EE application server vendors to correctly configure the single sign-on environment. The industry-leading SSO products have been designed to be highly flexible, supporting a wide range of credential types and configuration options. FileNet is only able to test a limited number of the infinite possible combinations of SSO vendor product configurations and application servers.
- In all cases, the third-party SSO product must be configured to work with the same underlying directory service (for example, Microsoft Active Directory and Novell e-Directory) used to resolve user and group credentials within FileNet P8.

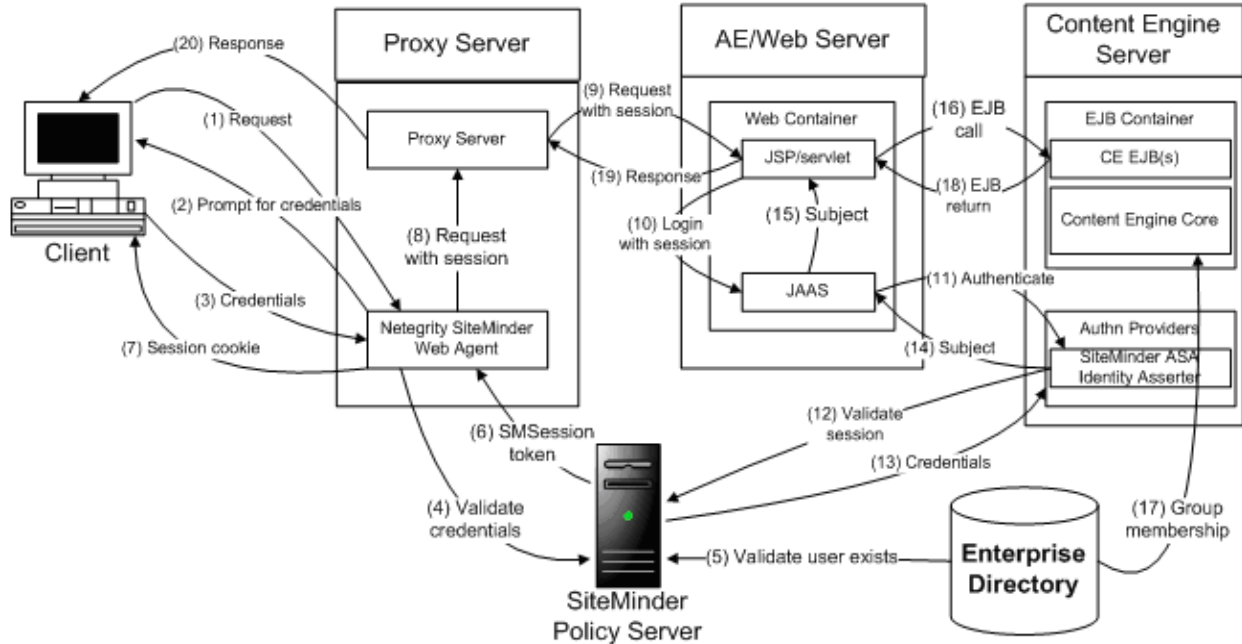
The topics below discuss two leading single sign-on products with which FileNet has done proof-of-concept integrations. Each of these products has configuration manuals that are literally thousands of pages in length, supporting many different credential types and authentication scenarios. The configurations presented below are intended to represent mere samples of what can be done.

CA/Netegrity SiteMinder Web Agent

The example presented in this topic focuses on one common Netegrity integration scenario. A browser-based client can authenticate through SiteMinder when HTTP requests are intercepted by a reverse proxy server running Netegrity's Site Minder Web Agent. In this scenario, a presentation layer application is hosted in a J2EE servlet or JSP environment. The clients think that they are accessing this application directly, but network access has actually been configured to flow through a reverse proxy server, which performs the authentication using a custom logon page or some other mechanism.

Once the client is authenticated, all further requests will carry a SiteMinder SMSession token as an HTTP cookie. A SiteMinder Application Server Agent on the web server extracts the SMSession token and performs a JAAS login on the client's behalf. The web server may then make calls to CE. The JAAS Subject is propagated with each call. The diagram below illustrates the steps that occur in this scenario.

NOTE This example uses Netegrity products (for example, SiteMinder Application Server Agent Identity Asserter) which are specific to the WebLogic platform. Different components and product names would apply and the steps would be different if a different J2EE application server were in use.



HTTP requests intercepted by a reverse proxy server running Netegrity.

The following steps occur in this graphic:

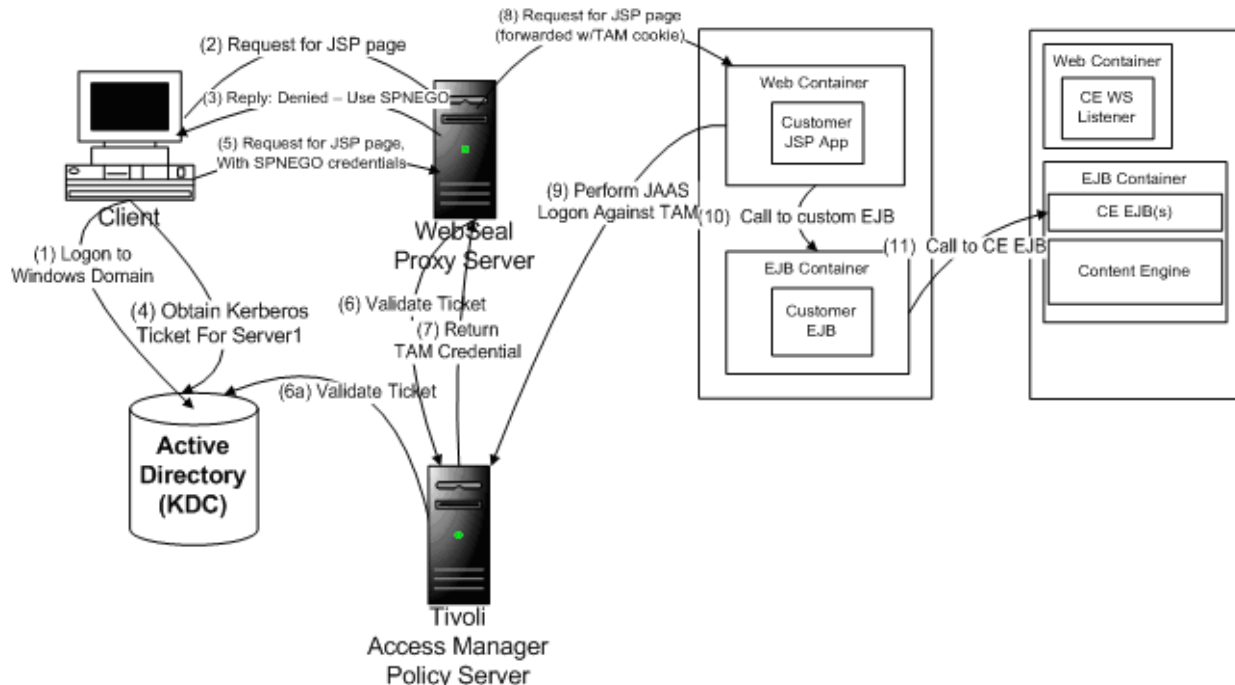
1. The user issues a request for a web page to the reverse proxy server. The first component to process this request is the SiteMinder Web Agent.
2. If the request does not contain a session token, the user is redirected to a custom logon page, and prompted for credentials.
3. The user enters credentials, which are received by the Web Agent.
4. The SiteMinder Web Agent forwards the credentials to the SiteMinder Policy Server.
5. The SiteMinder Policy Server validates the credentials by retrieving user information from the enterprise directory service.
6. After the user is authenticated, the SiteMinder Policy Server returns a session token to the SiteMinder Web Agent.
7. The SiteMinder Web Agent saves the session token as a cookie on the user's machine.
8. The SiteMinder Web Agent adds to the request an HTTP header holding the session token and forwards the request to the reverse proxy server.
9. The reverse proxy server forwards the request to the JSP/servlet running on the app server.
10. The J2EE web container hosting the JSP/servlet extracts the session token from the HTTP header and performs a JAAS login. A Netegrity JAAS Login Module is invoked.
11. JAAS authenticates the user by passing the session token to the SiteMinder ASA (Application Server Agent) Identity Asserter on the Content Engine server.
12. The SiteMinder ASA Identity Asserter forwards the session token to the SiteMinder Policy Server.
13. If the SiteMinder Policy Server determines the session token is valid, it returns user information to the SiteMinder Identity Asserter.
14. The SiteMinder Identity Asserter adds principals to a JAAS Subject and returns it to JAAS on the web server.
15. JAAS returns the Subject to the JSP/servlet.

16. The JSP/servlet calls the Content Engine EJB on the Content Engine server. The web container propagates the JAAS Subject to the EJB container.
17. To perform authorization, CE retrieves user and group information from the user directory.
18. The CE EJB returns to the JSP/servlet.
19. The JSP/servlet builds an HTTP response and returns it to the proxy server.
20. The proxy server returns the response to the client.

Note that when applets are used as a part of the client application, then the considerations in Applets and reverse proxy servers must be taken into account to ensure that the applet behaves properly in the Netegrity environment.

WebSeal from IBM Tivoli Access Manager (TAM)

In the example presented in this topic, a browser-based client that has performed a logon to a Windows domain in an Active Directory environment, uses an Internet Explorer (IE) browser to access a servlet or JSP page through an IBM WebSeal proxy. The JSP page calls CE through the Content Engine Java API, over the EJB transport. This use case describes the use of Kerberos credentials and Windows integrated logon to authenticate to WebSeal. Note, however, that any other browser-based authentication mechanism supported by WebSeal (for example, HTTP Basic or Forms-based authentication) will work just as well. The Windows Integrated Logon case below requires the use of Active Directory and the Microsoft Internet Explorer browser. Other browser-based scenarios, such as a forms-based authentication scenario, will work with other browsers and other directory services.



IE-based client in a Windows domain accesses a servlet or JSP page through an IBM WebSeal proxy.

The following steps occur in this graphic:

1. The client logs on to a Windows machine via an integrated logon. The client provides username/password credentials which are validated against Active Directory. The Microsoft Active Directory server also serves as a Kerberos Key Distribution Center (KDC). A Kerberos ticket granting ticket is issued.
2. The client uses a Microsoft Internet Explorer browser to access a JSP page.

3. Transparently to the client, the JSP request goes through a WebSeal proxy. WebSeal detects that the target web site/web page is a protected resource, and that the client is unauthenticated. WebSeal returns an HTTP Error response, with an HTTP header indicating that SPNEGO credentials are accepted. (Integrations with Kerberos-based authentication environments via the Simple and Protected GSS API Negotiation Mechanism (SPNEGO) standard are one type of perimeter authentication that is supported by many J2EE web container implementations.)
4. Internet Explorer makes Kerberos calls to the KDC on the client's behalf, to obtain a Kerberos service ticket for the target resource.
5. The Kerberos ticket is placed into an HTTP header, per the SPNEGO specification, and the original request is resent with this header.
6. The WebSeal proxy again intercepts the request. This time it makes a call to Tivoli Access Manager (TAM) to validate the client's credentials.
7. TAM validates the Kerberos ticket and authenticates the caller. A TAM credential (an LTPA token) is returned to WebSeal.
8. WebSeal modifies the original HTTP request header to include a trusted server credential together with the TAM credential of the client and forwards the request to the target server.
9. On the WebSphere server hosting the JSP page, the request is intercepted by a TAM module: a Trust Association Interceptor (TAI). The trusted server is verified, the TAM credential is extracted from the request, and a JAAS login is performed on the client's behalf, using the TAM credential. A JAAS Subject is obtained for the client.
10. The JSP page calls a custom EJB in the same J2EE application (this step is not necessary in this scenario, and could be omitted.)
11. The custom EJB calls Content Engine, through the Content Engine Java API, and over the EJB transport. The client's JAAS Subject is automatically propagated to the Content Engine server.

In this case, there is no client application code (the client is simply a browser accessing a web page), nor is there any explicit authentication logic in the customer's JSP or EJB layers. WebSeal takes care of verifying the user's Kerberos credentials and generating a TAM credential based on them. TAM authentication modules on the WebSphere application server perform the JAAS login.

For more information about Tivoli Access Manager and related products, and how they might be used in a given customer environment, contact your service representative.

Note that when applets are used as a part of the client application, then the considerations in Applets and reverse proxy servers must be taken into account, to ensure that the applet behaves properly in the WebSeal environment.

Web-Services-based client authentication via Ws-Security

The WS-Security overview describes the WS-Security standard and how it relates to FileNet P8 web services. This topic discusses how two specific WS-Security profiles, the Username Token profile and the Kerberos profile, are supported out-of-the-box by FileNet P8, and how support for additional standard profiles, as well as non-standard WS-Security compliant approaches, may be integrated with FileNet P8 web services, using the FileNet P8 Web Service Extensible Authentication Framework.

Clients of a web service must produce WS-Security compliant headers to use any of the FileNet P8 web services. Most web service-based applications are created using a toolkit, such as Microsoft's Visual Studio, which handles the creation of WS-Security compliant headers.

Username token credentials

The Web Services Security Username Token Profile (available from <http://docs.oasis-open.org>) specifies how username/password-based credentials can be passed in a WS-Security header. All web service clients that adhere to this profile should be able to interact with any web service that implements the profile. The XML `<wsse:UsernameToken>` and `<wsse:PasswordToken>` elements are defined, along with rules for how these fields must be used.

When Username tokens are sent in a WS-Security header, a secure, private channel, such as an HTTPS connection, must be used between the client and the server to prevent compromising the client's password. Two types of passwords are defined by this standard: a text password and a password digest. FileNet P8 supports only the text password option.

There are two optional elements that may be included in a Username token as counter measures against replay attacks: the nonce and creation timestamp fields.

- A nonce is a random value that the sender creates with each request. The server maintains a cache of recent nonces, and will reject any request that arrives with a nonce that has already been seen.
- The creation timestamp is used by the server to reject requests that are older than some configured time period.

Use of both of these fields is recommended, but not required, for all web service implementations.

An example of a WS-Security header containing a Username token is shown below (some of the namespace values have been truncated, for the sake of brevity):

```
<wsse:Security soap:mustUnderstand="1">
  <wsu:Timestamp wsu:Id="Timestamp-3290b465-8a4a-4e34-b5da-1e35d80d613b">
    <wsu:Created>2005-11-17T19:24:15Z</wsu:Created>
    <wsu:Expires>2005-11-17T19:29:15Z</wsu:Expires>
  </wsu:Timestamp>
  <wsse:UsernameToken xmlns:wsu="..." Id="SecurityToken-84353116-ed06-4a1a-b896-19337481c488">
    <wsse:Username> MyUsername </wsse:Username>
    <wsse:Password Type="...#PasswordText"> MyPassword </wsse:Password>
    <wsse:Nonce>QkzWRIL2COP9D4ELX4LyZQ==</wsse:Nonce>
    <wsu:Created>2005-11-17T19:24:15Z</wsu:Created>
  </wsse:UsernameToken>
</wsse:Security>
```

When a web service request containing a Username token arrives at a Content Engine (CE) web service, the web service listener extracts the credentials from the WS-Security header, and uses them to perform a JAAS login using an application-server-specific username/password Login Module. Once the JAAS login has successfully completed, the FileNet P8 web service listener is now in possession of a JAAS Subject, and can pass the call along to CE via the EJB transport.

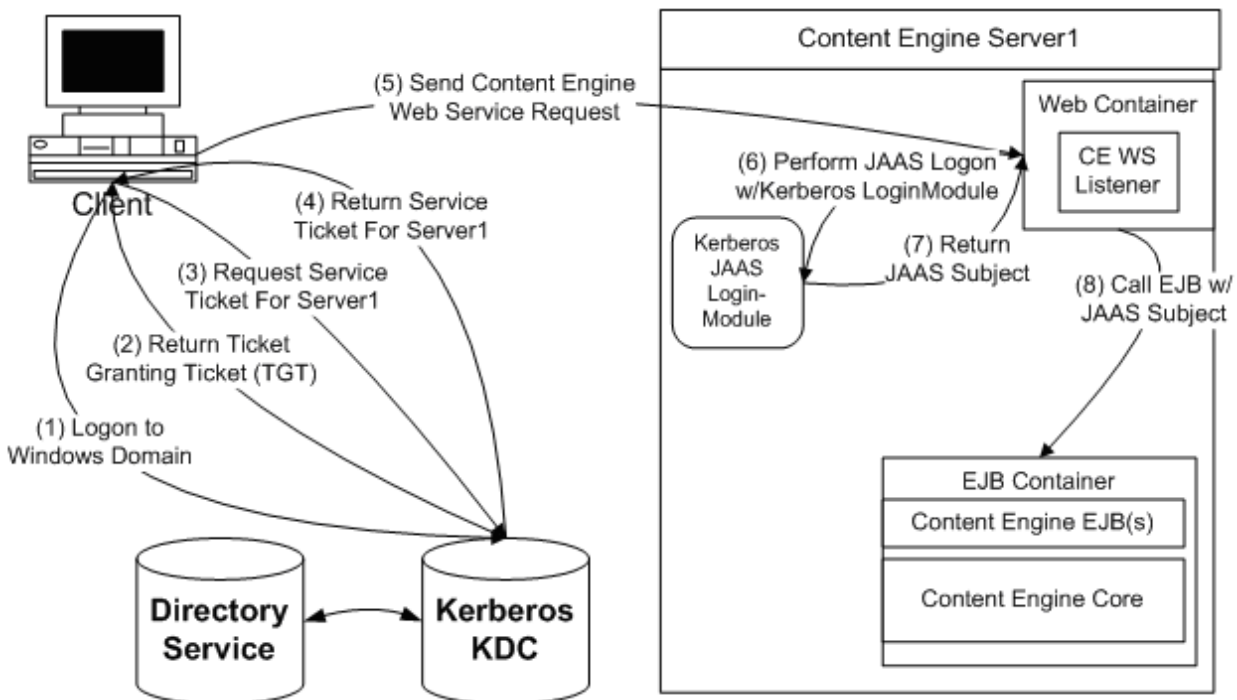
Kerberos Credentials

The Web Services Security Kerberos Token Profile (available from <http://www.oasis-open.org>) specifies how Kerberos-based credentials can be passed in a WS-Security header.

Kerberos overview

In a Kerberos environment, clients obtain "tickets" that grant them access to interact with a particular server for a particular period of time. Servers are able to verify the validity of these tickets and of the user's identity. Symmetric encryption is used to secure the tickets and keys that are exchanged in a Kerberos environment.

A Kerberos environment requires the presence of one or more Kerberos Key Distribution Centers (KDCs). A KDC generates the encryption keys and tickets that are used in the environment. The next graphic illustrates the exchanges that occur in a typical Windows Kerberos-based client/server interaction (a UNIX® Kerberos interaction would be conceptually identical):



Kerberos KDC generates encryption keys and tickets that are passed to the Web services listener.

The following steps occur in this graphic:

1. For authentication purposes, the client logs in to the Windows domain. The operating system collects the user's name and password and sends them to a KDC.
2. The KDC works with a directory service to validate the user's credentials and returns a ticket-granting ticket (TGT) to the caller.
3. Because the client wishes to access a particular server, a ticket-granting exchange must occur. The client sends a second request to the KDC, specifying the service that it wishes to access. (Note that if the client had previously been granted a service ticket and it has not yet expired, then steps 3 and 4 would be skipped).
4. The KDC issues a service ticket, which grants the caller access to that service.
5. The client sends its request with the service ticket encoded in a WS-Security Kerberos token to the server.

6. The web service listener on the Content Engine server performs a JAAS login using credentials supplied in the incoming WS-Security header.
7. The JAAS LoginModule validates the client's ticket. No roundtrip to the KDC is necessary. The server is able to use its own key (obtained from the KDC at startup time) to validate service tickets. The LoginModule produces a valid JAAS subject, based on the successful validation of the ticket. This JAAS subject is returned to the Content Engine web service listener.
8. The Content Engine web service listener passes the call along (with valid JAAS subject) to the Content Engine EJB layer.

When Kerberos tokens are sent in a WS-Security header, a secure, private channel, such as an HTTPS (SSL) connection, is not required between the client and the server, as the tickets sent in a Kerberos request are already encrypted. (Customers may wish to use an HTTPS connection anyway to ensure privacy for information contained in the request and response bodies.)

Microsoft's use of Kerberos

Starting with Windows 2000, Microsoft embraced the Kerberos standard, and based the Windows authentication process on it. A Windows domain controller acts as a Kerberos KDC, and a Windows domain is equivalent to a Kerberos realm. Microsoft's use of Kerberos to authenticate Windows clients improved Windows scalability, and allowed interoperability with other non-Windows clients.

However, Microsoft's Kerberos implementation is not entirely compatible with the MIT implementations. Microsoft has made minor changes to the protocol that have introduced compatibility issues. With some work and some caveats, however, Windows and UNIX Kerberos clients can be made to interact.

Kerberos forms the basis for the "Windows Integrated Logon" capability that allows a Windows client to logon to the network and access network resources across the Windows domain (and in some cases, in other Windows domains).

Microsoft provides a web services development toolkit called Web Services Extensions (WSE) for creating web service-enabled client and server applications. WSE supports the use of Kerberos credentials, obtained from the client's environment, to generate WS-Security-compliant Kerberos tickets. FileNet P8 supports the use of these tickets in FileNet P8 web services, allowing Windows clients who have logged onto the domain to access FileNet P8 web services without providing any additional credentials.

Through use of Kerberos tokens, web service client applications can be developed allowing single sign-on with FileNet P8 web services in a Windows Integrated Logon environment.

Sample Kerberos token

Kerberos tokens are encoded in a WS-Security header using the standard `<wsse:BinarySecurityToken>` element. The Kerberos Token Profile defines how each of the attributes within this token must be set when sending Kerberos tokens. An example of a WS-Security header containing a Kerberos token is shown below (some of the namespace values have been truncated):

```
<wsse:Security soap:mustUnderstand="1">
  <wsu:Timestamp wsu:Id="Timestamp-a475236f-c2c8-4c37-a280-f6ae043b09dd">
    <wsu:Created>2005-04-09T00:17:53Z</wsu:Created>
    <wsu:Expires>2005-04-09T00:22:53Z</wsu:Expires>
  </wsu:Timestamp>
  <wsse:BinarySecurityToken
    ValueType="wsse:Kerberosv5ST"
    EncodingType="wsse:Base64Binary"
    xmlns:wsu="http://.../oasis-200401-wss-wssecurity-utility-1.0.xsd"
    wsu:Id="SecurityToken-3b4344fc-374f-4e71-9d1c-e6a356a62f6b">YYID ... pPZt
  </wsse:BinarySecurityToken>
</wsse:Security>
```

(For information on the ValueType and EncodingType values, see the WS-Security specification.)

When a web service request containing a Kerberos token arrives at a FileNet P8 4.0 CE web service, the web service listener extracts the token from the WS-Security header, and uses it to perform a JAAS login

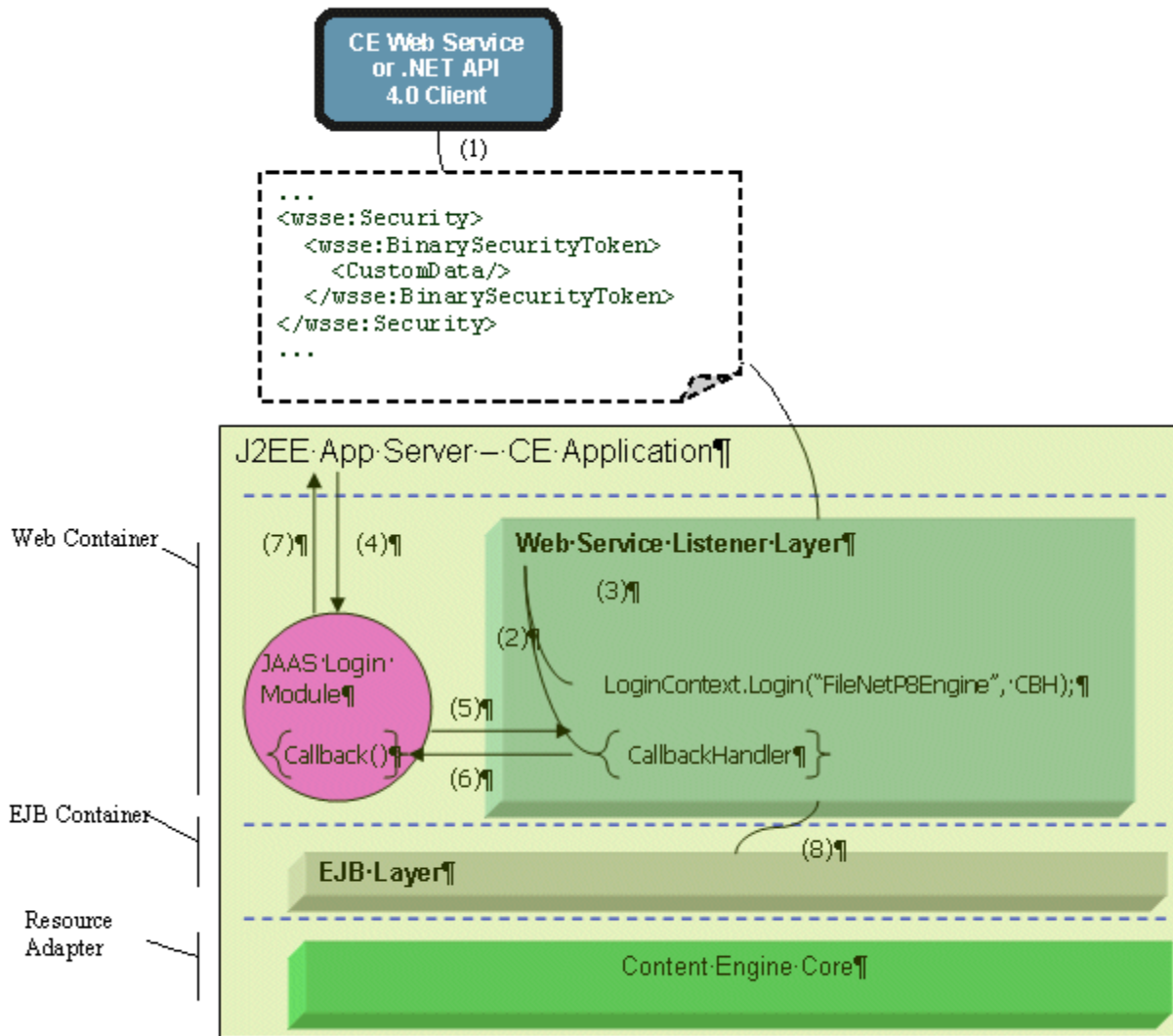
using a FileNet-provided, application server specific Kerberos Login Module. Once the JAAS login has completed successfully, the FileNet P8 web service listener is now in possession of a JAAS Subject, and can pass the call along to CE via the EJB transport.

See also Kerberos for Content Engine for more details on using Kerberos.

Web Service extensible authentication framework

The previous two sections discussed how the standard Username and Kerberos token profiles can be used to authenticate against FileNet P8 web services. These are the only two credentials types supported out-of-the-box in FileNet P8 for use over the web services transport.

For clients that must use the web service transport but cannot make use of one of the two out-of-the-box token types, FileNet has provided a Web Services Extensible Authentication Framework (WS-EAF). This framework consists simply of a set of conventions for writing a JAAS Login Module that is able to interact with the FileNet P8 Content Engine web service listener to obtain the credentials that are present in the WS-Security header of an incoming request packet. The next figure illustrates this interaction:



Web Services Extensible Authentication Framework.

The following steps occur in this graphic:

1. A Content Engine web service client sends a request containing a custom set of credentials packaged in a WS-Security header.

2. The request arrives at the CE web service listener. The web service listener extracts the WS-Security headers and examines them. It sees that they do not contain one of the out-of-the-box FileNet P8 credential types, so it invokes the FileNet P8 WS-EAF authentication mechanism. A JAAS CallbackHandler is created and seeded with the contents of the WS-Security header.
3. A JAAS login is performed, specifying the FileNetP8Engine JAAS configuration and the CallbackHandler created in the previous step.
4. The standard JAAS runtime looks up the login modules that are listed in the JAAS configuration file for the FileNetP8Engine stanza and invokes each of the listed Login Modules, passing in the CallbackHandler as a parameter.
5. The custom WS-EAF JAAS Login Module instantiates one or more standard JAAS callbacks and passes these callbacks to the CallbackHandler's handle() method.
6. For each callback that the client has requested, the Content Engine web service CallbackHandler supplies the callback with requested XML fragments from the incoming WS-Security header, such that they can be retrieved by the custom WS-EAF JAAS Login Module.
7. The Login Module is now in possession of the WS-Security header information, and is able to use this information to perform its proprietary authentication process. If the authentication is successful, then a JAAS Subject is populated and returned.
8. The Content Engine web service listener now has a valid JAAS Subject, and can call the Content Engine web service to handle the request, via the Content Engine EJB.

SAML credentials

The Security Assertion Markup Language (SAML) is a web services specification that defines how to encode security assertions in XML. FileNet P8 does not provide any explicit support for SAML tokens. However, the Web Services Extensible Authentication Framework defined in the previous topic can be used to build support for authenticating via SAML-based credentials within FileNet P8 applications.

Java-based clients over the web service transport

Java clients that use the web service transport are able to use only Username tokens. Neither Kerberos nor Web Service Extensible Authentication Framework support is available to Java-based clients over the web service transport.

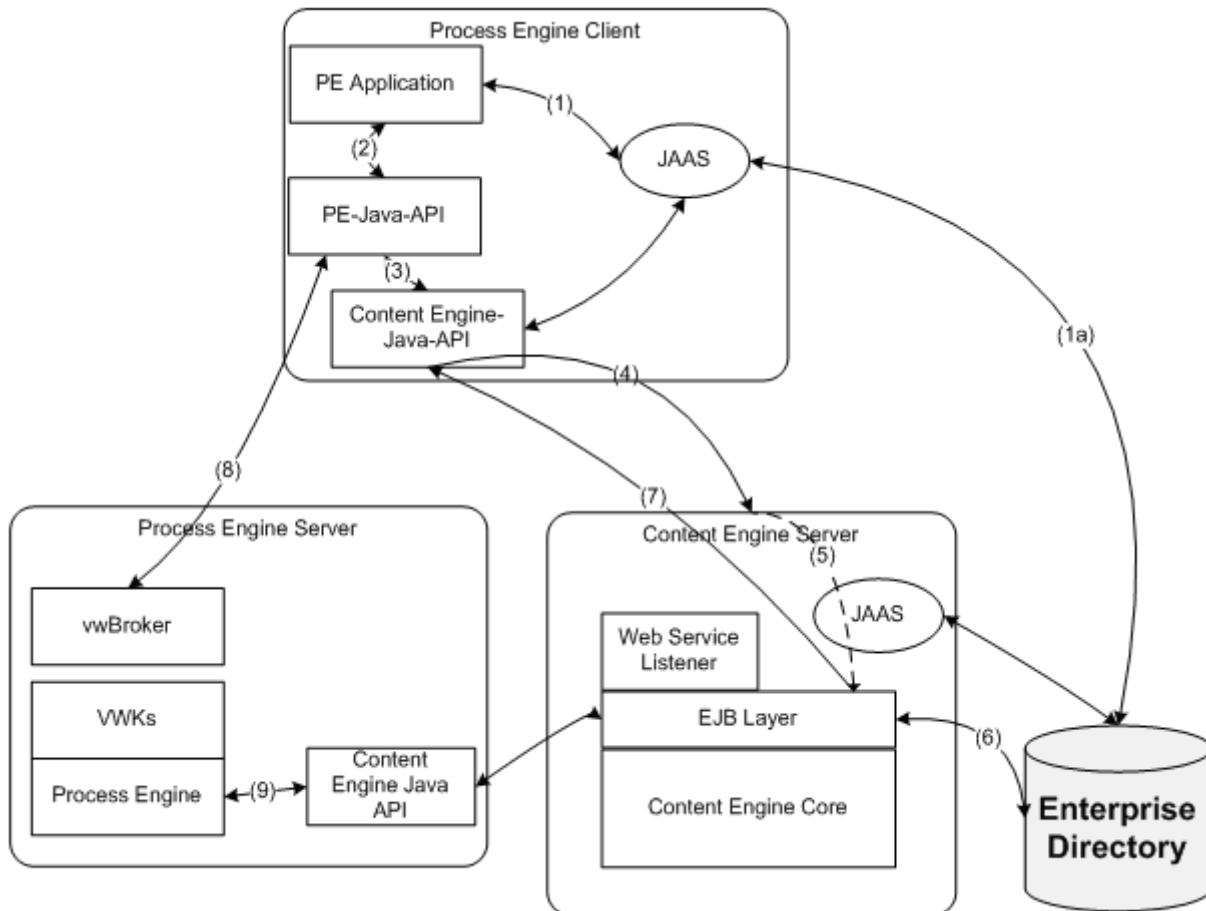
Process Engine authentication

In a change from previous releases, Process Engine relies on Content Engine for authentication and directory service access operations. The following sections describe the steps that occur as a client of Process Engine authenticates with Content Engine, and then sends the credentials obtained from Content Engine to Process Engine with each request.

Authenticating the Process Engine Java API client using Content Engine EJB Transport

The next figure shows a Process Engine Java API client in a FileNet P8 server environment. Note that both the Process Engine Java API and the Content Engine Java API must be present on the client machine (the Content Engine setup program installs both the Process Engine Java API and the Content Engine Java API components). Note also that the figure assumes that the Content Engine Java API is configured to work over the EJB transport. See the following section for a discussion of how this will differ if the Content Engine web service transport is in use.

This figure represents a two-level authentication process, where the client first authenticates with JAAS. The client then authenticates with Content Engine, with the application server validating the JAAS credentials by invoking one of the configured Login Modules to confirm that the client's credentials are of a type accepted by this server and that they are valid. This establishes the caller's identity, and provides access to FileNet P8.



Process Engine client first authenticates with JAAS on the app server and then with the Content Engine server.

The following steps occur as the Process Engine client application initializes and then makes a call to the Process Engine server:

1. The Process Engine client performs a JAAS login, using the Login Module configured in the client environment. This login probably occurs before any interaction occurs with the Content Engine or Process Engine APIs. It may happen automatically as a result of a J2EE configuration, or the client may invoke the Login Module programmatically.

Depending on what Login Module is in use, the Login Module may make a call to the enterprise directory service, possibly through a proxy or some intermediate SSO solution.

NOTE In the case of legacy applications (pre-4.0.0) which pass the user name and password to the Process Engine API, the API will perform a JAAS login using the FileNet P8 JAAS Login Context.

2. The Process Engine Application makes a call to the Process Engine Java API.
3. The Process Engine Java API sees that the client has not been authenticated to FileNet P8 yet, and therefore makes a call to the Content Engine Java API to obtain a FileNet P8 identity token.
4. The Content Engine Java API makes a call to the server. At the Content Engine server, the call arrives at the J2EE application server's EJB container with the caller's JAAS Subject. The application server examines its security policy configuration, as well as how the Content Engine EJB is configured, to determine what security policy is in place. The application server determines whether the JAAS Subject associated with the incoming request matches one of the authentication providers that are configured and enabled for use with the EJB. If it does, the application server makes a call to this authentication provider which performs any necessary checks on the JAAS Subject (this might involve contacting a directory service or SSO provider to validate the Subject). If any part of this application server authentication process fails, an appropriate error is returned to the caller (note that this would occur before any logic within the Content Engine EJB is invoked).
5. If the incoming JAAS Subject is validated by the application server, then the call is passed through to the Content Engine EJB, and the JAAS Subject is available to the Content Engine EJB.
6. Within the Content Engine EJB, the Principal name is extracted from the JAAS Subject, and searched for in the Content Engine user cache. If not found, Content Engine calls the enterprise directory service to expand the caller's identity information.
7. The EJB processes the incoming request. In this case, a FileNet P8 identity token is created, and returned to the caller (the Process Engine Java API).
8. If all the above steps are successful, the Process Engine Java API now has a FileNet P8 identity token, obtained from a Content Engine server. The Process Engine Java API makes a call to the Process Engine server, passing the identity token as a parameter. The Process Engine server examines the identity token, and the signature on it is validated. If the signature is valid, then the Process Engine server trusts that the token and the user identified by the token are valid.
9. This optional step is unrelated to the authentication process, but will occur in many cases. In this step, Process Engine wishes to retrieve the detailed user information (full name, DN, e-mail address, group memberships, etc.). It does so by calling Content Engine to retrieve the requested user and group objects.

The FileNet P8 identity token used to convey the identity of the caller from Content Engine back to Process Engine is protected by a pair of cryptographic keys. One of these keys is a secret shared between the Process Engine server and the Content Engine server, while another key is dynamically generated for use with each token.

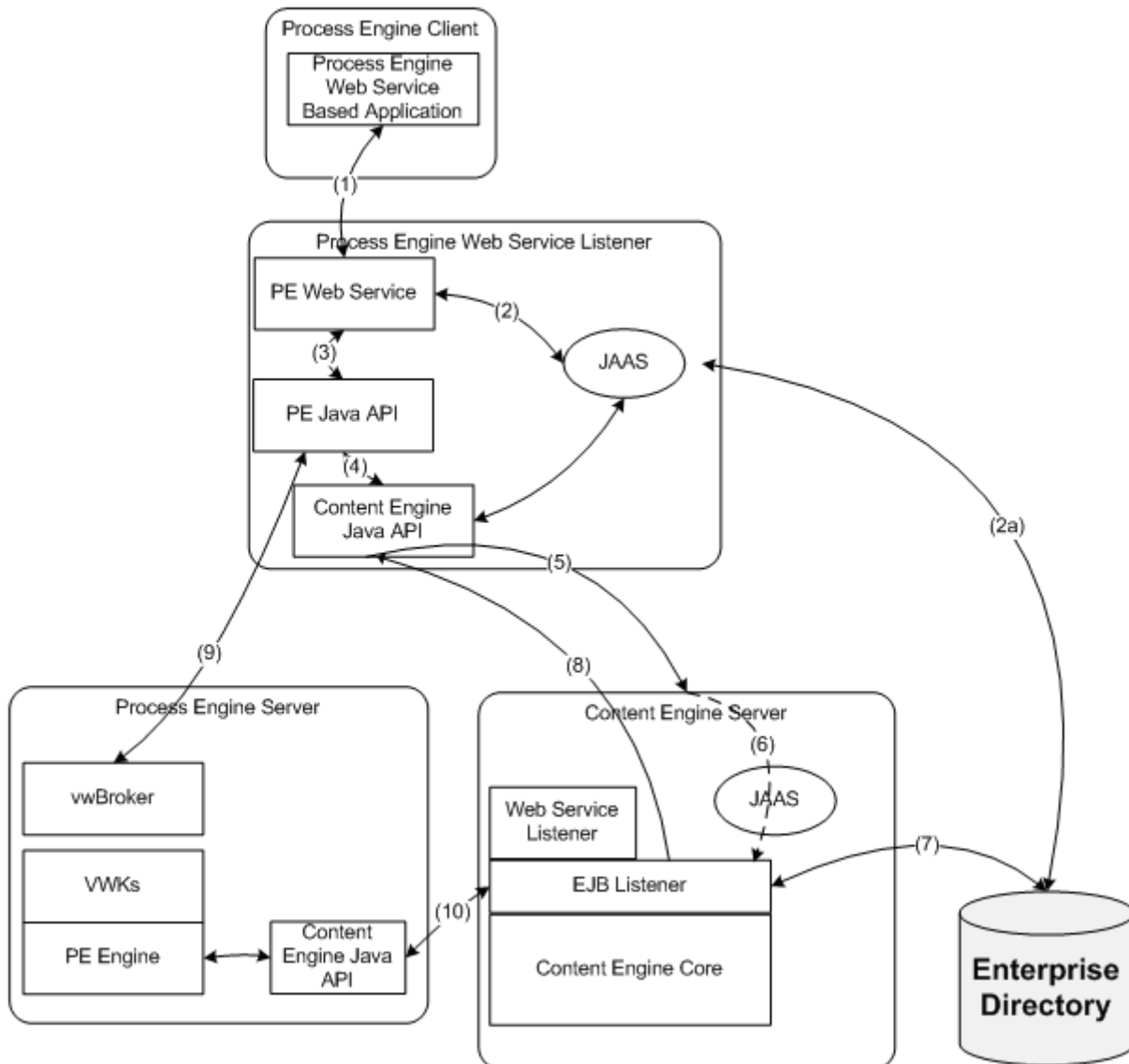
Authenticating the Process Engine Java API client using the Content Engine web service transport

Process Engine authentication through Content Engine over the web service transport is supported only for the username/password case. Alternate credential types over the web service transport are not supported.

The diagram for this scenario is the same as in the previous topic, with the exception that the arrow in step 4 goes to the Web Service Listener, rather than going to the EJB Listener. The Web Service Listener then extracts the user's credentials from the WS-Security header, and performs a JAAS login using a Login Module that has been configured at the server. Once that step is complete, the web service listener forwards the call along to the EJB listener, and everything thereafter is the same as in the previous topic.

The Process Engine Web Service API client

The following graphic shows the Process Engine web service listener running on a separate server from the Content Engine server. Although this is correct from a conceptual point of view, the default configuration is for the Process Engine web service listener to be co-resident with the Content Engine server.



Process Engine web service client sends a request to the Process Engine web service listener.

This scenario is conceptually similar to that discussed in "Authenticating the Process Engine Java API client using Content Engine EJB Transport" section above, with an added layer. In this case, the Process Engine web service client sends a web service request to the Process Engine web service listener (step 1). The Process Engine web service listener is then responsible for performing the JAAS login. Once the

Process Engine web service listener has obtained a valid JAAS Subject, it makes a call to the Process Engine Java API. From that point on, the steps in this scenario are the same.

NOTE The only credential type supported for this scenario are username/password credentials. The Process Engine web service does not support Kerberos credentials, nor does it support the Web Services Extensible Authentication Framework (WS-EAF). This support is planned for a subsequent release.

Application Engine authentication

FileNet P8 Application Engine server hosts the Workplace web application, Workplace Java applets, Process Engine applets, and application development tools. Application Engine is the presentation layer for both process and content. A number of different components run on the Application Engine. The sections below discuss how each of these components deals with authentication and single sign-on (SSO) integration.

Workplace XT application

This end-user web application provides access to the document management and business process management capabilities of FileNet P8 in much the same manner as the Workplace application. You will see similarities in the features offered by both applications, but there are some important differences, such as the following:

- Workplace is built using the Web Application Toolkit and can be customized using that toolkit, but Workplace XT has no equivalent toolkit and is used out-of-the-box.
- Workplace supports both application-managed and container-managed authentication, but Workplace XT supports only container-managed authentication. The container-managed authentication support for Workplace XT is identical to the Workplace application's container-managed authentication described below.

NOTE Unless otherwise noted, the information below for the Workplace application also applies to Workplace XT.

Workplace application

This end-user web application provides access to the document management and business process management capabilities of FileNet P8. Workplace also supports extended FileNet P8 capabilities such as forms management, records management, and portals. This is one example of a Java-based thin client solution, as described in the section Browser-based clients of J2EE application servers. (Most of the considerations discussed in that section apply for the Workplace application.)

Workplace is built using the Web Application Toolkit, and runs within a web container on a J2EE application server, positioning it well to participate in the JAAS-based authentication framework of FileNet P8. The Web Application Toolkit is an extensible framework for building web applications. Programmers can use the toolkit to customize Workplace functionality or to build customized web applications.

The following topics discuss how each of the high-level authentication options discussed in Browser-based clients of J2EE application servers apply to the Workplace application.

Application-managed authentication

Application-managed authentication (the mode supported by earlier versions of Workplace) is basically a forms-based authentication, but the Workplace application performs the redirection of unauthenticated user requests to a log in page, and encodes the credentials supplied to the log in page, in the user's JSP session. This mode supports username/password credentials only. The credentials collected from the Application Engine custom log in page are used to programmatically perform a JAAS login. This mode is still the current default behavior of Workplace.

Container-managed authentication

NOTE The information in this topic applies to the Workplace and Workplace XT applications.

In this mode, the application does not control the authentication process. The deployment descriptor for the application specifies the security constraints required to access application pages.

The deployment descriptor specifies the authentication method that should be used. The following standard methods defined by the Servlet specification (see Browser-based clients of J2EE application servers) are supported:

- **Forms-Based Authentication:** The container redirects the user to an HTML page, where the user's credentials are collected.
- **Basic Authentication:** The container uses standard HTTP options to direct the user's browser to pop up a dialog box and prompt for user name/password credentials.
- **HTTPS Client Authentication:** This mechanism requires each user to have its own Public Key Certificate (PKC), and requires the use of an HTTPS (SSL) connection between the client and the server.

Perimeter authentication

This option is how most SSO products integrate with a J2EE application server. Client browsers running Workplace are redirected to a proxy server that authenticates the caller, and places a token in an HTTP header for them. When the request reaches the server, the container extracts the credentials and invokes SSO provider software that performs a JAAS login using them. This is known as a perimeter authentication because the actual authentication occurs outside of the container. Clients are already authenticated before their servlet requests arrive at the server. See JAVA-based client authentication and the examples in Single sign-on integrations via JAAS for more information.

Perimeter authentication lets Workplace leverage standard integrations between the application server vendors and the SSO technology vendors.

NOTE Support for SSO in Workplace and Workplace XT is limited to two specific combinations that IBM FileNet has qualified, as discussed in Single sign-on integrations via JAAS. If you are implementing SSO in an IBM Tivoli Access Manager WebSEAL 6.0 environment, you must configure WebSEAL for transparent junctions. For more information and configuration details, see your IBM product documentation.

Workplace applets

The Workplace and Workplace XT applications include the following Java applets:

- Process Task Manager
- Process Administrator
- Process Configuration Console
- Process Designer
- Process Tracker
- Publishing Designer (NOTE Publishing is not supported in Workplace XT.)
- Search Designer
- Image Viewer
- Process Simulation Analyzer
- Process Simulation Console
- Process Simulation Designer

The section Applets and reverse proxy servers in the JAVA-Based Client Authentication (JAAS) topic discusses some of the concerns that must be taken into consideration when using Java applets across a reverse proxy server. The Workplace and Workplace XT applets address these concerns in the current release.

The Workplace and Workplace XT applets exchange XML data with the Application Engine server, and this XML data contains URL references that are not, by default, translated properly by reverse proxy servers. For this reason, some custom configuration work is required to get reverse proxy servers to work properly with the applets.

Application Integration clients

The Workplace Application Integration toolkit allows third-party client applications to integrate with Workplace and Workplace XT. Windows-based applications, such as Microsoft Word and Microsoft Outlook, can leverage Application Integration COM components for this purpose. These COM components interact with the Application Integration servlets, sending XML requests over HTTP, and receiving XML responses.

Application Integration clients are not able to participate in single sign-on solutions (SSO); they are restricted to username/password-based authentication. Application Integration clients can co-exist with thin client applications that participate in an SSO solution. Application Engine can simultaneously support clients using container-managed authentication for SSO (for example, Workplace XT clients and browser-based Workplace clients coming in through a reverse proxy server), and clients that are using application-managed authentication (such as Microsoft Office clients using the Application Integration Toolkit).

If Workplace is configured to use one of the supported end-to-end SSO solutions using a reverse proxy server, then the reverse proxy server must be configured to pass Application Integration traffic through unmodified, or the Application Integration clients must be configured to bypass the reverse proxy server, going directly to the Application Engine server.

FileNet P8 Portlets clients

FileNet P8 Platform includes the FileNet P8 Portlets product, a portal integration framework that provides commonly-required content and process functionality within third-party portal products. This framework includes portlets that provide end-user functionality such as authoring content, browsing features for accessing content, and providing a view of a user's inbox.

When portlets are used within Workplace or Workplace XT in the My Workplace page, they will participate in the same SSO integrations as Workplace or Workplace XT itself.

NOTE For this scenario, Workplace must be configured for container-managed authentication.

To support SSO solutions within a third-party portal integration, the portlets container must first be configured for container-managed authentication. When this configuration is in place, the FileNet P8 portlets recognize that the container has taken care of authentication for them.

In cases where links from a portlet launches Workplace XT or Workplace (configured for container-managed authentication), an SSO infrastructure must also be in place to propagate the SSO provider identity tokens to the container that is hosting the application. Container-managed authentication is required to accept these tokens.

WebDAV clients

As of the current release, the WebDAV implementation has moved from Content Engine to Application Engine. WebDAV is now implemented as a separate servlet, residing in the same web container as Workplace.

WebDAV is an HTTP-based API that does not present an HTML user interface to clients. For this reason, WebDAV is not able to take advantage of forms-based authentication schemes. In the current release, the WebDAV servlet must always be configured for application-managed authentication, managing its own user challenge process. The WebDAV servlet will be configured so that there is an application-level BASIC HTTP challenge/response mechanism used to gather WebDAV client credentials.

If Workplace is configured to use one of the supported end-to-end SSO solutions using a reverse proxy server, then the reverse proxy server must either be configured to pass WebDAV traffic through

unmodified, or the WebDAV clients must be configured to bypass the reverse proxy server, going directly to the Application Engine server.

Application-managed authentication

If Application Engine is configured to use forms-based authentication (be it container-managed forms-based authentication, the default application-managed forms-based authentication, or a perimeter authentication scheme using a form to collect credentials) then WebDAV will not be able to participate in this. This will typically be the case, so the WebDAV servlet will be required to manage its own user challenge process. The WebDAV servlet will be configured so that there is an application-level BASIC HTTP challenge/response mechanism used to gather WebDAV client credentials.

Container-managed authentication

When Workplace is configured for a container-managed authentication using a non-HTML login method, such as BASIC or CLIENT-CERT authentication, then the WebDAV servlet can be configured as a resource protected by this mechanism as well.

If a perimeter authentication mechanism is in place that uses a non-HTML mechanism such as HTTP Basic authentication, then WebDAV clients can be authenticated by that perimeter mechanism and participate in an SSO environment.

Kerberos for Content Engine

Kerberos support is available to clients of the Content Engine Web Services API, the Content Engine .NET API, and the Content Engine COM API. This support includes the FileNet Enterprise Manager (EM) and Capture applications. Many third-party COM and web-service-based applications are also available that can take advantage of this Kerberos support.

Introduction

Content Engine (CE) running on Windows servers can use Kerberos for single sign-on (SSO) authentication. This is also known as "integrated logon" since CE takes advantage of an earlier Windows logon to securely establish a user's identity without asking the user for a password again.

Kerberos facilitates user authentication over an untrusted network of disparate systems. One feature of Kerberos allows the issuing of a service ticket that can be used to authenticate a client with a particular service. In short, the client contacts a Key Distribution Center (KDC), which is a domain controller (DC) on a Windows system, and asks for a ticket to a service with some Service Principal Name (SPN). The KDC knows everyone's password keys and uses the client's key to decrypt the request, and then uses the SPN's key to encrypt the service ticket, which it passes back to the client.

The KDC also creates a short-term "session" key specifically for this ticket which can be used by both the client and the service. The client passes this ticket plus some additional encrypted timestamp information to the service, which then uses its own password key to decrypt parts of the ticket, learns the name of the client and also checks the timestamp information for the integrity of ticket, thus authenticating the client if everything is okay. These service tickets are good for some period of time, typically 8 or 10 hours, and can be reused by the client anytime during that period that it needs to authenticate itself to the service.

See also the "Kerberos Credentials" section of Web-Services-Based Client Authentication Via Ws-Security.

Prerequisites

Since Kerberos can be difficult to get working correctly, the first step is to make sure that a number of prerequisites are met.

System prerequisites

Only Windows clients can use Kerberos authentication, although either Windows or UNIX CE server systems can accept Kerberos credentials. CE server systems must use Windows 2003 or later. .NET clients should use Windows 2000 or later systems. Earlier versions of Windows, such as Windows NT® or Windows 98, do not support Kerberos.

Regardless of whether you use Windows or UNIX CE servers you must use Windows Active Directory as your directory service. In the Windows case, each CE server must be the member of an Active Directory domain.

Domain / account prerequisites

A Windows client system needs to be in a domain (not a workgroup) and the user needs to be logged on as some type of domain account.

NOTE You must log on using a domain account. Logging on a Windows client system as a local account will not work.

A Windows server system also needs to be a member of some domain (not a workgroup), but there is no restriction on whether the logged on account is in the domain or local. UNIX servers need only access to Active Directory.

The client system and the CE server system should be in the same domain (or, in the case of UNIX servers, access the same Active Directory) unless the steps described in Cross-Realm Kerberos authentication have been followed.

Stand-alone .NET client prerequisites

A .NET stand-alone client needs at least .NET 1.1 and WSE 2.0 installed on the Windows system, although .NET 2.0 and WSE 3.0 (or later) are recommended.

CE Java server prerequisites

1. CE needs at least the 1.4.2 Java runtime environment (JRE), but it is recommended that the 1.4.2 Java Development Kit (JDK) be installed to make use of some debugging utilities. Although not required, the JAVA_HOME environment variable may be set to allow the Java tools in the JDK to be accessed on Windows systems as follows:

```
set JAVA_HOME=e:\j2sdk1.4.2_11
%JAVA_HOME%\bin\ktab
```

2. The Java virtual machine (JVM) requires that there be a Kerberos configuration file specifying such values as the following:

```
[libdefaults]
default_realm = MYDOM.EXAMPLE.COM
kdc_timesync = 1
ccache_type = 4
forwardable = true
proxiable = false
default_tgs_enctypes = des-cbc-md5 des-cbc-crc
default_tkt_enctypes = des-cbc-md5 des-cbc-crc
[realms]
MYDOM.example.COM = {
kdc = mydomainsys:88
admin_server = mydomainsys
default_domain = mydom.mycompany.com
}
[domain_realm]
.mydom.mycompany.com = MYDOM.MYCOMPANY.COM
```

The Kerberos "realm" values (equivalent to Windows domains) should be adjusted as well as the "kdc" (Kerberos Key Distribution Center, which is the Primary Domain Controller on Windows systems) and "admin_server" values.

This should be saved on Windows systems in the c:\winnt\krb5.ini file. Note that c:\winnt must be used even if Windows is installed in c:\windows or is on some other drive. An alternative to using c:\winnt\krb5.ini is to save the file elsewhere and specify its location as an argument when starting up the JVM, as in:

```
-Djava.security.krb5.config=c:/cemp/config/krb/krb5.conf
```

On UNIX this file should be /etc/krb5/krb5.conf and on Linux®, this file should be /etc/krb5.conf.

3. Microsoft increased security in Windows 2000 SP4, Windows XP SP2 and Windows 2003 and in doing so broke Java's ability to use the system's cached Kerberos Ticket Granting Tickets (TGTs) for integrated logins. This can be patched by running <installdir>\FileNet\Content Engine\Kerberos\AllowTgtSessionKey_fix_for_Krb5LoginModule.reg. This uses RegEdit to add a new value to the Windows system registry which will disable the new security feature. This registry edit may also be used harmlessly on early Windows Service Packs.

Middleware, web application servers

In general Kerberos is not supported in those cases where a .NET stand-alone client is going through a web application layer before going to the CE server. The problem is that the web application layer must typically do actions for (impersonate) any number of users. This impersonation is not supported by .NET's WSE 2.0 API and is not supported in the current release.

The exception to this rule is that browser-based clients may use SPNEGO authentication (which uses either Kerberos or NTLM authentication) and connect to a web application, which in turn calls CE. This will work because the application server itself uses Kerberos to authenticate the user and then this

identity is propagated to CE. Setup for this scenario is beyond the scope of this topic, but see WebLogic's Configuring Single Sign-On with Microsoft Clients: <http://e-docs.bea.com/wls/docs81/secmanage/sso.html>.

Creating the Kerberos Service Principal Name (SPN) identity

Some Background

A Kerberos Service Principal Name (SPN) is simply a name chosen to represent some service (CE on a particular server in our case). The SPNs used by CE should always be in one of these two:

```
FNCEWS/cemp01  
FNCEWS/cemp01@MYDOM.EXAMPLE.COM
```

The leading FNCEWS is CE's service name and is needed on all its SPNs. The other parts of the SPN show where the CE server resides, as a simple DNS name or a simple DNS name qualified with a domain.

As an example, the CE server's undistinguished DNS name (the hostname), might be MYCEMP01 and its domain might be MYDOM.EXAMPLE.COM. In this case the two SPNs would be:

```
FNCEWS/mycemp01  
FNCEWS/mycemp01@MYDOM.EXAMPLE.COM
```

The hostname should always be lower case and the domain name always upper case. Windows systems are mostly case-insensitive but not everything is, so care should be used to always use the proper case.

The SPN "identity" is a Windows domain user account that has been mapped to the SPN. This special user account has a password and from that the Key Distribution Center (KDC) derives a key, which will be used to encrypt parts of the Kerberos ticket. This key is a shared secret that is known only to the KDC, which issues Kerberos tickets, and to the service itself (CE in this case).

With these definitions of SPN and the identity account, this brings up a number of questions and their answers...

How does the client know what the SPN is? In the case of the FileNet Enterprise Manager, it derives this from the URL it uses to connect to the CE. If that URL is, for instance, <http://mysvr:9080/wsi/FNCEWS40SOAP>, it uses an SPN of FNCEWS/mysvr. If the URL is, <http://mysrv.mydom.example.com:9080/wsi/FNCEWS40SOAP>, it uses FNCEWS/mysrv@MYDOM.EXAMPLE.COM. Other .NET clients use a simple algorithm like this or directly specify the SPN.

How does the KDC know about the SPN and how does it know what key to use with that? The answer, on Windows systems, is that there must be one domain user account set up (with some password, of course), this user account must be specifically mapped to one or more SPNs and that account provides the key. The KDC is referred to on Windows systems as a Domain Controller.

How does the CE server know what its SPN is? Well it actually does not need to know this, but it can figure out the name of the identity user account by using FNCEWS_ + hostname (for example, FNCEWS_mycemp01 for our example CE server named MYCEMP01). If some other identity user account name must be used for some reason, as it would if this server was a member of a cluster, then the name can be directly specified by the serviceAccountName option (see "KrbServiceLoginModule Options" and "Using Kerberos with a Cluster of CE Servers").

How does the CE server know what key to use for decrypting its part of a Kerberos ticket? It does this by logging on to the identity user account by using the password stored in a keytab, a special Kerberos table of users and their passwords. This special login gets the Kerberos Ticket Granting Ticket (TGT), which in turn gives the server access to its key.

The following steps of making an SPN Identity are not trivial, partly because Windows allows Kerberos interoperability with Java and UNIX, but has never tried to make it user friendly.

Step 1 - Choose the SPNs

As mentioned earlier the SPNs should be in the form

```
FNCEWS/hostname  
FNCEWS/hostname@DOMAIN.COM
```

where you should substitute your own CE server's name for hostname and that server's domain for DOMAIN.COM. The hostname should be an undistinguished (i.e., no dots in it) DNS name and all lower-case. The domain name should be all upper-case. In the example setup, the hostname is MYCEMP01 and it is in the MYDOM.EXAMPLE.COM domain, so the two SPNs would be FNCEWS/mycemp01 and FNCEWS/mycemp01@MYDOM.EXAMPLE.COM.

If this is a cluster, set hostname to the cluster name (see "Using Kerberos with a Cluster of CE Servers").

Case matters when choosing the SPNs! Make sure the hostname is lower-case and the domain name is upper-case.

Step 2 - Create an Active Directory user account

Create an AD domain user account that will be the "identity" behind the SPN. This account will be mapped to the SPNs in the next step and, crucially, has a password which, through some hashing, serves as a symmetric encryption/decryption key for parts of the Kerberos tickets.

The name of this account should be FNCEWS_ + hostname and will generally be similar to the short SPN name chosen above with an underscore (_) in place of the slash (/). There are two exceptions to this, however: 1) for clusters, where hostname would be a cluster name instead; and 2) if the derived FNCEWS + hostname name would be longer than 20 characters, as it would with hostnames longer than 13 characters. The latter exception is because of a bug on some versions of WebSphere's Java with long User Principal Names (UPNs) and complications on all platforms if the UPN does not match the "User Logon Name (pre-Windows 2000)" name, which is also restricted to 20 characters or less. If either of these exceptions holds, then you must choose a 20 character or shorter unique user name (for example, FN_long_user_name_12) and configure the Kerberos login module to use this new name by setting the serviceAccountName option (see "KrbServiceLoginModule Options"). This "identity" user account name is case-sensitive and should be remembered as it will be needed when creating this account and for some later steps.

To create the account, open the Active Directory Users and Computers tool on a Domain Controller in the CE's Windows domain. Create a Windows domain user (not computer) account that will be used for the SPN. This account must be in the same domain as that of the CE's system. Enter the "identity" user account name, make sure the "Password never expires" is checked and nothing else, and then enter a password for that account and confirm it. Remember this password as it will be needed later.

Using the "Active Directory Users and Computers" snap-in again, select the account you just created, right-click it and then select "Properties..." from the context menu. Select the "Account" tab and then check "Use DES encryption types for this account" (you will need to scroll down to the bottom of the "Account Options" control to find this option).

This "identity" user account should not be used for other purposes other than acting as a Kerberos identity for a single CE server or cluster. In particular this account should not be shared by several CE servers unless, of course, they are all in the same cluster.

Step 3 - Map the Active Directory user account to the SPN

After the "identity" user account is created, it must be mapped to the proper SPNs. This is done by using Microsoft's setSPN utility, which is available on Windows 2003 Servers (and later). Windows 2000 Servers also have a setSPN utility, but this will have to be downloaded from the Internet. Type something like these two lines in a command window on a Windows CE's system or a Windows Domain Controller system...

```
setSPN -a FNCEWS/mycemp01 FNCEWS_mycemp01  
setSPN -a FNCEWS/mycemp01.mydom.example.com FNCEWS_mycemp01
```

(Substitute in the SPN you have chosen and the name of the "identity" account just created for mycemp01 and the domain mydom.example.com).

Microsoft's ktpass utility is another way to map SPNs, but do not use this for CEs on Windows systems! This utility is needed for Unix Kerberos realms, which are not yet supported.

Step 4 - Create a Kerberos keytab entry for the SPN on the CE system

To avoid having CE prompt for the SPN's password each time it needs it, set up a keytab (key table) that has the identity user's proxy account and password. Do this by entering a command line to run the "ktab" Java utility on the CE server system. This command line varies depending on the type of the application server.

On Windows-based WebLogic or JBoss application servers, enter:

```
%JAVA_HOME%\bin\ktab -a FNCEWS_mycomp01@MYDOM.EXAMPLE.COM
```

On Windows-based WebSphere application servers, enter (on a single line):

```
%JAVA_HOME%\bin\java com.ibm.security.krb5.internal.tools.Ktab -a FNCEWS_
comp01@MYDOM.EXAMPLE.COM
```

On UNIX-based application servers, replace %JAVA_HOME%\bin\ with \${JAVA_HOME}/bin/ in the lines above.

After typing this line, enter the password when prompted.

The user name is case-sensitive and must be exactly the same case in the ktab command line as the user name defined in Windows Active Directory. For instance, if the user account was created with the name "FNCEWS_myname", then there might be mysterious, hard-to-identify problems later if "FNCEWS_MyName" were used on the ktab command.

Notice the underscore ("_") between FNCEWS and the system name. FNCEWS_mysystem is the accountname, not the SPN, which has a slash "/" delimiter instead (as in FNCEWS/mysystem). If you mistakenly enter the SPN, simply run ktab again using the account name. You can clean up these mistakes by deleting bad entries using the same command line as above, but substituting -d (delete) for the -a switch.

The ktab utility is part of the JDK distribution and this will create a file named krb5.keytab in the logged on account's "home" directory. This home directory would be something like:

```
c:\Documents and Settings\administrator.MYDOM\ (Windows)
/home/username (Unix)
```

NOTE Logging on as another user will change this home directory, which can be a problem if the krb5.keytab file is different in that directory or even absent.

Since the keytab contains sensitive information (encrypted passwords), you as the administrator would probably not want to use the default keytab as above. Instead, you would probably want to put restrictive file-system permissions on the keytab so that only a minimum number of accounts can even read it. To do this, specify an extra option, -k path / mykeytab, on the ktab command line.

Enabling Kerberos on the application server

WebLogic

To enable Kerberos under WebLogic, you must set up a special Engine Kerberos Service Authentication Provider, as follows:

1. Copy either the Engine-authenticator-wl8.jar file (for WebLogic 8.1.x) or the Engine-authenticator-wl.jar (for WebLogic 9.1 and later) to the following location:

%WL_HOME%\server\lib\mbeantypes (Windows).

\${WL_HOME}/server/lib (UNIX).

Both JAR files can be found in the Program Files\FileNet\ContentEngine\Kerberos directory for Windows, or \${WL_Home}/server/lib/mbeantypes for UNIX.

2. Start the WebLogic server and run the administrative console.
3. In the administrative console's Security / Realms / *myrealm* / Providers / Authenticators pane (where *myrealm* is the default name of the security realm, which may be different in your environment),

create a new Engine Kerberos Service Authenticator and set its name to, for example, EngineKrbAuthenticator.

4. In the pane that follows, change the Control Flag setting to SUFFICIENT.
5. Click Create.
6. Go back to the Security / Realms / *myrealm* / Providers / Authenticators pane and click Re-order the Configured Authentication Providers.
7. Click EngineKrbAuthenticator and use the arrows to shift it above any LDAP providers, but below the DefaultAuthenticator.
8. Click Apply.
9. Navigate back to the EngineKrbAuthenticator page and click Details or Provider Specific.
10. Make any changes necessary on this page. For instance, set the Debug option.
11. Save the changes.

WebSphere

To enable Kerberos under WebSphere, do the following:

1. Copy the Engine-authn.jar file to the following location:

%WAS_HOME%\lib (Windows).

\${WAS_HOME}/lib (UNIX)

Both JAR files can be found in the Program Files\FileNet\ContentEngine\Kerberos directory for Windows, or \${WL_Home}/server/lib/mbeantypes for UNIX.

2. Start the WebSphere server and then run the administrativeconsole.
3. In the Security or Global Security page, click Authentication Methods > LTPA.
4. Enter a password and confirm it. (This password will be needed by other servers, so remember it.)
5. Click Security > Global Security and change Active Authentication Mechanism to LTPA.
6. In the Security or Global Security page, click JAAS Configurations > Application Configurations.
7. Click New and enter FileNetP8KerberosService for Alias, click Apply, and then click JAAS Login Modules.
8. Click New and in Module Classname enter:

```
com.filenet.engine.authentication.kerberos.login.KrbServiceLoginModule
```

and leave other fields as-is. Then click OK.
9. If desired, add any options by clicking the new KrbServerLoginModule entry, click Custom Properties, then New, and then enter the option name (for example, *debug*) and its value (for example, *true*). Click OK and then click JAAS Login Modules.
10. Click New and in Module Classname enter:

```
com.ibm.ws.security.server.lm.ltpaLoginModule
```

and leave other fields as-is. Then click OK.
11. Click New and in Module Classname enter

```
com.ibm.ws.security.server.lm.wsMapDefaultInboundLoginModule
```

and leave other fields as-is. Then click OK.

12. Save the changes.

JBoss

To enable Kerberos under JBoss, do the following:

1. Copy the Engine-authn.jar file into the %JBOSS_HOME%\server\default\lib directory (or \${JBOSS_HOME}/server/default/lib on UNIX/Linux). This JAR file can be found in the <...>\FileNet\Content Engine\Kerberos directory.
2. Edit the %JBOSS_HOME%\server\default\config\login-config.xml file (or \${JBOSS_HOME}/server/default/config/login-config.xml on UNIX/Linux) by adding

```
<!-- Kerberos login module -->
<login-module code=
"com.filenet.engine.authentication.kerberos.login.KrbServiceLoginModule"
flag="sufficient">
<module-option name = "debug">true</module-option>
</login-module>
```

right before the **LdapExtLoginModule** or **LdapLoginModule** line in the "FileNet" stanza. Also change the flag of the **LdapExtLoginModule / LdapLoginModule** line from "required" to "sufficient" if necessary.

3. Edit the **login-config.xml** file again and add the following stanza after the last of the other <application-policy> entries...

```
<application-policy name="FileNetP8KerberosService">
<authentication>
<login-module code="com.filenet.api.authentication.jboss.login.FnClientLoginModule"
flag="required">
</login-module>
</authentication>
</application-policy>
```

KrbServiceLoginModule options

The KrbServiceLoginModule (or WebLogic Engine Kerberos Authentication Provider) does the Kerberos service authentication on the CE server. This login module has several options that can change its behavior:

debug - when true will output additional debugging information on the console and/or server log. The default is false. You may wish to set this true when first setting up Kerberos or afterward if attempting to debug a problem with Kerberos authentication.

useShortNameAsPrincipal - when true will use the user's short name as the principal, if false then will use the full Kerberos name in the form: username@REALM.COM. The default is false. This option is ignored on WebSphere which effectively always requires short names.

storeGssContext - when true will add a GSSContext as a private credential. This may be used by the server to encrypt and sign messages between the server and the client, but will need substantial programming on both to accomplish this. When false then this private credential will not be added. The default (and recommended) setting for this is false.

loginUsingTicketSpn - if true will attempt to login using the SPN (Service Principal Name) of the Kerberos service ticket. The default is false, which will attempt to login using the normal FNCEWS_computername account name. It is recommended that this be left false.

serviceAccountName - this specifies the account name that the service will use when it is logging in. If not specified, this defaults to FNCEWS_computername. This option should only need to be set to some account that is shared by every server if CE is clustered or if on WebSphere systems and the default name would be longer than 20 characters. This option is ignored if loginUsingTicketSpn is true.

tgtLoginConfigName - this option specifies the name of a JAAS configuration that will be used when the service initially logs in to get its Kerberos TGT (Ticket Granting Ticket).

For example, if tgtLoginConfigName=KrbTgtLogin, then there could be a JAAS configuration entry such as:

```
KrbTgtLogin {  
  com.sun.security.auth.module.Krb5LoginModule required  
  debug=true useKeyTab=true storeKey=true  
  keyTab="c:/etc/krb5.keytab";  
};
```

If this is not set, the default is to use an internal configuration that is tailored for the application server. It is recommended that this option not be set as the default will almost always work correctly.

keytabPath - this option specifies the keytab file, which will be used for the service's initial TGT login, and will be set like the following examples:

```
keytabPath="file:/c:/etc/krb5.keytab" (WebSphere)
```

```
keytabPath="c:/etc/krb5.keytab" (WebLogic, JBoss, or other potentially supported application servers)
```

If this option is not set, then the default Java keytab will be used, which is usually found in < home>/krb5.keytab. For example:

```
C:/Documents and Settings/mike.MYDO/krb5.keytab.
```

The keytabPath option needs to be used when the application server is running as a Windows service or if the keytab is not in the default location.

cacheSize - specifies the size of the ticket cache used on JBoss application servers. This cache works around "request is replayed" errors that happen because of the way that JBoss sometimes uses a Kerberos ticket twice during a normal authentication and triggers the error on the second use. This defaults to 100 on JBoss and is ignored on other application servers. It is recommended that this option not be set, unless running on JBoss servers that have been getting "Request is a replay" errors, in which case you could try values greater than 100.

Using Kerberos on the client

To use Kerberos on the client, see "Using Kerberos on the Client" in Working with Security in the Content Engine API Developer's Guide.

Using Kerberos with a cluster of CE servers

The SPN used for a cluster is different than that used for individual CE servers. For a cluster, pick a unique SPN, say FNCEWS/cluster01, and create a single domain user, FNCEWS_cluster01, that will be the Kerberos identity account for that SPN. Run the setspn utility to map this SPN to this identity account by typing:

```
setspn -a FNCEWS/cluster01 FNCEWS_cluster01  
setspn -a FNCEWS/cluster01.mydom.example.com FNCEWS_cluster01
```

All CE servers must then be set up to use this cluster-wide identity rather than the server-name identity normally used. Do this as follows:

For WebSphere and JBoss (and other potentially supported application servers):

Add a new option, `serviceAccountName=FNCEWS_cluster01`, to the JAAS configuration for `KrbServiceLoginModule` on each application server.

For WebLogic:

Set the Service Account Name option for the Engine Kerberos Authentication Provider.

In cases where one client references a server's URL directly (recommended for EM) and another client could reference the cluster URL of that server (possible with customized .NET clients), then there must be additional SPN mappings to the same identity account. For example:

```
setspn -a FNCEWS/cluster01 FNCEWS_cluster01
setspn -a FNCEWS/cluster01.mydom.example.com FNCEWS_cluster01
setspn -a FNCEWS/mycomp01 FNCEWS_cluster01
setspn -a FNCEWS/mycomp01.mydom.example.com FNCEWS_cluster01
setspn -a FNCEWS/mycomp02 FNCEWS_cluster01
setspn -a FNCEWS/mycomp02.mydom.example.com FNCEWS_cluster01
```

It is important to remember the following:

- All of these SPNs must map to the same identity account (`FNCEWS_cluster01` in the example above).
- All the clustered servers must have the `serviceAccountName=FNCEWS_cluster01` option set in the JAAS configuration as previously mentioned.
- You must set up an `FNCEWS_cluster01` keytab entry on each server.

Finally, it is extremely important that no single SPN be mapped to more than one identity account. For example, the following two `setspn` commands, even if they were done at different times would cause unexpected "The network path not found" errors on the clients:

```
setspn -a FNCEWS/mycomp01 FNCEWS_mycomp01
setspn -a FNCEWS/mycomp01 FNCEWS_cluster01
```

This example has the same SPN, `FNCEWS/mycomp01`, mapped to two different identities: `FNCEWS_mycomp01` and `FNCEWS_cluster01`. Unfortunately, this can occur innocently enough by starting with only a single CE server and later expanding to a cluster of CE servers. Likewise, there is no way to check for duplicates of this sort in Microsoft's `setSPN` utility. See the later "Solving Kerberos Problems" topic for what to do if this error condition occurs.

Cross-Domain authentication

Multiple Windows domains, where the clients are in one or more domains and the CE server is in another, can be made to work with Kerberos but special considerations must be taken into account. The first is that the domains (or realms in Kerberos terminology) must accept the identity established in another domain. This means that one domain "trusts" another.

There are three major ways that Windows systems have of establishing this domain to domain trust:

1. creating a domain forest;
2. establishing a transitive trust between two domain forests;
3. establishing external trusts between sets of two domains.

Domain forests were introduced in Windows 2000 Server and allow one or more hierarchical domain trees with parent/child relationships. In the forest, the trust is *transitive* in that one domain trusts another domain no matter where it is in the forest. Windows 2003 Server went one step further and allowed transitive trusts from one forest to another, although these "inter-forest" trusts are only available if both forests have been promoted to Windows 2003 mode. External trusts originated in Windows NT and are explicit trust links that are set up between one domain and another that are not transitive in that if domains A and B trust each other and domains B and C trust each other, then this does not imply that A

trusts C or C trusts A. Two-way external trusts mean that each domain trusts the other (one-way external trusts are not supported by Kerberos).

These different type of domain to domain trusts work well with Kerberos authentication, although each domain must have its directory configuration set up in the GCD and the application server must have LDAP providers set up for each domain. The GCD directory configuration is done using EM (right-click the P8 domain name, select Properties from the context menu, click Directories and add a new configuration for each domain).

- WebLogic security can be enabled for each domain by adding and configuring a new Active Directory LDAP authentication provider for each.
- JBoss requires multiple **LdapExtLoginModule** or **LdapLoginModule** setups in the **FileNet JAAS** stanza.
- Multiple LDAP user registries may be set with WebSphere, but only in version 6.1 and beyond. Earlier versions only support a single LDAP provider.

The second consideration for cross-domain Kerberos is that the SPNs will most often need an “ @ ” qualifier, such as **FNCEWS/mycemp01@MYDOM.EXAMPLE.COM**. The shorter, unqualified SPN, like **FNCEWS/mycemp01**, may also be used in some cases, but might have problems if the server name **mycemp01** is ambiguous as that name appears more than once in a forest or if that shorter form is used with an external two-way trust or transitive two-way trust. If the shorter SPN does not work, then the .NET client will likely get "The network path was not found" error.

EM also needs special handling for cross-domain Kerberos authentication as it tries to derive the SPN from the configured CE server URL. There will often be cases where the URL will need a fully qualified domain name to get the correct SPN. For example, a CE URL of **http://mycemp01.mydom.example.com:9080/wsi/FNCEWS40SOAP** will be turned into an SPN of **FNCEWS/mycemp01@MYDOM.EXAMPLE.COM**, which should work. A shorter, unqualified URL such as **http://mycemp01:9080/wsi/FNCEWS40SOAP** would produce an SPN of **FNCEWS/mycemp01**, which might not work.

A final note is that the client's domain from Kerberos's point of view is the domain of the logged-on user, which will not necessarily be the domain of the client system. For instance, a client system is in domain NORTH that trusts the domain SOUTH and the user as logged onto the system as Frank@SOUTH; in this case Kerberos will use SOUTH as the client's domain.

Solving Kerberos problems

Troubleshooting problems with Kerberos can be a bit complex, given the number of computers and amount of software that may be involved.

The error codes that follow are displayed in bold italic (for example, "KDC has no support for encryption type (14)"). Please note, however, that exact wording depends on the application server vendor and version). The following examples use mycemp01 as the CE server name and MYDOM.MYCOMPANY.COM as the Windows domain name.

In most cases, the important part of the error will be towards or at the end of the error message. For instance, if the error message returned by the client was "WSE594: InitializeSecurityContext call failed with the following error message: The network path was not found. ", the important part of this message is the "The network path was not found. "

.NET client or EM problems

"Unable to connect to the remote server. No connection could be made because the target machine actively refused it."

If you get this error, carefully check the URL, particularly the associated port number, that you are using to connect to CE. Also make sure that Content Engine is running.

"The network path was not found"

This error might indicate one of several underlying problems:

1. EM has a bad CE server URL.

Enterprise Manager calculates the Kerberos SPN by taking the host part of the URL and prefixing it with FNCEWS/, so that, for example, a URL of `http://mycomp01:9080/wsi/FNCEWS40SOAP/` would yield an SPN of `FNCEWS/mycomp01`. This should match the SPN previously set up for the server. It will also turn fully qualified domain names like `http://mycomp01.mydom.example.com:7001/wsi/FNCEWS35DIME/` into an SPN of `FNCEWS/mycomp01@MYDOM.EXAMPLE.COM`. These derived SPNs usually work well, but can sometimes have problems. Particularly watch out for URLs that are not DNS names, like `http://localhost:9080/wsi/FNCEWS40SOAP/` or similarly ones using IP addresses, like `http://123.45.67.89:7001/wsi/FNCEWS40SOAP` as these will yield bad SPNs.

2. The SPN is correct, but was never mapped by the `setspn` utility to the corresponding "identity" user account, `FNCEWS_mycomp01` in this example. Try running `" setspn -l FNCEWS_mycomp01 "` to list the SPNs that have been mapped to this account. Check for misspellings, such as `" _ "` or `" \ "` instead of `" / "`.
3. The SPN was mapped to more than one "identity" account. One option to check for this is to use the `LDIFDE.EXE` utility (part of Microsoft's Windows 2003 Server support tools) to dump the account information to a text file by issuing a command like `" LDIFDE -d "dc=mydom,dc=mycompany,dc=com" -f mydom.txt "`. Then open the `mydom.txt` file with a text editor and search for the SPN string, `" FNCEWS/mycomp01 "` in this example. If duplicates are found, you can remove an extra one by typing something like `"setspn -d FNCEWS/mycomp01 FNCEWS_baduser"`.
4. The SPN is not known in the client's domain. This situation can arise if the client's logged-on user domain and CE server's Windows domains are either not the same or are not in the same Windows domain "forest" where the domains implicitly trust each other. See the Cross-realm Kerberos authentication section for a work-around for Windows domains that use external or transitive two-way trusts.
5. The SPN had one of the above problems and was recently fixed. In this case the .NET framework has cached the bad result and the only way to clear this is to reboot the client system. It is a good idea to reboot the client anytime .NET reports a "The network path was not found" error.

"There are currently no logon servers available to service the login request."

This error can sometimes be caused because of connectivity problems between the client and domain controller systems for all involved domains. If the connectivity seems to be okay, this error might also mean that the time servers on the client and server are too far out of sync, particularly in cross-domain setups. Usually the servers must have times within 5 minutes of each other, but sometimes having them even 1 minute out of sync can cause problems.

"A specified login session does not exist. It may already have been terminated."

This could mean that the client workstation's logged-on user is a local account and not a domain account. Kerberos requires that the logged-on user is a domain user account.

There might be other less common problems with the client. One way to see if the message is even making it to the CE server is to turn on Kerberos Service debugging on the CE server. Do this by setting `debug=true` in the JAAS configuration for the `KrbServiceLoginModule` on WebSphere or, on WebLogic, by setting the `Debug` option to `true` for the Kerberos Service Authentication Provider. If the request is getting through to the CE server then the server's console and/or log will have additional debug lines for each request.

CE server problems

If the Kerberos problem does not seem to be on the client, turn on Kerberos Service debugging. Do this by setting the debug=true option in the JAAS configuration for the KrbServiceLoginModule on WebSphere; on WebLogic do it by setting the Debug option to true for the Kerberos Service Authentication Provider.

What follows are a few of the more common Kerberos problems. Some error messages will have a number, for example (9), that represents a standard Kerberos error code. This number can appear in either the message itself or, as is the case with WebSphere, as the minor error code for the error. Here are the problems:

"The client or server has a null key (9)"

This could indicate that the CE server has not had the AllowTgtSessionKey_fix_for_Krb5LoginModule.reg registry patch applied. This patch is necessary on certain later service packs and versions of Windows.

"KDC has no support for encryption type (14)"

"Cannot find key of appropriate type to decrypt AP REP - RC4 with HMAC"

"Cryptographic key type rc4-hmac not found"

These errors might mean that the "identity" domain user account does not have the "Use DES encryption types for this account " option set. Another possibility is that the wrong ktab utility was used (say the Sun ktab utility was used on a WebSphere system). It can also mean on WebSphere systems that the krb5.ini file does not have these settings in the [libdefaults] section:

```
default_tgs_enctypes = des-cbc-md5 des-cbc-crc
default_tkt_enctypes = des-cbc-md5 des-cbc-crc
```

"Password has expired - change password to reset (23)"

The "identity" user account's password must be changed. Also remember to run the ktab utility on the CE server to change the keytab's password if necessary.

"Pre-authentication information was invalid (24)"

The most likely error is that the user name and password in the keytab does not exactly match the user name and password of the "identity" user account given when creating the account. Notice that this match is case-sensitive for both the user name and the password. This could also mean that the keytab file was not found for the user account (in the FNCEWS_mycomp01@MYDOM.EXAMPLE.COM form), but there should also be a "could not find user in keytab" error prior to this. There is also the possibility that the system clocks are too far out of sync, see the "Clock skew too great " error.

"Integrity check on decrypted field failed (31)"

This could mean that "identity" user account mapped to the SPN which the client used does not match the user account that the CE's KrbServiceLoginModule used. A possible reason for this is that the login module's serviceAccountName is not set to the correct mapped SPN name for a cluster.

This error can also occur on a WebSphere system if the user name in the keytab does not exactly match the user name (case-sensitive) given when the "identity" user account was created. This means that the case of the user name given in the ktab command really matters!

"Request is a replay (34)"

This request has the same Kerberos token as an earlier request. This is probably because a .NET client is reusing a KerberosToken (or KerberosToken2) object rather than re-constructing it for each request. If on JBoss, then this might mean that the cacheSize option's value is too small and must be increased. For the security minded, it could also mean that intruders are trying to impersonate users using a "replay attack" (but, seriously, only consider this for a production system which has been using Kerberos for a while and has all the bugs worked out).

"Clock skew too great (37)"

This error can happen if the clocks on the client machine and the CE server are more than some number of minutes apart. Commonly this is 5 minutes, but can also be 1 minute. The fix for this is to more closely synchronize the clocks on the two machines.

"The user could not be found in the keytab"

"Key for the principal FNCEWS_mycomp01@MYDOM.EXAMPLE.COM not available in default keytab"

"No Kerberos creds in keytab for principal FNCEWS_mycomp01@MYDOM.EXAMPLE.COM"

These errors could either indicate that the "identity" user (for example, FNCEWS_mycomp01@MYDOM.EXAMPLE.COM) was not in the keytab or that the keytab file itself could not be found. Use "ktab" by itself from the command line to list the contents of the default keytab file and check for misspellings. The keytab file is, by default, the c:\Documents and Settings\
<user> \krb5_keytab file. Where <user> is the name of the account that the application server is running as. Note that if the application server is running as a service, there will probably be no corresponding <user> directory and it is best if the keytabPath="c:/my_keytab" KrbServiceLoginModule option is used to specify what and where the keytab is.

"Could not create default AuthorizationToken during propagation login"

This is a WebSphere error that probably means that the security mechanism is set to "SWAM" rather than "LTPA". See the notes above on setting up WebSphere on how to change this.

"Principal user@MYDOM.EXAMPLE.COM not found"

This error can occur if the GCD does not have a directory configuration for the MYDOM.EXAMPLE.COM Active Directory. In EM, right-click the name of the P8 domain and select "Properties" from the context menu and then click Directories.

This error can also happen in a multi-domain environment if the GCD does not have the directory configuration for a client domain set up, as each client domain must individually have its directory configuration set up.

"JAAS configuration FileNetP8KerberosService not found"

This error will occur if the FileNetP8KerberosService JAAS configuration entry has not yet been configured.

"Cannot get kdc for realm MYDOM.EXAMPLE.COM"

This error can happen if the kdc= line in the CE's krb5.ini file has not been edited correctly to reference the primary domain controller for the MYDOM.EXAMPLE.COM domain. Another possibility is that the Kerberos configuration file, krb5.ini, cannot be found by the CE as it is either not in the c:\winnt directory or the -Djava.security.krb5.conf= setting is wrong.

This can also happen in a multi-domain setup which uses external two-way trusts if the CE's krb5.ini file has not been setup to configure each client domain.

“Null key”

This error can happen on WebSphere systems if the server's SPN identity (for example, FNCEWS_msys@MYDOM.COM) cannot be found in the CE server's currently configured keytab.

It can also happen if the keytab file itself cannot be found for some reason, such as using the default keytab, but that keytab was created by one user and the app server is running as another (a particular problem if the app server is running as a Windows service). One fix for this is to specify the keytabPath option for the KrbServerLoginModule, for example, keytabPath="c:/config/keytab".

“Null name”

This error will occur on WebSphere systems if the identity user name used for Kerberos's identity account is longer than 20 characters. The default identity user name is derived by CE to be the string FNCEWS_ + hostname. Unfortunately this default name will be too long if hostname itself is 14 or more characters long. If this is the case, this can be fixed by using some other name for this identity user account and specifying that name in the serviceAccountName option for KrbServiceLoginModule. For example, serviceAccountName=FN_long_hostname_123).

Another possibility, also on WebSphere servers, is that the encryption type of the key saved in the keytab does not match the encryption type used when encrypting the Kerberos ticket. One way for this to happen is if the “identity” account does not have the “Use DES encryption types for this account” option set in the account's property dialog. (See “Step 1 - Create an Active Directory domain user account”.)

This problem might require setting the --Dcom.ibm.security.jgss.debug=all and -Dcom.ibm.security.krb5.Krb5Debug=all properties on the JVM to diagnose as described below.

There can be many other CE server Kerberos problems. If Kerberos Service debugging is not enough to isolate the problem, try turning on additional system debugging, which will give even more information. CE WebLogic and JBoss servers can specify the sun.security.krb5.debug property, usually on the Java command line with -Dsun.security.krb5.debug=true. This produces a lot of trace output on the JVM's console. There are two Java command line switches for CE WebSphere servers:

```
-Dcom.ibm.security.jgss.debug=all  
-Dcom.ibm.security.krb5.Krb5Debug=all
```

which also provide extra messages in WebSphere's log.

Authorization

When a security principal that has already been authenticated attempts to access FileNet P8 objects, Content Engine or Process Engine will attempt to retrieve that principal's user and group memberships from the directory service provider. If successful, the user or group will be authorized to carry out actions described by the access rights placed on the objects.

The topics in this section describe authorization: the various features used to manage security and apply security to objects, as well as the details of security behavior.

About access rights

What are access rights?

Most objects are independently securable, meaning they are secured by their own Access Control List, or ACL. An object's ACL is the collection of all the access rights, or ACEs, placed on it. Access rights are also called permissions.

An object's access rights determine which users can view the object and what those users can do. For example, to see a folder a user needs at least the folder's View properties permission. To add a document to the folder, the user needs the Add to Folder permission.

Access rights vary by object and control all operations on that type of object. Documents (and only documents) have permissions that let the user make new versions of the document, whereas folders (and only folders) have an Add to Folder access right, and so on.

Security identifier (SID): Embedded into each ACE is a globally unique security identifier (SID), ensuring that it can be distinguished from all other possible accounts. This SID is created by the authentication process on the application server and embedded into the account. FileNet P8 simply uses this SID and can display its value to properly authenticated users and system administrators.

Permissions and Levels: Each FileNet P8 ACE contains a set of permissions that the grantee is either allowed or denied. For example, Alice might be given permission to read a document (View Content) and look at the document's properties (View Properties). Bob might have these permissions and also have permission to add a document of a particular class (Create Instance) and delete a document (Delete). Unless these permissions are associated with Alice's ACE, she will not be able to carry out these actions on a particular document or class of documents.

In fact, each ACE has entries for all possible permissions for the particular object, with each permission set to either Allowed or Denied (or if not set at all the permission is implicitly denied). This is how Alice and Bob can have different permissions. The Create Instance permission might be marked Allow for Bob's ACE, and marked Deny for Alice's ACE. Deny settings have precedence over Allow settings. So, in Alice's case, even if she belongs to a group whose ACE indicates its members are allowed Create Instance permission, she will be denied that permission because of the Deny setting on her individual ACE.

Accounts reside in a directory server: FileNet P8 ACEs all represent accounts residing in one of the supported directory servers used for authorization. However, FileNet P8 creates and manages two special accounts, #CREATOR-OWNER and #AUTHENTICATED-USERS, which are not persisted in the authentication provider.

FileNet P8 uses LDAP, the Lightweight Directory Access Protocol, to provide accounts to CE where they become associated with FileNet P8-specific permissions and other descriptors explained in this topic. The use of LDAP ensures that account information is safely encrypted in transit between the authentication provider or directory service and CE. See Directory service provider integration for information about the supported directory servers.

Access Control Lists (ACLs): Once users are authenticated and logged onto a particular object store, their access to information, documents, workflows, etc. is controlled by security settings either on those objects or on objects that control the access to those objects. Each securable object has an Access Control List (ACL) that indicates who is granted and who is denied permissions to its properties and any associated content. An ACL can contain any number of grantees, which may be both users and groups. Each grantee's set of access permissions is stored as an access control entry (ACE).

Simply put, then, each ACL is a collection of ACEs whose purpose is to ensure that only those users and processes have the access that the system designers intend.

Understanding how ACEs and their associated permissions are inherited is an essential part of understanding and designing the security behavior for your application. Each ACE, besides indicating whether an access permission is allowed or denied, also contains information about the source of each permission. That is, a permission can either be applied directly, inherited from another object, or applied

from a security template. The source has a direct effect on the permission's subsequent inheritance behavior, even determining whether it is inheritable to some other object. The next section describes these various aspects of an ACE.

Object store administrators can view and modify an object's security using either Enterprise Manager or the Workplace or Workplace XT applications. However, each provides a different security interface with different capabilities and behavior: How Workplace applications display security shows the application interface. Enterprise Manager's security editor includes a screen shot of the following data fields:

ACE name

The name of a user or group is associated to each ACE. This is in "principal name" format, which may vary depending on the underlying authentication service. Underlying the "Name" of the ACE is its fully qualified Distinguished Name or (if using Active Directory) its Principal name. Some FileNet P8 security interfaces will display this full name.

ACE source: Default, Direct, Inherited, Template

Every ACE has a "source" which you can view in Enterprise Manager's security editor:

- Default Permissions placed on an object by the Default Instance Security ACL of its class, as well as permissions placed on a subclass by its parent class. Default ACEs are directly editable; if you do, its source type becomes Direct.
- Direct Permissions added directly to an object. Direct ACEs are directly editable.
- Inherited Permissions placed on the object by a security parent or by setting up a relationship with an object-valued property whose Security Proxy Type has been set to Inherited. Inherited ACEs are not directly editable. See Understanding security inheritance and Configure security inheritance for information.
- Template Permissions placed on the object by a security policy. Template ACEs are not directly editable and do not appear on classes. Rather, a document, folder, or custom object class may have a default Security policy which will pass template ACEs to the instances of the class, if all the conditions for the template apply. (See Security policies.)

Workplace and Workplace XT use the term "System" to identify Inherited and Template source types.

ACE security levels

A security "level" is a predefined collection of rights which simplify applying and administering individual rights. For example, the Full Control level for a document includes all possible document rights, whereas the Major Versioning level includes only those rights required to allow the user to perform the various tasks required to create a major version of a document. Documents, folders, custom objects, choice lists, and other securable objects all have their own predefined levels appropriate to the type of object. Levels can be viewed and assigned using the object's security editor. Applications can be designed to expose only the levels to users, or they can allow users to choose to assign levels or the individual rights. Enterprise Manager's security editor includes a "custom" level which is displayed if individual rights have been set that are different from the pre-designed rights-to-level mappings.

The topic in the Mapping security levels to individual access rights provides specific definitions of security levels.

Allow/Deny and order of evaluation

Each ACE has one access type: either Allow or Deny. When evaluating the access granted by a particular ACL, the current system applies ACEs in the following order of precedence (higher on the list takes precedence over lower):

ACE source and type	Display
Direct/Default Deny	Type <input type="radio"/> Allow <input checked="" type="radio"/> Deny
Direct/Default Allow	Type <input checked="" type="radio"/> Allow <input type="radio"/> Deny
Template Deny	Type <input type="radio"/> Allow <input checked="" type="radio"/> Deny
Template Allow	Type <input checked="" type="radio"/> Allow <input type="radio"/> Deny
Inherited Deny	Type <input type="radio"/> Allow <input checked="" type="radio"/> Deny
Inherited Allow	Type <input checked="" type="radio"/> Allow <input type="radio"/> Deny

You cannot remove or change an inherited access right, but you can override one by directly allowing or denying an access right. To edit an inherited access right, the administrator must modify the parent that is the source of the inherited access right.

Because Deny has precedence over Allow within each category (for example, a Template Deny takes precedence over a Template Allow), if you explicitly deny an access right to a group and explicitly allow it to a member of that group, the access right will be denied to the member.

Thus, if an ACL contained two ACEs that were identical in every respect except that one was an Inherited Deny and the other a Direct Allow, the Direct Allow would take precedence, with the result that the user would be allowed the ACE.

Default security

This topic answers the question, "What is the security behavior if administrators and users do nothing to change it?"

Object store administrative groups

Members of the groups added to the Object Store Wizard as object store administrators have Full Control of object stores and their contents, which means that while using Enterprise Manager they can perform any valid action on any item. See the Reference section for the specific actions.

End users

When creating an object store, the administrator selects one or more groups that will have basic, nonadministrative access rights. For example, if the administrator selects the Domain Users group as the nonadministrative group when creating an object store, users of the Workplace and Workplace XT applications can:

- Add folders at the top level of the object store.

NOTE A new folder acquires its initial security from the Folder class, which grants Full Control to the folder creator (also called Owner Control), Full Control to members of the object store administrative groups, but only View Properties access to Domain Users. A user must have Add to Folder access rights to put documents in the folder. This means that, by default, users can create top-level folders and add items to their own folders. However, users cannot add items to the folders created by other users.

- Add documents (with Add to Folder access rights to the selected folder).
- View the properties and content of all folders.
- View the properties and content of all documents.
- Run the designer applications but not those that are workflow-related.

Other access rights are not set one way or the other, which means they are implicitly denied to members of nonadministrative groups.

NOTE For any given access right (for example, View Properties), an access right has three possible settings: Allow, Deny, or neither. If an access right is neither explicitly allowed nor explicitly denied, it is "implicitly denied."

Users of other client applications (such as WebDAV and Application Integration for Office) are subject to the same security as application users but typically cannot perform as many operations.

Security for integrated components and third-party products

FileNet P8 Platform is integrated with several FileNet and third-party software products. The topics in this section briefly describe the security features of some of those products and the requirements when used in conjunction with FileNet P8 Platform.

FileNet P8 security is enforced for all operations performed via an integrated application, although the details of implementation will vary depending on the application.

FileNet does not support third-party operating systems, databases, or other software, and may or may not certify updates, fixes, or service packs released by other software vendors. For any incident involving third-party software or for any updates to these products, customers must obtain information, support, and updates from the third-party software vendor.

FileNet may inform you (and may on occasion make a patch available) when a third-party patch is required for correct functioning of FileNet software. When another vendor's patch interferes with the functioning of FileNet software, we will resolve the problem or advise you not to apply that particular patch until the issue has been resolved.

Browsers

For each Workplace or Workplace XT session, the application creates a cookie on the client containing the session key. The cookie persists in the client memory cache for the duration of the session and is cleared when either the session or the browser is closed on the client. Connectivity problems may occur on browsers configured to refuse cookies.

Database security

FileNet P8 Platform security is enforced for database information accessed through Workplace, integrated applications, and administrative applications. Access to the GCD database is configured during initial installation. Object store databases are secured in the same way. See "Specify FileNet P8 Accounts" in the Installation Guide.

Security for Autonomy K2 server

Content Engine software logs on to Verity using the credentials of the Autonomy K2 server account.

Content Engine secures Autonomy K2 (Verity) collections by placing an ACL on each collection that allows access only to the Autonomy K2 server account. No user will be able to use external Verity tools to access the collection unless an appropriate user and password is provided.

Content Engine does not utilize or depend on Verity's own internal security features like securing its servers, collections, and objects inside collections.

For more information, see the entries for the Autonomy K2 server accounts in Users and Groups and the description of K2 access to file storage areas in Storage Area Security.

Security for fixed content providers

FileNet P8 supports several fixed content providers.

FileNet Content Federation Services Server for Image Services

During the creation of an Image Services (IS) fixed content device for Content Federation Services, you must provide Content Engine with the ID and password of an Image Services user that you will use to administer CFS. For information about this CFS-IS Administrative User, see:

- Users and Groups
- FileNet P8 Content Federation Services Server for Image Services Guidelines (see especially the section Create a CFS-IS Administrative User)
- Also see the Guidelines document for other security information.

EMC Centera

See About EMC Centera in Help for Content Engine Administration for information about Centera server security.

SnapLock

See About SnapLock in Help for Content Engine Administration for information about SnapLock security.

Security for FileNet P8 eForms

FileNet P8 eForms is an expansion product that layers on top of the authentication mechanisms in place for Workplace and Workplace XT applications. eForms appears as an element of each application, it inherits its session from the application, and the entry points that touch eForms are part of the application deployment.

The Form Data, Form Policy, Form Template classes (all subclasses to the document class) are not available for creation unless the expansion product is installed and enabled. Once created, they are regular document objects that receive default permissions from their class like any other object, and could then take advantage of explicit, inherited, and security policy security as needed.

Security for Records Manager

The FileNet P8 Records Manager application security uses the following features:

- Markings
- Cross-object store references
- Security parent folders

Records Manager is an extension to the Workplace and Workplace XT applications, and uses the application's security user interface. See [Manage Security](#) in the online help for Workplace.

Mapping security levels to individual access rights

FileNet P8 security levels are logically defined sets of individual permissions. They are intended to simplify the process of assigning permissions to objects. However, there are a few things to keep in mind:

Whereas Enterprise Manager displays both levels and the individual rights that comprise them, some applications, including Workplace, display only the level.

Applications may name the level differently than Enterprise Manager.

The topic About access rights - Levels section explains the relationship between security levels and individual access rights. The topics in this section list the access rights that make up the security levels for each of the main securable objects.

FileNet P8 domain root security levels

The table below maps domain root security levels to the rights from which they are comprised. For example, the Use stores and services level includes only View all properties, while Full Control includes all rights available to this object. In Enterprise Manager, you can view and modify access rights and permissions on the Security page of the FileNet P8 domain root property sheet.

In Enterprise Manager, the domain root's security page displays:

- GCD administrators, users and groups who have been granted Full Control.
- #AUTHENTICATED-USERS who by default receive Use stores and services.

Rights	Security levels	
	The security level listed at the top of the column is made up of the Rights marked with ✓.	
	Full Control	Use stores and services <Default>
View all properties	✓	✓
Modify all properties	✓	
Create new stores	✓	
Delete stores	✓	
Read permissions	✓	
Modify permissions	✓	

Object store security levels

The table below maps object store security levels to the rights from which they are comprised. For example, the Use object store level includes the rights Connect to store, Create new objects, Modify existing objects, and Delete objects. Access rights to an object store are limited to only that object store and do not extend to other object stores in the FileNet P8 domain.

In Enterprise Manager, the object store's security page lists the users and groups that were added while running the object store wizard as follows:

Users and groups added as "users" receive the Use object store level.

Users and groups added as "administrators" get Full Control and are therefore object store administrators.

Rights	Security levels		
	The security level listed at the top of the column is made up of the Rights marked with ✓.		
	Full Control	Use object store	View object store <Default>
Connect to store	✓	✓	✓
Create new objects	✓	✓	
Modify existing objects	✓	✓	
Delete objects	✓	✓	
Set owner of any object	✓		
Read permissions	✓		
Modify permissions	✓		
Modify certain system properties			

See Object store access rights for more information.

Class definition security levels

The table below maps class definition security levels to the rights from which they are comprised. For example, the View Properties level includes the rights View all properties and Read permissions.

In Enterprise Manager, you can view and modify access rights and permissions on the Security page of a class definition's property sheet.

Rights	Security levels			
	The security level listed at the top of the column is made up of the Rights marked with ✓.			
	Full Control	Modify properties	Link	View properties <Default>
View all properties	✓	✓	✓	✓
Modify all properties	✓	✓		
Link	✓	✓	✓	
Create instance	✓	✓		
Create subclass	✓	✓		
Delete	✓			
Read permissions	✓	✓	✓	✓
Modify permissions	✓			
Modify Owner	✓			

Folder security levels

The table below maps folder security levels to security rights. For example, the View Properties level includes the rights View all properties and Read permissions.

In Enterprise Manager, you can view and edit these permissions on the Security page of a folder's property sheet or the Default Instance Security page of the folder class.

Enterprise Manager, you can view and edit these permissions on the Security page of a folder's property sheet or the Default Instance Security page of the folder class.

Rights	Security levels			
	The security level listed at the top of the column is made up of the Rights marked with ✓.			
	Full Control	Modify properties	Add to folder	View properties <Default>
View all properties	✓	✓	✓	✓
Modify all properties	✓	✓		
Reserved12 *	✓			
Reserved13 *	✓			
File in Folder / Annotate	✓	✓	✓	
Unfile from folder	✓	✓	✓	
Create Instance	✓	✓		
Create subfolder	✓	✓		
Delete	✓			
Read permissions	✓	✓	✓	✓
Modify permissions	✓			
Modify owner	✓			
Minor versioning (Inherit only)	✓	✓		
Major versioning ** (Inherit only)	✓	✓		
View content ** (Inherit only)	✓	✓		
Change state ** (Inherit only)	✓	✓		
Publish ** (Inherit only)	✓	✓		

* Deprecated permissions.

** Also used by Publishing.

Document security levels

The table below maps document security levels to the rights from which they are comprised. For example, the View Properties level includes the rights View all properties and Read permissions.

In Enterprise Manager, you can view and edit these permissions on the Security page of a document's property sheet or the Default Instance Security page of the document class.

Rights	Security levels						
	The security level in the column is made up of the Rights marked with ✓.						
	Full Control	Minor Versioning	Major Versioning	Modify Properties	View properties	Publish	View content <Default>
View all properties	✓	✓	✓	✓	✓	✓	✓
Modify all properties	✓	✓	✓	✓			
Reserved12 *	✓						
Reserved13 *	✓						
View content	✓	✓	✓	✓		✓	✓
Link a document/Annotate	✓	✓	✓	✓		✓	
Publish**	✓					✓	
Create Instance	✓	✓	✓	✓			
Change State	✓	✓	✓	✓			
Minor versioning	✓	✓	✓				
Major versioning	✓		✓				
Delete	✓						
Read permissions	✓	✓	✓	✓	✓	✓	✓
Modify permissions	✓						
Modify owner	✓						
Unlink document		✓	✓	✓		✓	
Create subfolder (Inherit only)							

* Deprecated permissions.

** Publish is defined in Workplace to include Modify permissions.

Custom object security levels

The table below maps custom object security levels to the rights from which they are comprised. For example, the View Properties level includes the rights View all properties and Read permissions.

In Enterprise Manager, you can view and edit these permissions on the Security page of a custom object's property sheet or the Default Instance Security page of the custom object class.

Rights	Security levels			
	The security level listed at the top of the column is made up of the Rights marked with ✓.			
	Full Control	Modify properties	Link	View properties <Default>
View all properties	✓	✓	✓	✓
Modify all properties	✓	✓		
Reserved12 *	✓			
Reserved13 *	✓			
Link / Annotate	✓	✓	✓	
Create instance	✓	✓		
Delete	✓			
Read permissions	✓	✓	✓	✓
Modify permissions	✓			
Modify Owner	✓			
Minor versioning (Inherit only)				
Major versioning (Inherit only) **				
View content (Inherit only) **				
Change state (Inherit only) **				
Publish (Inherit only) **				
Create subfolder (Inherit only)				
Unfile folder from folder (Inherit only)				

* Deprecated permissions.

Application Integration users do not see custom objects. Workplace users see custom objects if enabled by a site preference, but a custom object's Link access right is hidden in Workplace.

Other object security levels

The table below maps security levels to the rights from which they are comprised for objects that are not otherwise defined in this help section. Examples of such classes are events, subscriptions, and lifecycles. Note that the Link access right allows the user to associate the object with another object; it is hidden in the Workplace and Workplace XT applications.

In Enterprise Manager, you can view and modify access rights and permissions on the Security page of the object's class definition's property sheet.

Rights	Security levels			
	The security level listed at the top of the column is made up of the Rights marked with ✓.			
	Full Control	Modify properties	Link	View properties <Default>
View all properties	✓	✓	✓	✓
Modify all properties	✓	✓		
Link	✓	✓	✓	
Create instance	✓	✓		
Create subclass	✓	✓		
Delete	✓			
Read permissions	✓	✓	✓	✓
Modify permissions	✓			
Modify Owner	✓			

Markings

This topic explains what markings are and how they affect the effective security on objects. Primarily designed for use by the FileNet P8 Records Manager application, markings are available to any application that needs the kind of property-based layer of security that markings provide.

- Overview
- Limitations
- Marking security: Add, Remove, Use
- Effect of Copy to reservation
- Constraint Mask
- Allow, Deny permissions
- Hierarchical and non-hierarchical marking sets
- FileNet P8 domain resource
- Marking Set TTL (caching)
- Active markings
- Marking administration (create, modify, remove)
- Not available with Choice Lists

See Content Engine Help for information on how to use Enterprise Manager to administer markings and marking sets.

Overview

Markings allow access to objects to be controlled based on specific property values. When a marking is applied to an object, the resulting access permissions for the object are a combination of the settings of its original access permissions and the settings of the marking's Constraint Mask for each marking that is applied to it. The result of this combination is the effective security mask.

In general terms the way the markings works is:

1. A marking set is defined, containing several possible values called markings.
2. Each marking value contains a set of access permissions which define who can assign that specific value to an object property, who can modify or remove that specific value, and, once it is assigned, who will have access to the object it is assigned to.
3. The marking set is assigned to a property definition on a class such that the value of that property on instances of the class must be one of the markings defined by the marking set.
4. Values can only be assigned by users authorized by the associated marking and access to the object is restricted based on the marking once it is applied.

Markings do not replace conventional access permissions on an object, but rather are co-equal with them in determining access rights. In other words, if an object has one or more markings applied to it in addition to one or more permissions in its permissions collection (ACL), then access to that object is only granted if it is granted by the permissions and by the markings. Another way to think about how this works is:

1. A user or process tries to access an object.
2. First the Content Engine resolves the object's ACL to determine who can access the object and what those users can do.

- Then it computes the markings applied to the object to see which users to stop (defined by the marking's Security list) and what they should be stopped from doing (defined by the marking's constraint mask).

You can have multiple properties assigned to a single class with marking sets associated, and they will all be used to determine the final access to the object. The collection of all markings being actually applied to a particular object is displayed by the Enterprise Manager as the object's "active markings" (see below).

Limitations

The number or size of markings in a single marking set is limited by available system memory. To perform an access check on a marked object, the entire marking set and all its markings must be loaded into memory. This is not going to work if there are millions of markings. For this reason, FileNet recommends that you limit the number of markings in a marking set to no more than 100.

Another limiting factor would be one that occurs with any UI attempting to display extremely long lists of choices, but this kind of application design problem is not inherent to markings.

Marking security: Add, Remove, Use

Marking security consists of the following permissions:

Add marking and Remove marking

A user with Add rights to a marking can set the property value associated with the marking, if it has not been set. Only those markings to which the user has Add rights will show in the list of marking values available to be set in a property. A user with Remove rights to a marking can remove the marking value.

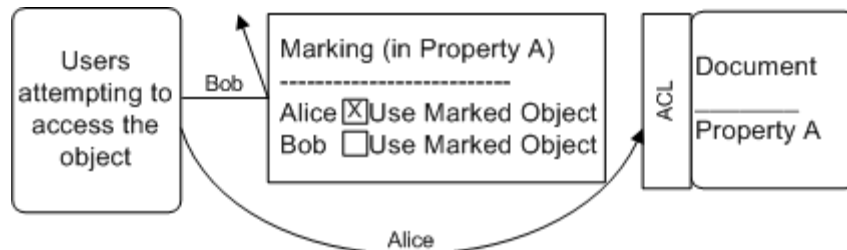
For example:

- A Document has a property associated with a marking set. No value has been specified for the property.
- The marking set has markings Red, Blue, and Green.
- Alice has Use rights to Red; Use and Add rights to Blue and Use, Add and Remove rights to Green.

When Alice views the Document properties, she can set the property value to Blue or Green but not Red. If the property was set to Green, she could alter it to be Blue. If the property was set to Blue, Alice would be unable to alter the property's value.

Use Marked Objects

Use right determines whether the presence of the marking on an object constrains access to that object. If the user has **Use** right to the marking, access to the object will not be constrained.



In this example, Alice has the Use Marked Objects access right which lets her bypass the marking. Her access to the object will be evaluated by the object's ACL. Bob does not have Use Marked Objects and therefore will neither see nor have access to the object, regardless of any permissions the object's ACL might grant him.

Markings and marking sets are Content Engine objects, each with a class description:

- Markings are objects that combine metadata behavior with access control behavior in a way that allows an object's access control to change by changing a property value.
- Marking sets are containers for markings. Marking sets are associated with a Property Template which can then be used to add a property to one or more classes.

Effect of Copy to reservation

If the string property containing a marking has kept the default Copy to Reservation setting of True, then a user who has Use Marked Objects but does not have the Add Marking permission will not be able to check out a document, even if the user has Full Control of the document itself.

This is by design, since to checkout a document creates a new Reservation document which must have the marking on it due to its property's Copy to Reservation value of True. This requires the marking to be "Added" to the new document, and since the user in this scenario does not have the Add Marking permission, the checkout will be prevented.

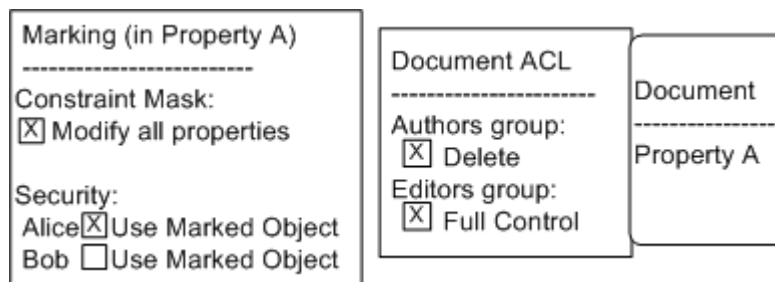
Constraint Mask

By default, all the rights are checked, meaning all constraints are masked and only those that have the Use Marked Objects access rights on the marking will be able to view and access the object.

When one of the rights in the constraint mask is unchecked, it indicates that users with this privilege on that object are allowed through the marking restriction even if they do not have the Use Marked Objects access right on the marking. In this way, the constraint mask can be used to design more granular control at the marking level.

Here are some examples to illustrate the security behavior of the constraint mask:

- If the constraint mask has all permissions selected (turned on), and if Alice does not have Use Marked Objects rights to that marking, then Alice will have no access and will not see the object, even if she has Full Control on the object's ACL.
- If the constraint mask has all permissions selected (turned on) except View All Properties and Delete which are deselected (turned off), and if Bob does not have Use Marked Objects rights to that marking, then Bob can see and delete the object, provided that he is granted those permissions on the object's ACL.
- If the constraint mask has all permissions deselected (turned off), and even if Carol does not have Use Marked Objects rights to the marking, then Carol can do everything to that object granted her by the object's ACL. (Deselecting all permissions in the constraint mask effectively renders the marking inactive.)
- If Dave has Use Marked Objects rights to the marking, the constraint mask has no effect on his resulting access. His access will be solely determined by the object's ACL.



In this graphic:

- Alice and Bob are members of the Authors group. The only property selected in the constraint mask is 'Modify all properties'. The ACL on the document gives Authors the 'Delete' permission.

- Alice has the 'Use Marked Objects' right, and therefore the marking's constraint mask does not apply. She can delete the document (and anything else that the ACL grants to Authors).
- Bob does not have the 'Use Marked Object', and therefore the marking's constraint mask applies to him. The constraint mask specifies 'Modify all properties' and that means that Bob does not have the 'Modify all properties' right on any object to which this marking is applied, even if it is explicitly granted to him by the ACL. The document has not granted the 'Modify all properties' right to Bob in the first place since he is not a member of the Editors group and therefore the marking has no impact on him. Also, Bob can delete the document (regardless of whether or not he has the 'Use marked objects' right) since the marking constraint mask does not affect the Delete right, and because it has been granted to him by virtue of his membership in the Authors group.
- Alice and Bob are not members of the Editors group. Because the Editors group is not listed on the marking, Editors do not have the 'Modify all properties' right despite being granted 'Full Control' by the document itself. The reason for this is the constraint mask in the example only specifies the 'Modify all properties' right. As a result, either having or not having the 'Use marked object' right on the marking can only affect the 'Modify all properties' right on any given object marked with this marking.

Allow, Deny permissions

Each access control entry listed on a marking value's security page is marked either Allow or Deny.

Allow

Allow is the default setting for each new security added to a marking's security list. It is also the most common way to set up marking security behavior. Unless clearly stated, this topic describes Allow security types.

Deny

Typically markings are used to determine who will be denied access to evaluate the security rights of the object. Users who have the Use Marked Objects access right will not be limited by the constraint mask of the marking. However, an administrator may set up deny rights on the marking which will override any allow access otherwise granted to the marking. For example:

1. The security on a document grants #AUTHENTICATED-USERS full control access.
2. The document has a single-valued property associated with a marking set with possible values of Chicago, New York, and Boston.
3. The property value is set to Boston.
4. The Boston marking has a constraint mask of full control allow (all permissions selected).
5. The group Everyone_Boston has Use/Allow rights to the Boston marking.
6. The Sales group has Use/Deny rights to the Boston marking.

In this scenario:

- Users who are not members of Everyone_Boston cannot access the document.
- Users who are members Everyone_Boston can access the document, unless they are also members of Sales.
- Users who are members of Everyone_Boston and Sales cannot access the document. The deny setting on the marking overrides the allow setting and ensures that no one in Sales sees the document even if they are in the Boston office.

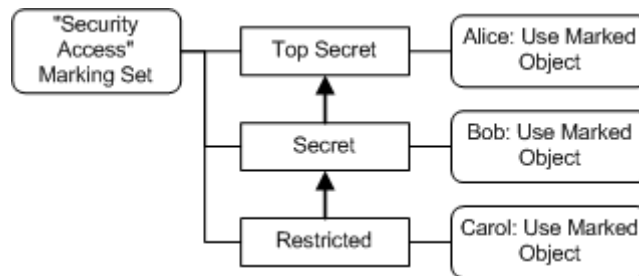
See the next section for how the Allow and Deny permissions are applied to hierarchical marking sets.

Hierarchical and Non-hierarchical

There are two types of marking sets: hierarchical and non-hierarchical:

- Non-hierarchical (also called "list") marking sets contain one or more markings, each independent of all others. This type of marking set can be assigned to either single or multi-valued properties.
- Hierarchical marking sets arrange markings in a simple, single branch series. Each hierarchical marking has a superior marking in the same set, except for the first, or top marking. Hierarchical marking sets support an order of precedence when evaluating access rights. A marking in a hierarchical marking set explicitly grants access rights to some set of security principals. In addition, it also implicitly grants the same rights for all markings that are inferior to (or below) it in the hierarchy. Hierarchical marking sets can be assigned only to single-valued properties.

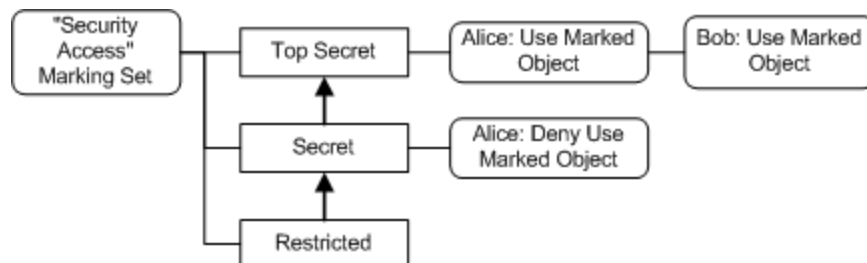
In the following, the user Alice has been granted the Use Marked Objects right by the Top Secret marking. Since this is a hierarchical marking set, the effect of this is that she not only has the Use Marked Objects right for the Top Secret marking, she also implicitly has it for the Secret and the Restricted markings as well. This means that for Alice, objects that are marked with either the Restricted or Secret markings behave as if they are marked with the Top Secret marking. Similarly, Bob has implicit Use Marked Objects right to the Restricted marking, because it is lower in the hierarchy than Secret.



There is an exception to the rule that permissions granted on superior markings are implicitly granted on inferior markings. Recall that there are two types of access permissions:

- Allow permissions (the more common of the two)
- Deny permissions

Deny permissions always take precedence over allow permissions. The way this works for hierarchical marking sets is that deny permissions on inferior markings always take precedence over allow permissions on superior markings. This behavior is illustrated in the example below:



In this example Bob is implicitly granted access to objects marked Secret or Restricted as well as being explicitly granted access to objects marked Top Secret due to the fact that he is granted the Use Marked Objects access right on the Top Secret marking which is superior to both Secret and Restricted.

Alice, on the other hand, is denied access to objects marked Top Secret or Secret but is granted access to objects marked Restricted. The reason for the difference is that Alice is explicitly denied the Use Marked Objects right by the Secret marking. Since Deny permissions are implicitly present on all superior markings, it is implicitly present on the Top Secret marking. She is also explicitly allowed access by the

Top Secret marking, but since Deny permissions take precedence over Allow permissions, she is still denied access to objects marked with Top Secret. What may be less clear is why she is granted access to objects marked with Restricted.

The reason is because she implicitly obtains access rights to objects marked Restricted by being granted Use Marked Objects rights by the Top Secret marking which is superior to Restricted. The Deny permission on the Secret marking has no effect on the Restricted marking because the Restricted marking is inferior to the Secret marking.

From this example three general rules can be observed:

1. Allow permissions affect markings downward in the hierarchy; that is, an Allow Permission placed on a superior marking is implicitly present on inferior markings.
2. Deny permissions affect markings upward in the hierarchy; that is, a Deny Permission placed on an inferior marking is implicitly present on superior markings.
3. Deny permissions take precedence over Allow permissions.

FileNet P8 Domain resource

Markings and marking sets are persisted in the FileNet P8 domain resource, the GCD. This gives them FileNet P8 domain-wide scope, that is, they are available and have the same meaning across all Object Stores in a FileNet P8 domain served by a common GCD. The marking-enabled property templates and the actual properties based on these templates are, however, specific to the object store in which the property template was created.

Marking Set TTL (caching)

Modifications to markings or marking sets are subject to the Marking Set Cache Entry TTL setting which affects how often the marking set cache is updated on the server and the current Enterprise Manager machine.

However, the marking set cache is updated whenever any change or addition is made to markings or marking sets. Therefore, the cache is most likely up-to-date by the time the MarkingSetsTTL forces a refresh of the cache.

Active markings

"Active markings" is the term the Enterprise Manager uses in its security editor on its Active Markings/Owner button. You will see this button text on object instances whether or not there are actually active markings applied to the object. This button will just say "Owner" for those objects that cannot have markings applied, which include all class definitions.

Marking administration (create, modify, remove)

The Enterprise Manager is your tool for creating and applying marking sets. Applications can also utilize one of FileNet P8's APIs. Here are some general rules governing administration:

Renaming a marking set

Renaming a marking set after it has been persisted is not allowed. The marking set name can be modified only during the creation of markings. Once it has been persisted, the marking set name is read-only.

Changing the marking value

The value of a marking can be modified even when properties exist which use that marking value. Existing objects that have property values set to the old marking value will not be updated to reflect the new marking value and, therefore, will need to be updated manually. If the property value is not updated, the marking security on those objects will no longer have any effect.

Modifying the constraint mask and security

You can change the marking's constraint mask and security any time without affecting existing properties.

Deleting a Marking Set

Marking sets that are referenced by at least one property template cannot be deleted. The marking set can be deleted if it is not associated with any property templates or after any associated property templates have been deleted.

Removing a marking

Individual markings can be deleted from a marking set at anytime, even if a property value is set to that marking.

Export issues

Since P8 domain resources contained in the GCD are not exportable, markings are not directly deployable to another P8 domain.

Not available with choice lists

Markings cannot be used in conjunction with choice lists.

Object access rights and security

The topics in this section describe the access rights that apply to securable objects and also provide some security details.

Securable objects

An independently securable object has its own Access Control List (ACL) that specifies its security and ensures that access rights are checked each time a user tries to access it. A dependently securable object depends on some other object as its security parent.

The important securable objects are:

- FileNet P8 domain (Domain Root)

- Object stores

- Classes

- Documents plus several subclasses of the document class:

 - Publish templates

 - Entry templates

 - Publication documents

 - Stored searches

 - Search templates

 - Workflow definitions

 - Workflow queues, rosters, and logs

- Folders

- Custom objects

- Events and subscriptions

- Lifecycle policies and actions

- Security policies

- Annotations

- Choice list class

- Property template class

The following objects are dependently securable, meaning that their security is provided by, and is the same as, the object that is their security parent:

- Content elements have the same security as their document.

- A property assigned to a securable object has the same security as that object.

- The individual choices in a choice list have the same security as the object that the choice list is assigned to.

- A lifecycle state in a lifecycle policy

Object store access rights

When running the Object Store wizard, you specify the users and groups that should be object store administrators and those who should have non-administrative access rights. You can view and modify these security assignments any time while running Enterprise Manager.

With one exception, administrative users and groups get Full Control on the object store ACL and likewise on all security ACLs of all securable objects. Note that this does not include the permission to create object stores, file storage areas, content cache areas, and related actions like deleting and moving. These permissions belong only to the users and groups who were specified as GCD administrators during the initial installation and creation of the FileNet P8 domain. A user or group can, of course, belong to both.

The exception mentioned above is the permission Modify certain system properties which determines which users can set certain system properties (Creator, DateCreated, LastModifier, DateLastModified) that are normally system only. Users and groups who will be running system level tools (like import and migration tools) may need access to this permission.

Non-administrative users and groups get the following security levels:

- Use object store on the object store ACL.
- Modify Properties on the root folder (to a Workplace or Workplace XT user, the root folder represents the object store).
- View Properties and Create Instance on all classes.
- A Custom level on the Default Instance Security of document classes that includes View All Properties and Create Instance.

See the Reference section for more information about these security levels.

Relationship of object store permissions to permissions on objects contained by the object store

Several permissions that appear on the Security tab of each object store's property sheet have a hierarchical relationship to other permissions on classes and objects contained in that object store:

- If a user or group is granted rights to "Delete objects", "Create new objects", or "Modify existing objects" on the Security tab of the object store, and if the user or group also has the right to delete or modify on the Security tab of the actual object instance as well as the "Create instance" permission on the Security tab of the object's class (for example, a document class), then the user or group can delete/modify/create the objects based on these classes.
- If a user or group is allowed these permissions on the object store, but does not have the delete/modify/create permissions on the object instance or its class, then the user or group cannot delete/modify/create the object.
- If a user or group is denied these permissions on the object store, then the user or group cannot delete/modify/create the objects even if the object's instance or class gives the user or group these permissions.

Workplace and Workplace XT

To sign in, a user must have at least View Properties access rights to the object store that contains the user and site preference files.

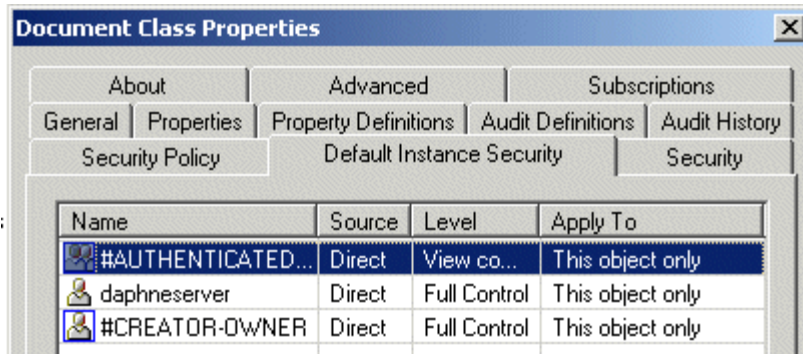
Users see all object stores configured through Enterprise Manager but must have View Properties access rights to the root folder in order to open an object store.

Class access rights

The Enterprise Manager displays all available classes in its Document Class and Other Classes nodes. The administrator can define any number of new subclasses of most of these classes and can edit the access rights of any class.

Class security tabs

When you use the Enterprise Manager to display the properties of a document class, you see three security-related tabs:



- The Security Policy tab displays the name of the document class' default security policy, if there is one; lets you change the default security policy; and lets you run the security policy wizard to create a new policy.
- The Default Instance Security tab displays the access rights that will be applied to each new document (the first version) based on the class. The #CREATOR-OWNER (the user who creates the object using this class) is granted Full Control by default.
- The Security tab displays the access rights of the document class itself. By default users are granted Create instance so they can create new instances of the class (in this case new documents), but are not granted Create subclass, since subclassing is by default reserved to system administrators.

Classes receive default security from their parent class

The Content Engine provides the Document Class which you can modify and use as the start of your own hierarchy of document classes. For example, you could create this hierarchy of document classes:

- Document Class** (predefined, can be modified)
- Tax Bills** (subclass of Document Class)
- Property Tax Bills** (subclass of Tax Bills)
- Estate Tax Bills** (subclass of Tax Bills)

Each document class receives its default security from its document class parent. Subsequent updates to a parent class propagate to the child classes.

New objects acquire initial security from a class

The Default Instance Security collection of permissions on a document class provides the initial security for a new document based on that class. Similarly with folders and every other class that allows you to create "instances" of it. This initial security is not treated as Inherited and is therefore editable. Modifying a class does not affect the security of existing objects based on that class.

See also Class definition security levels.

Folder access rights

Folders have the following security characteristics:

- They are instances of the folder class and therefore obtain default security from their class like all other objects.
- Each folder is independently secured by its own ACL.
- Folders pass security to their directly contained subfolders, if the subfolder has set its "Inherit parent permissions" property to True, and if the parent folder has inheritable permissions.
- A folder can become the security parent of the documents and custom objects it contains, if both the folder and the contained objects have been properly configured.
- Folders contain several permissions marked "(Inherit only)" that will be inherited by the documents contained in that folder if there is a properly configured security parent relationship between the folder and the documents it contains.
- A folder can be associated with a security policy, but can only use Application security templates, which require custom programming to apply the template to the folder.

Folder access rights

Full Control	Modify Properties	Add to Folder	View Properties
Includes all access rights and permits all operations including: <ul style="list-style-type: none"> • Modify security • Change owner • Delete the folder 	Create subfolders; modify folder properties	Add documents and custom objects to the folder	View the folder properties and security; open the folder

Default security

The new folder acquires its initial security, with source type of "Default", from the Default Instance Security page of its class. By default, the user who creates the folder gets Full Control.

Modifying the security of a particular folder class has no effect on the security of existing folders that are instances of that class. By contrast, modifying a folder's security may affect subfolders. Specifically, a child subfolder inherits only access rights from the parent folder if its "Inherit parent permissions" property is set to True and the parent has permissions that are set to be inheritable.

Folder behavior using the Workplace or Workplace XT applications

When a user opens an object store using the Workplace or Workplace XT applications, the object store's root folder displays only those items the user is allowed to see. Users cannot add documents to the root folder but must select another folder. However, using the designer Java applets, users can save stored searches, search templates, workflow definitions, and publish templates to the root folder of an object store.

When a user deletes a folder using the Workplace or Workplace XT applications, the folder gets deleted without deleting the contained documents. Enterprise Manager offers additional options for deleting contained documents when deleting folders.

The application's Preferences folder, which contains the various site and user preferences files, is visible in Enterprise Manager but is not visible using the Workplace or Workplace XT applications. This is configured using the IsHiddenContainer property.

Document access rights

Documents have the following security characteristics:

- They are instances of the document class and therefore when first created they obtain initial security from the Default Instance Security ACL of their class (like all objects do by default). The first document version passes the same permissions to subsequent versions, all of which have source type of Default.
- Each document version is independently secured by its own ACL.
- Documents are the security parent of any annotations associated with them.
- A document can have a folder as a security parent from which it will receive "Inherit" permissions.
- A document can be associated with a security policy from which it will receive "Template" permissions. Security policies can be configured to place security on documents as they pass through various versioning states.

See also the Documents section of Help for Content Engine Administration. For security information about compound documents, see Compound document security. and About compound documents in Help for Content Engine Administration.

Access rights defined

Full Control	Minor Versioning	Major Versioning	Modify Properties	View Properties	Publish	View Content
All access rights listed in the columns to the right and: <ul style="list-style-type: none"> • Modify security • Change owner • Delete the document 	<ul style="list-style-type: none"> • Check out a document • Check in a minor version • Cancel a check out 	<ul style="list-style-type: none"> • Change state (promote and demote) • Check out a document • Check in a major version • Cancel a check out 	<ul style="list-style-type: none"> • Modify property values 	<ul style="list-style-type: none"> • View properties and security 	<ul style="list-style-type: none"> • Publish the document 	<ul style="list-style-type: none"> • View content

NOTE The Unlink document access right is not included in any of EM's predefined levels. It can be assigned directly.

NOTE The Unlink document access right is not included in any of EM's predefined levels. It can be assigned directly.

Initial security acquired from the document class

A document acquires its initial default security from its document class and, if the application's site preference allows it, the application user can modify the security. By default, the user who creates a document gets **Full Control** (also called **Owner Control**).

Modifying document class security has no effect on the security of existing documents.

Content element security

The document's content elements are not independently securable but can only be accessed through their document version. In effect, a document's content has the same security as the document. In Enterprise Manager, a document's security information appears on the Security page of its property sheet. In Workplace or Workplace XT, a user can display a document's Info page to view security information and to assign a security policy.

Browsing vs. searching

To navigate to a document by browsing, a user must have access to all folders and subfolders in the path to the document. By contrast, a search template checks only the document's security even if the search specifies a folder. This means that searches may allow a user to find documents that could not be found through browsing.

Compound document security

Component Relationship objects define the structure of a compound document and specify rules for binding from the parent to a child component. Component Relationship objects are not independently securable, and inherit the security of their parent document. This parent-based style of security means that users with view rights to a child component document are not guaranteed view rights to all component relationship objects that may reference them.

Required access rights for parent and child components

A user needs the access right **Link a document / Annotate** on a parent component document to be allowed to create component relationship objects that reference the document as a parent component. Also, **Unlink document** or **Delete** rights on the parent component document are required to delete a component relationship object, and **Modify all properties** rights on the parent component document are required to modify the properties of a component relationship object.

View all properties rights are required on a child component document to assign the document to the **Child Component** object-valued property of a component relationship object.

Component Relationship Actions and Parent Component Requirements

Component relationship action	Requirements	Comments
Create	<p>User must have Link a document / Annotate rights on the parent component to create component relationship object that references the parent.</p> <p>User must have View all properties rights on the child component to create component relationship object that references the child.</p>	<p>In addition to security requirements, a parent document <code>CompoundDocumentState</code> property must be set to <code>CompoundDocument</code> for any child component relationship objects to be created that reference the document as the parent.</p>
Delete	<p>Users must have Unlink document or Delete rights on the parent component to delete a component relationship object that references the parent.</p>	<p>Any user with Unlink document rights on the parent component is also granted Delete rights on any direct child component relationships object.</p> <p>This enables a user without delete rights on a parent document to delete child component relationship objects, that is, to modify the structure of the compound document.</p>
Modify properties	<p>Users must have Modify all properties rights on the parent component to modify properties on a component relationship object.</p>	<p>Property update rights allow users to modify order number, change child document referenced, and modify custom properties on component relationship objects.</p>

Requirements for Child Component Actions

Child Component Action	Security requirements
Delete	Before deleting a child component, query for any component relationship objects which deny deleting the child component as long as the component relationship object exists. This query is performed bypassing the credentials of the user initiating the delete action since that user may not have View all properties rights on all existing component relationships that reference the child component.
Checkout, Delete, Demote version, Promote version, Modify properties	These actions on a child component may require that component relationship objects that reference the child component be updated. This requires View all properties and Modify all properties rights on all component relationship objects that reference the child component. The user initiating the action may not have these rights on the component relationship objects so the server must be able to bypass the credentials of the user to query, retrieve and update properties as needed on these objects.

Administration

Enterprise Manager (EM) exposes the ability to modify a component document structure according to the user's access rights to the parent document. The access rights required to make modifications are listed below, along with the text that EM uses to display these on the Security property page of the Document Properties property sheet. EM disables the appropriate user interface controls if the user does not have the proper access to make a modification.

User Action	Required Access	Displayed in EM
Add compound document link	FN_ACCESS_LINK	Link a document / Annotate
Remove compound document link	FN_ACCESS_UNLINK	Unlink document
Reorder / modify compound document link	FN_ACCESS_WRITE	Modify all properties

NOTE The Unlink document access right is not included in EM's Full Control access level and must be directly assigned.

See also About compound documents in Help for Content Engine Administration.

Document lifecycle policies and lifecycle actions access rights

Document lifecycle objects - lifecycle actions and lifecycle policies - have the following security characteristics:

- Both are instances of their own classes: the Document Lifecycle Policy class and Document Lifecycle Action class. Therefore they obtain initial security from the Default Instance Security ACL of their class, like all objects do when first created. Both classes are subclassable. You can view and modify these classes under Enterprise Manager's Other classes node.
- Lifecycle actions and policies are independently securable. They are not required to have the same security as the security placed on the document class (or individual document) to which they are attached. It is obviously a much simpler security model if they do. However it can be configured differently if required by the needs of the application's security.
- Document lifecycle actions and policies do not have a security parent relationship with any other object. Specifically, a lifecycle policy does not have a security inheritance relationship with the document class to which it is associated.
- In Enterprise Manager, individual lifecycle actions and lifecycle policies are displayed in subfolders of the Object Stores > *object store name* > Document Lifecycles node. If these folders are empty it means that none have been created for that object store. Both objects have property sheets containing a Security tab which allows you to view and modify the security on that object.
- Like other objects, lifecycle actions and lifecycle policies have an owner property. The owner need not be the same as the owner of the document with which the lifecycle policy is associated.
- See Document lifecycle objects - Permissions on lifecycle policies for information about how the "Link a document" access right to a lifecycle policy can affect the user's ability to checkout a document.

Document lifecycle policy access rights

Full Control	Modify Properties	Link	View Properties
All in Modify Properties plus: <ul style="list-style-type: none"> • Delete • Modify permissions • Modify owner 	All in Link plus: <ul style="list-style-type: none"> • Modify all properties • Create instance 	All in View Properties plus: <ul style="list-style-type: none"> • Link a document 	<ul style="list-style-type: none"> • View properties • Read permissions

The Link access right is hidden in the Workplace and Workplace XT applications.

Document lifecycle action access rights

Full Control	Modify Properties	Link	View Properties
All in Modify Properties plus: <ul style="list-style-type: none"> • Delete • Modify permissions • Modify owner 	All in Link plus: <ul style="list-style-type: none"> • Modify all properties • Create instance 	All in View Properties plus: <ul style="list-style-type: none"> • Link a document lifecycle policy 	<ul style="list-style-type: none"> • View properties • Read permissions

Document lifecycle policies and lifecycle actions access rights

The Link access right is hidden in the Workplace and Workplace XT applications.

Annotation access rights

Annotations have the following security characteristics:

- They are instances of the Annotation class. Therefore they obtain initial security from the Default Instance Security ACL of their class, like all objects do when first created.
- Each annotation is independently persistable and independently secureable by its own ACL.
- Annotations are not versionable and are not automatically passed from one version of an annotated document to the next version. In order for the next version to include the same annotation as its previous version, the annotation would have to be newly created for that document version.
- The document or folder that has an annotation will be the security parent of the annotation if the grantee has been set with an inheritable depth. The permissions will appear on the annotation with a source of "inherited", which is normal in security parent relationships.
- Just like document content, an annotation's content elements, if there are any, are not independently securable and can only be accessed through the annotation object itself. In effect, then, an annotation's content has the same security as the annotation. In the Enterprise Manager, an annotation's security information appears on the Security page of its property sheet. This is available from the Annotation tab on the document or folder property sheet.
- Like other objects, annotations have an owner property. An annotation's owner need not be the same as the owner of the document or folder that contains the annotation.

Annotation access rights defined

Full Control	Modify Properties	View Properties	View Content
All access rights listed in the columns to the right and: <ul style="list-style-type: none"> • Modify permissions • Modify owner • Delete 	Modify property values	View properties and security	View content

Custom object access rights

Custom objects are containable objects (can be filed in folders) but have no content and are not versionable. They are securable, and similar to documents they acquire security from the following sources:

- The Default Instance Security tab of the class (acquired when the custom object is created).
- A security parent, if configured.
- A security policy. Because custom objects are not versionable, a security policy can apply only application security templates, and only according to a custom application written using FileNet P8 Platform APIs.

Access rights defined

Full Control	Modify Properties	Link	View Properties
All access rights listed in the columns to the right and: <ul style="list-style-type: none"> • Modify security • Change owner • Delete the custom object 	Modify property values	Associate the custom object with other objects	View properties and security

The Link access right is hidden in Workplace.

See also Custom object security levels.

Workflow definition access rights

Workflow definitions are versionable documents with the same security as regular documents. A user must have View Properties access rights to launch or to participate in a workflow.

Acquisition

A workflow definition acquires its initial security from the Default Instance Security ACL of the Workflow Definition class. Subsequent updates to this class have no effect on the security of existing workflow definitions. A workflow definition is otherwise like any other document with regard to security.

Inheritance

A workflow definition can inherit security from a folder security parent that contains the workflow definition, and can also obtain security from a security policy.

Workflow attachments

A document can be attached to a workflow. The workflow cannot alter the document's access rights in any way.

Event and subscription access rights

Event objects and subscription objects have the same access rights.

CAUTION If you modify security within an event script and you also use security propagation, FileNet recommends that you use synchronous events to minimize the chances of deadlocks that may arise with asynchronous events.

Access rights defined

Full Control	Modify Properties	Link	View Properties
All access rights listed in the columns to the right and: <ul style="list-style-type: none">• Modify security• Change owner• Delete the event or subscription	Modify property values	Associate an event with a subscription or script; associate a subscription with a document class or a workflow	View properties and security

The Link access right is hidden in the Workplace and Workplace XT applications.

Script access rights

When launched, a script contained in any of the following objects runs with Content Engine server credentials and is therefore considered a trusted script.

- Classification
- Event
- Lifecycle Action

Any other script runs with the credentials of the user who starts the script; the software enforces security for that user.

Publishing access rights

Publishing creates a new document, called a publication document, according to the specifications in the publish template selected by the user, and the optional style template specified in the publish template. For example, a manager might maintain a Procedures Guide (a Microsoft Word document) and publish it in HTML format to the /Department folder, which is available to all members of the department.

A publish template acquires security from the Publish Template class

You can think of a publish template as a wizard that Workplace users can use to create new documents by publishing existing documents. Advanced users create publish templates using the Publishing Designer application. A publish template is saved as a versionable document in the Publish Template class and acquires the class' Default Instance Security.

Access rights defined

A publish template has the same access rights as any other versionable document except that the Publish access right appear only in Enterprise Manager, which prevents users and authors from publishing a publish template.

A publication document has the security specified in the publish template

To publish a document, the user selects the document and the publish template. The user must have Publish access rights to the document and View Content access rights to the publish template, as well as the permission to add objects to the destination folder.

The publish template specifies the location, properties, and security of the publication document. The template author has the following options for applying security on the publication document:

- Specify security for the publication document
- Copy security from the document class of the publication document
- Copy security from the source document
- Use security from the destination folder (the publication document's location)
- Apply security from a security policy

See Specify publication document security in the Publishing Designer Help for more information on these options.

Republishing

When an author updates a previously published source document, the author may also want to update the publication document. The publish template specifies what occurs when a source document is republished. A republished document can replace the existing publication document or add a version to it, and:

- Copy security from the previous version, if any
- or-
- Apply security specified in the publish template

Deleting the source

The publish template specifies whether to delete publication documents when the source is deleted (a source document can have many publication documents). To delete a published source document, a Workplace user must have Owner Control of the source and any publication documents deleted with it. If the user has insufficient access rights for one or more of the deletions, the operation results in an error and no deletions occur.

Style template

A style template, created on the Content Engine using the Publishing Style Template Manager, specifies how to translate a document from its original format into HTML or PDF. No special security settings are required as the Transformation Engine takes ownership of each publish request in its queue and assumes that the user has Full Control.

Search access rights

A stored search or search template is stored as a versionable document in the Stored Search class. For security purposes, a search is like any other versionable document except that the Publish access right is hidden except in Enterprise Manager, which prevents users and authors from publishing searches. See the Search Designer Help for information on creating searches.

NOTE A stored search appears on the Workplace and Workplace XT browse pages; it looks and acts like a folder. A search template appears on both the search and browse pages; the user usually must complete the search criteria and then execute the search.

Acquisition and inheritance

The user who creates a search specifies its security. Unless the user makes a change, the search acquires the security of the Stored Search class along with any security policy that may be assigned. Subsequent updates to the Stored Search class have no effect on existing searches.

A stored search can optionally inherit from a folder security parent and from a security policy.

Access rights defined

Access right	Meaning
Owner Control	Any operation including modify security, change owner, and delete
Minor Versioning	Not supported in Search Designer
Major Versioning	Check out and check in the search as a major version, cancel a checkout
Modify Properties	Edit custom properties, rename the search
View Content	Execute the search, view search criteria, copy the search
View Properties	View properties and security
Publish (visible only in Enterprise Manager)	Publish a stored search or search template

Using searches in Workplace and Workplace XT

Workplace and Workplace XT users have various ways to access documents: by browsing, by searching, and by running a workflow.

- To execute a search, the user must have at least View Content access rights to the search.
- A user who browses to a document or search must have at least View Properties access rights to the folders in the path.
- A search can retrieve documents that the user could not access by browsing.
- For a property search, the document list includes a document if the user has at least View Properties access rights to the document.
- For a content search, the document list includes a document if the user has at least View Content access rights to the document.

Document entry template access rights

The entry template provided by Workplace and Workplace XT is an alternate "Add document" wizard created using the Entry Template Designer application. It can set a new document's security or allow the user to set it. You can also use Entry Template Designer to create alternate wizards for adding folders and custom objects.

An entry template is saved as a versionable document in the Entry Template class and acquires security from the class. An entry template is like any other versionable document with respect to security. Entry templates can only be modified through the Entry Template Designer if the user has permission to promote a version (Major Versioning).

Workflow rosters and queues

The Process Engine defines and enforces security for the rosters and queues used in processing workflows.

Workflow security

Using the Process Configuration Console, you can assign access rights to workflow rosters, work queues, component queues, and user queues. The following table describes what each access right allows you to do.

In a...	having this access right...	means you can...
Workflow roster	Query	View the roster summary of the work item. You can also view the work item itself if you have read access to the queue containing the work item.
	Create	Launch a workflow.
	Query & Create	Do both of the above.
Work or component queue	Query	View work items.
	Query & Process	Lock, modify, save, and complete work items. Note that Process access applies to the queue in which the work item is locked, rather than to the destination queue (the queue to which the work item is dispatched upon completion of the step). The destination is under system, not user, control. CAUTION See Component queue security issues for important security-related information.
User queue (a database table with a server specification, such as Inbox(0))	Query	View work items.
	Query & Process	Lock, modify, save, and complete work items. Note that Process access applies to the queue in which the work item is locked, rather than to the destination queue (the queue to which the work item is dispatched upon completion of the step). The destination is under system, not user, control.
User queue (user's subset of work items in the queue, such as Inbox)	No access rights	View work items assigned to you. In addition, you can lock, modify, save, and complete work items assigned to you. Note that you do not have full access to the work item—you can only see and modify those data fields, workflow groups, and attachments to which the workflow author has given you access.
	Query	View work items assigned to you.

	Query & Process	<p>Lock, modify, save, and complete work items.</p> <p>Note that Process access applies to the queue in which the work item is locked, rather than to the destination queue (the queue to which the work item is dispatched upon completion of the step). The destination is under system, not user, control.</p>
--	-----------------	---

Important tips regarding security

The following are several items to be aware of when assigning access rights to workflow rosters and queues.

If...	then...
the user is a member of the Process Engine Administrator Group,	the user automatically has full rights to each roster and queue, even if you don't explicitly assign him access rights.
you do not assign anyone to a specific access right for a roster or queue,	<p>you give everyone this specific access right to the workflow roster or queue. For example, if you only assign Query access rights to a user, the user can still create or process workflows if you have not explicitly assigned those access rights for the workflow roster or queue, respectively.</p> <p>CAUTION To give a specific access right to all users, leave the access right blank. Do not assign an all-inclusive group such as <code>Domain Users</code> (Active Directory). Assigning large groups to a workflow roster or queue can adversely affect database and memory usage.</p>

TIP To prevent (nearly) everyone from accessing a workflow roster or queue, assign at least one user to each possible access right for the workflow roster or queue. For example, to prevent most access to a queue, assign the `Query & Process` access right to one member of the Process Engine Administrator Group, who has implicit access to the queue anyway.

Object ownership

NOTE Prior to the current release, the security owner of an object was always set, at object creation time, to be the creator of the object. This gives the creator elevated rights to the object, which may not always be appropriate. For example, when using FileNet Records Manager, it is not always desirable for the declarer of a record to have elevated rights to the RecordInfo that is created by the declaration operation. In the present release, the object store administrator can set the default owner of new instances of a class. See Default Instance Security and Ownership below.

This topic consists of the following sections:

- Object ownership
- Default Instance Security and Ownership
- NULL Owner
- Changing Ownership
- Creator-Owner substitution

Related topics

Take or change ownership

Object ownership

All objects have an owner property. Ownership of an object confers special privileges on that object, including the right to load the object and the right to read and modify the Permissions collection on the object and modify the owner. (As explained below, markings can be used to override the special privileges implicitly granted to owners.)

An object store administrator might need to take or change ownership of an object. For example, if a user has left documents in an exclusive checkout state but is no longer available, the administrator could take ownership of the document and cancel the checkout.

You can take ownership of an object if you have the object's "Modify owner" permission. You can assign ownership of an object to another authenticated user or group if you have the object store's "Set Owner of any object" permission.

Default Instance Security and Ownership

Each class that allows instances of itself to be created has a Default Instance Security Descriptor associated with it, exposed as the Default Instance Security tab in Enterprise Manager. The Default Instance Security defines the default Permissions for new objects of that class. Default Instance Security includes a default Owner. The behavior of the Default Owner is as follows:

- If the Default Owner is set to a valid Security Principal, then that Security Principal will be assigned as the owner of instances of objects that are created.
- If the Default Owner is NULL, the owner of instances of that class will be set to NULL.
- If the Default Owner is set to #CREATOR-OWNER, then the owner of instances of objects that are created using the default will be set to the identity of the object's creator.
- In all cases, the Default Owner can be overridden at object creation time by the caller.

The default value for Default Instance Security Descriptors is established at Object Store creation time. The default value for Default Owner is always set to #CREATOR-OWNER.

NULL Owner

Objects may have NULL Owners. When the Owner is NULL, it means that the special access rights implicitly granted to the Owner are not granted to anyone. Access checking behavior is otherwise unaffected.

Changing Ownership

The owner of an object can only be changed to the caller's security identity (assuming the caller has `idmAccessRightWriteOwner`) if the caller had the `idmAccessRightWriteAnyOwner` right granted by the object store. If so, the owner can be changed arbitrarily. The purpose for this special capability is to provide a backdoor for allowing certain privileged users to recover access to any object (since once ownership is acquired, the Permissions collection can be modified and additional access rights can be granted to any user, including the owner).

There are two exceptions to this rule:

- Exception 1 is the case when the object has a security proxy wherein changing the owner is prohibited regardless of what privileges the caller has been granted.
- Exception 2 is the case when the object has one or more Markings applied that either prevent a user who has otherwise been granted `idmAccessRightWriteAnyOwner` from even connecting to the object at all, or mask the `idmAccessRightWriteOwner` access right even if the right to connect is granted.

NOTE Even with these two exceptions, a "backdoor" is still available, except that it requires a few more steps and is restricted to GCD Administrators, that is, users granted `idmAccessRightWrite` on the `EntireNetwork` object (which represents the FileNet P8 domain). Users who have write access to the FileNet P8 domain can update access rights on Markings and can therefore grant themselves any rights on any Marking.

Creator-Owner substitution

`#CREATOR-OWNER` is a special grantee that is a place holder for the future owner of an object. This grantee appears in Default Instance Security permissions lists, Default Instance Owner, Security Templates permissions lists, and permissions lists on objects that can have security children - Folders for instance.

Substitution of the `#CREATOR-OWNER` for the actual owner occurs under the following circumstances:

- When Security Templates are applied.
- When an object inherits security from a parent.
- When a security descriptor is initialized from the class Default Instance Security. An exception to this rule is when the Default Instance Owner is set to NULL in which case Permissions from the Default Instance Security that specify `#CREATOR-OWNER` as the Grantee are ignored and not copied to the Permissions collection of the created object.

NOTE Creator substitution does not occur at Permission evaluation time. As a result, `#CREATOR-OWNER` does not function as a generic macro that always evaluates to the current owner.

Property modification access

To be able to edit a property, a user generally needs Write access to the property. However, Content Engine provides an optional way to add another layer to the required access rights to modify custom properties. System administrators can do this by configuring the Modification Access of property templates. (Because this feature depends on a system property called Modification Access Required, the acronym for modification access behavior is MAR.)

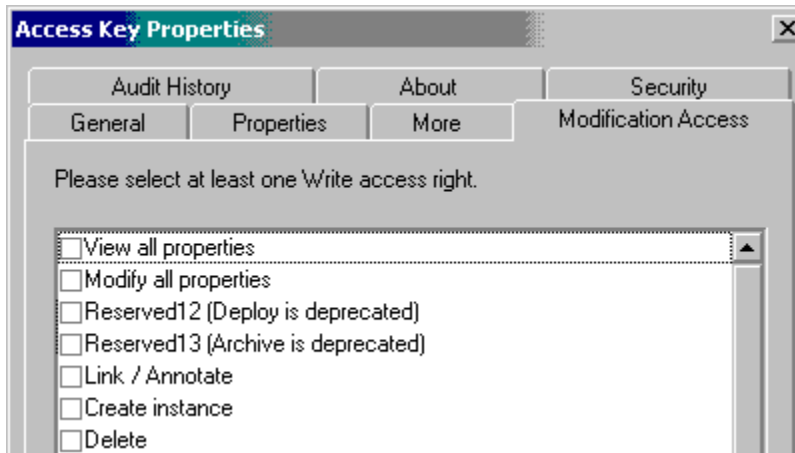
NOTE Property modification access behavior is a feature primarily intended for the FileNet P8 Records Manager application, especially in connection with markings. It is available for use by non-records management applications that need granular control over user ability to modify properties.

This topic describes modification access, normal property security, and what happens if you configure a property's modification access.

- About MARs
- Normal property security (no modification access)
- Property security if modification access has been configured
- Exceptions to the normal access requirements

About MARs

The property sheets of all property templates have a Modification Access page that contains a list of access rights:



By default, these access rights for all property templates are unselected. If left unselected, then the property template has no modification access behavior and "normal" property security applies. However, if you select one or more access rights, then properties based on the property template will have different security than normal, described below.

Normal property security (no modification access)

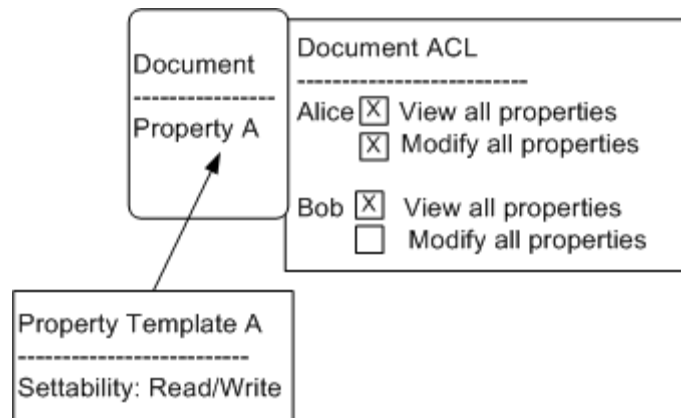
Custom properties are all based on property templates. In Enterprise Manager's Property Templates node are listed the property templates that have been added to the object store. The property template's own ACL, contained in its property sheet's "Security" page, controls who can access the property template. Default security on property templates is one of the decisions made by the Object Store Wizard: object store administrators have Full Control, and object store users have View all Properties and Read Permissions. You could change these default assignments, but only if there is a clear reason to do so. Administration of property templates is not made available by Workplace or Workplace XT, nor by any of the applications based on these applications—eForms, Records Manager, Application Integration.

Security on a property that has been added to a class and then placed on an instance of the class is governed not by the security of the property template, but by its object's security. The initial state of the security for any object is created from the default instance permissions defined via its class and thereafter edited through the Security tab of the object's property sheet. For example, the Default Instance Security of a document class is copied onto a new document, and it is this security that governs access to the document's properties.

However, keep in mind that property templates have a "Settability" setting (found on the "More" tab of the property sheet) that can be configured so that the properties based on them will be read-only, read/write, settable only on create, or settable only on checkin. These settings take precedence over any permissions granted by the object's security. In other words, a property marked read-only is not modifiable even if the user is granted the "Modify all properties" access right on the document. (See Property Template properties (More tab - String), or any of the other "More tab" property topics, for a description of the choices for settability.)

There are two property-related access rights that appear on the ACLs of all securable objects:

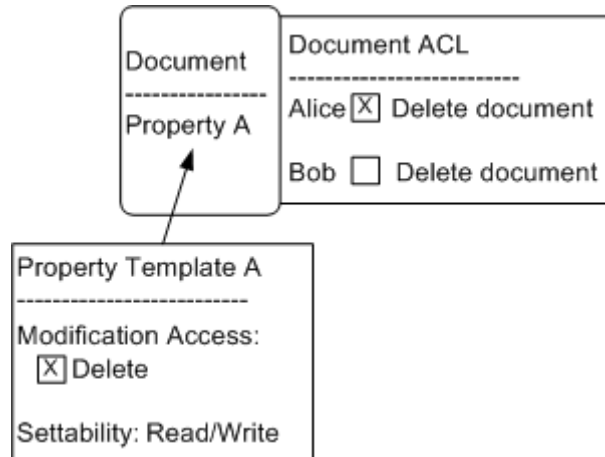
- **View all properties:** whether the user can see the object's properties.
- **Modify all properties:** whether the user can modify the values assigned to the properties (taking into account any settability restrictions on modifying the property).



In this graphic showing "normal" property security, the property template's settability is Read/Write which carries to the property when it is added to the document class and onto the document. The document's ACL grants Alice permission to see and modify all properties, while Bob can see all properties but cannot modify them. Therefore, Alice could change the value of Property A (and any of the other readable/writable properties contained by the document), while Bob can see Property A (and all others) but cannot change its value.

Property security if modification access has been configured

By setting a property's Modification Access Required property, you can establish another layer of access permissions that affects who can modify the property. You can use the MAR to specify that certain properties can be modified only by users with a specified set of rights, as opposed to the "normal" condition in which users need only the "Modify all properties" access right to the object.



In this graphic, Property Template A has a MAR in which the "Delete" permission is set. Property A has been added to the document. The ACL on the document permits Alice to delete the document while it prevents Bob from deleting the document. Therefore Alice can modify Property A and Bob cannot (assuming the property is settable).

NOTE Enterprise Manager does not collect the property modification access settings in a single view or page.

As mentioned above, if you do not specify any values, meaning that all the access rights on the Modification Access page are left unselected, then there is no MAR behavior and normal property security applies. A property template's Modification Access settings are settable only on create and therefore cannot be changed later on.

Exceptions to the normal access requirements

As described above, setting or modifying a property generally requires Write access to the object to which the property belongs. A different right is required for the following:

- To modify Permissions, SecurityPolicy, or SecurityParent, a user requires WriteAcl access.
- To modify the Owner property, a user requires WriteOwner access.

Required minimum access rights by operation

This topic describes the required minimum access rights required by Content Engine to carry out the following document-oriented operations:

- Create minor version
- Create major version
- Checkout minor version
- Checkout major version
- Cancel checkout
- Check in minor version
- Check in major version
- Promote or demote
- Delete minor or major version

The following assumptions must be made with respect to the access right requirements listed in this topic:

- In addition to the access rights specified below, the caller also has the minimum access rights required to connect to the object that is the target of the operation. (The minimum access rights required to connect to an object include Connect to store granted by the Object Store in which the object resides, and one or more of the following: View properties or Modify owner granted by the target object; or Set Owner of any object granted by the Object Store.)
- No Document Life Cycle is attached.
- No Markings are applied.
- No Security Proxy is in effect.
- No Security Policy has been applied.
- There are no object properties with Target Access Required with values set.
- There are no object-valued properties with a deletion action of Delete Action Prevent.
- The caller has sufficient access rights to delete objects related by object-valued properties that have a deletion action of Delete Action Cascade.
NOTE System properties that have this behavior include Containers, Annotations, Workflow Subscriptions, and Dependent Documents (the latter two are technically not system properties but are almost universally present).
- There are no properties with non-standard settability requirements (settable only on create or settable only before checkin).
- No changes have been made to either the Owner or Permissions simultaneous to the operation.

Create minor version

Granting object	Required minimum access rights
Object Store	Create new objects
Document Class Definition	Create instance
Document (instance just	View properties

created)	<p>-OR- Modify owner -OR- Set Owner of any object</p> <p>NOTE These rights are not, strictly speaking, necessary to create the object; however, it is possible for users to programmatically create an object to which they cannot connect.</p>
----------	---

Create major version

Granting object	Required minimum access rights
Object Store	Create new objects
Document Class Definition	Create instance
Document (instance just created)	View properties -OR- Modify owner -OR- Set Owner of any object (see NOTE above)

Checkout minor version

Granting object	Required minimum access rights
Object Store	Create new objects -AND- Modify existing objects
Document Class Definition	Create instance (for creating the reservation)
Document (target document)	Minor Versioning

Checkout major version

To checkout a major version of a document, the user must have the following minimum permissions

Granting object	Required minimum access rights
Object Store	Create new objects -AND- Modify existing objects
Document Class Definition	Create instance (for creating the reservation)
Document (the target document)	Major versioning

Cancel checkout

To cancel a document checkout, the user must have the following minimum permissions:

Granting object	Required minimum access rights
Object Store	Create new objects -AND- Delete objects
Document (the reservation)	Delete -OR- Minor Versioning -OR- Major versioning
<i>Other</i>	If exclusive reservation, the caller's SID must be equal to that of the owner of the reservation.

Check in minor version

To check in a minor version of a document, the user must have the following minimum permissions:

Granting object	Required minimum access rights
Object Store	Modify existing objects
Document (on the reservation)	Minor versioning
<i>Other</i>	If exclusive reservation, the caller's SID must be equal to that of the owner of the reservation.

Check in major version

To check in a major version of a document, the user must have the following minimum permissions:

Granting object	Required minimum access rights
Object Store	Create new objects
Document (on the reservation)	Major versioning
<i>Other</i>	If exclusive reservation, the caller's SID must be equal to that of the owner of the reservation.

Promote or demote

To either promote or demote a document, the user must have the following minimum permissions:

Granting object	Required minimum access rights
Object Store	Modify existing objects
Document (on the reservation)	Major versioning

Delete minor or major version

To delete either the minor or the major version of a document, the user must have the following minimum permissions:

Granting object	Required minimum access rights
Object Store	Delete objects
Document (on the target document)	Delete

Security policies

A security policy serves as a collection of security templates, each of which contains a predefined list of permissions, or Access Control Entries (ACEs), that can be configured to apply to a document, custom object, or folder.

Except where specifically mentioned, this topic describes the association of security policies with documents and document classes. To fully understand this topic, you should be familiar with document versioning and the versioning states Released, In Process, Reservation, and Superseded.

REMINDER Security policies are just one way to apply ACEs to an object's ACL. The other sources are the object's class, a security parent, direct edits to the object's security, and by programmatically setting the object's access rights.

About security policies

Security policies allow system administrators to apply access control to large numbers of documents without directly editing the ACL on each document.

Security policies, in conjunction with versioning states, allow a system administrator to configure the system to automatically modify ACLs on documents when their versioning state changes. For example, the administrator can configure the system to automatically grant access to a document to a wide audience when it is released.

Sequence tables detail the versioning states through which documents proceed following check in, check out, and other versioning actions.

To create a security policy, you run one of the security policy wizards provided by Enterprise Manager and by Workplace and Workplace XT. The wizard creates a security policy that the system administrator can then customize by adding security templates. Once created, the security policy can be associated with documents, folders or custom objects. Alternatively, the administrator can make the security policy the default value for the security policy property for one or more classes. Making a specific security policy the default for a class ensures that all instances of the class are associated with that security policy unless the value is explicitly overridden.

The security policy class can have subclasses, just like the other classes in Enterprise Manager's "Other Classes" node. The security policy wizard lets you create a security policy using a subclass, whereas Enterprise Manager's wizard supports only the base class. You could also use one of the supported P8 APIs to create a security policy using a subclass.

About templates

There are two kinds of security templates:

- Versioning security templates automatically update the permissions on documents as their versioning state changes to one of the four possible document versioning states: Reservation, In Process, Released, and Superseded, for which there are four corresponding versioning security templates.
- Application security templates can be configured to apply a list of permissions to a document, custom object, or folder according to logic programmed into an application using P8 APIs.

A security template applies to a document version if (1) the document version has an associated security policy, and (2) the associated security policy has a template for the document's version state. For example, if the document goes into the Released versioning state, and the security policy has a Released template, then the permissions listed in the Released template apply.

Templates cannot be shared between security policies and cannot be independently loaded or saved apart from their security policy. Permissions on an object that originate from a security policy will appear on the object's ACL with a Source type of "Template." As such they cannot be directly edited using Enterprise Manager's Access Control Editor or by using the P8 API.

Newly created security templates contain no default permissions placed on them by the Content Engine. Administrators may add permissions at creation time while running the security policy wizard, or at any later time. Applying a template that contains no permissions to an object will have the effect of removing any existing permissions on that object that were previously applied by a security policy.

Assigning security policies

Security policies can become associated with documents in two ways:

By assigning it as the default security policy for a document class. In this case, the default security policy is automatically associated with the object instance at the time of creation unless the default is explicitly overridden. The default security policy will continue to be associated with all versions in the document's version series, unless you do something to change the association. By having the same security policy for all documents in a class, you have a simple, easily understandable and manageable security scheme. If, on the other hand, you change a single document version's class, the new class's default security policy (provided there is one) would be immediately applied to that document version and the old security policy (if there was one) would be removed. However, changing a version's class does not override a security policy that was directly assigned to that version by a user, nor does it change any earlier versions of the same document.

By assigning it to a specific document version. Each document version in a version series could, theoretically, have a different security policy assigned to it. The document class's default security policy will be placed on each instance of the class, but you can override the default with a different security policy. You would do this manually, using Enterprise Manager to open the document version's property sheet and changing the security policy. This would be cumbersome and difficult to manage from a system administrator's point of view, and should be done only as an exception to the normal application by the document class.

Preserve Direct ACEs

In addition to the list of security templates associated with it, each security policy has an important property called Preserve Direct ACEs (also called Preserve Direct Permissions). This property, which can be set to either True or False, governs whether or not direct permissions are preserved in the target object's ACL when a security template is applied to it. The value of this property applies to all the templates contained by the security policy.

By default, this property is set to True, because this is likely to be the most common use case. In fact, the security policy wizard does not ask you to set a value and just sets it to True. After you have created the security policy, you can open its property sheet's General tab to view or change the Preserve Direct ACEs setting.

Effects of changes

Documents get their security policy from the default security policy of their class, if the class has one. Therefore, if you change a document's class to a class that has a different default security policy, the document will immediately take on that new security policy. This also means that if you change a document's class to one that has a security policy that is not the class' default (that is, the new class has a security policy associated with it that was not its original default policy), the document will not take on that new policy but will keep its association with the former security policy.

These scenarios describe the security behavior of changing a document's class, and assume that the security policies in question have appropriate templates.

If you change a document version's class, and both the new class and the old class have different default security policies, the permissions applied by the old security policy are immediately removed and the permissions of the new security policy are immediately applied to the version. The permissions already applied by the old document class' Default Instance ACL are not changed. In other words, the new document class does not apply its Default Instance permissions to the document.

If you change a document version's class, and the version has a policy that does not match the previous class, the software assumes that the version has a security policy that was applied by a user and leaves it as is.

If you change a document's class, and the old class has a security policy and the new class does not, the permissions applied by the old security policy are immediately removed. Since the new class has no security policy, the permissions on the document would come from the other usual sources: from being directly applied, from the Default Instance ACL of the new document class, and from a security parent, if there is one.

Rules of association

Here are the rules of association for security policies:

- A security policy can contain both versioning and application templates.
- A security policy can contain zero or more application security templates and from zero to four versioning security templates. Although it is permissible for security policies to have no templates, it is effectively only a placeholder until you add one or more templates. Workplace and Workplace XT require you to set at least one template.
- A security template can have zero or many permissions assigned to it. A template containing no permissions is allowed programmatically and by Enterprise Manager's security policy wizard. The security policy wizard provided by Workplace and Workplace XT require that you add at least one permission to a template.
- A security policy can be assigned to zero or many folders, zero or many documents, or zero or many custom objects, or any combination of the three object types.
- Documents, folders, and custom objects may have zero or one security policy.
- The class definitions for documents, folders, custom objects, and their sub-classes can each have zero or one default security policy associated with them. When instances of the class are initially created, the instance will take the default security policy if there is one. In the case of documents, the default security policy will be passed to subsequent versions, unless a different security policy is explicitly provided. In this case, the new security policy is passed to subsequent versions; it will not automatically revert to the class default.
- A single security policy can be associated with many document, folder, and custom object classes.
- A single security template cannot be shared between security policies. Each security template belongs uniquely to the security policy it is associated with.
- The versioning security templates in one security policy have no relationship with those in another security policy, even though they have the same default names (Released, In Process, Reservation, Superseded). These default names can be changed.

Changes to an existing security policy do not propagate to existing documents until the version state changes or the next version is created. If, however, you change the document's current policy to some other security policy by directly changing the document version's property sheet, then these changes are immediately applied to the version and the permissions applied by the former security policy are immediately removed.

Support in Workplace and Workplace XT

Workplace and Workplace XT fully support the effects that security policies have on document security. Advanced users of these applications with authorship permissions can run the application's security policy wizard and create security policies. The user can then assign the security policy to individual documents wherever the security page is presented (providing the user has Modify Permissions access) and on multiple documents at one time using the My Search page results list.

Because the applications do not permit editing classes, you must use Enterprise Manager to assign a security policy as the default for a document class. As explained above, the usual way to use a security policy is to assign it as the default of a document class, and then let the class apply the policy to new instances. This ensures that all documents in that class will be associated with a single security policy.

Importing and exporting security policies

Security policies can be exported and imported between object stores.

Custom objects and folders use application security templates only

The custom object class and the folder class can also be assigned a default security policy that will be associated with instances of those classes, just like they are for documents. There is an important difference: custom objects and folders can only receive permissions from application security templates, since versioning security templates can only be applied to documents. The relationship between a custom object or folder and an application security template is done using a GUID assigned to the template. The GUID is called by the application whenever the program's logic requires the security state described by the template. Associations can only be made by customized applications created using a P8 API.

Enterprise Manager's security policy wizard lets you create application security templates as part of defining a security policy. It can also automatically assign a GUID to the new template or use a 32-bit GUID that your own program generates.

Storage area security

This topic describes how to create and configure the security on the shared directory location under which file storage areas and content cache areas will be created. See Content Storage for information about how to create and administer these storage locations in Enterprise Manager. See also "Create a File Storage Area" in the Installation Guide.

General

FileNet P8 objects, including document objects, are stored in the object store's database. This is handled automatically when you successfully complete the Object Store wizard, and no additional set up is required. However, the files referenced by the content element property of a document object must be stored in one of the following content storage areas:

- Database storage area, in the same database where non-content bearing objects are stored, is automatically included and available when you create an object store. No additional security configurations are needed. See About database storage areas for more information.
- File storage areas located on either a Windows or a UNIX directory.
- Fixed storage area, requiring a fixed content device provided by one of the supported providers. See About fixed storage areas for more information.
- Content cache areas, which deliver better performance by providing local storage of frequently accessed documents without having to request them over the network.

NOTE Content cache areas are configured and secured exactly like file storage areas.

File storage area security

Content Engine operating system account

The operating system (OS) user (referred to here as the Content Engine OS user account) who logs on to the Content Engine server and starts the local application server process is the account that must be used to secure the folders and files in a file storage area. From a practical standpoint, the account that is used to install the application server should be the same account that is used to start the application server process. As an administrator, you will always log on using the same OS account to secure the folders and files in the file system that Content Engine will use for a file storage area.

Optionally, you can use an OS group account. All OS user accounts would be members of the group account. If you do not set up such a group, then that single user account must be used to log on to each Content Engine and file storage server instance.

- For Windows-based Content Engine and file storage areas, these OS login accounts must reside in the same Windows domain or in trusted Windows domains.
- For UNIX-based Content Engine and file storage areas, security is configured using NFS, the protocol suite developed by Sun Microsystems that allows different makes of computers running different operating systems to share files and disk storage. Configuring NFS security should be done by experienced network administrators. Consult your operating system documentation for details.
- For a mixed environment of UNIX and Windows, you will need an NFS Gateway product in order to provide interoperability between Windows-based and UNIX-based clients.

NOTE When we speak of Content Engine running under an OS user account, we are referring to a specific instance of the JVM that is executing as a process on the host operating system. This particular JVM is hosting the application server that launches Content Engine. The JVM executes with a set of credentials associated with a specific account known to the OS on which the process is launched. This account may or may not be defined as a user or group in the directory service that

Content Engine uses to authenticate users. It is not relevant whether or not Content Engine knows about the account. What is relevant is that the OS must know about it.

Shared root directory - Windows and UNIX

A Content Engine file storage area is always created under a shared root directory that has been created by the OS user account. The OS user is responsible for securing the root directory in a way that grants full control access to the Content Engine server (represented either by the OS user or group account) and preventing all other users from obtaining unauthorized access to files in the area.

The Create a Storage Area Wizard creates the storage area folder structures below the root directory, but does not directly set permissions on any of these folders. Instead, the folders are secured via inherited security. The inheritance scheme differs between Windows and UNIX and is discussed in detail below. Note that when Content Engine creates content element files within a storage area, it does not directly set permissions on the files; it always allows permissions to be inherited.

NOTE When properly configured using the instructions in this topic, content files are completely safe from unauthorized users.

Inherited Security - Windows file system

Under Windows, file and folder permissions can be inherited via true inheritance. Folders and files can receive inherited permissions from their parent folder, without an application setting permissions on the newly created file or folder. The Content Engine file security scheme under Windows requires the system administrator (who is logged on as the OS user account) to create the root directory in a way that provides proper inheritance and provides proper access control.

Secure the root directory of a file storage area as follows:

- Remove unwanted inherited/inheritable permissions. This may also include preventing the root directory from inheriting permissions from its parent directory, since the parent will normally grant Read access to general groups like 'users'. Be careful not to accept the default security setting when creating a root directory, since the default will normally grant Read access to a broad set of users.
- Grant the Content Engine OS account full control access to the root directory. Make this inheritable by all folders and files created under the root.
- As explained above, you can optionally grant full control access to the root directory to an OS group account that the Content Engine OS account belongs to. This must also be made inheritable by all folders and files. If you use a group account, it is not necessary to grant access to the individual user accounts.
- If there is an administrative user or group that will need access to the files of the store, then grant Full Control access to the root folder to these accounts and make the permissions inheritable by all folders and files.
- If you are running Content Search Engine (K2) administration servers, you must grant K2 access to any Content Engine file storage area that will be indexed, as follows:
 - Windows-based file storage areas: the user that the K2 service runs as must have Read access to the file storage area root directory. This is an operating system user who is defined as the logged on user of the service.
 - UNIX-based file storage areas: one way to grant access is to set the Set UID bit of all the K2 program files, and then also set the owner of each program file to a particular user. Then this particular user must have Read/Execute access to the file storage area root directory.

NOTE If you are using a group instead of individual user accounts, then the K2 operating system user must be made a member of the OS group.

UPGRADE NOTE On an upgraded file storage area, the Verity collections will be in a subdirectory of the file storage directory. In this case, the K2 operating system user must be granted the access described above to this collections subdirectory.

Sharing the root folder - Windows

Once it is secured, you must share the root folder so it can be accessed across the network. Permissions on a share cannot define access that is any less restrictive than the permissions of the folder that is shared. The recommended scheme is to grant full control access on the share to the same set of users and groups that have been granted full control access on the shared folder.

NOTE Do not confuse security on the share with security on the shared folder. These are two different Windows security features. Consult Windows operating system online help documentation for definitions and procedures.

Inherited Security - UNIX file system

Under UNIX, file and folder permissions are inherited via the file creation mask (called the umask) of the user that creates the file or folder. Files and folders do not inherit the access rights of the parent folder. The Content Engine file security scheme under UNIX is to require the system administrator to properly configure the creation mask of the Content Engine OS account to grant the proper permissions on newly created files and folders. The root folder must be created with the same permissions as those granted by the OS account's creation mask. The creation mask of the OS account should grant the following permissions:

- USER: Grant the Content Engine OS user account read/write/execute permissions.
- GROUP: Grant the Content Engine OS group account read/write/execute permission. Note that this will apply to either a private group (same Id as the user), or a specific OS group account, depending on how the OS user account is configured. (Use of a non-private group account to control security is optional.)
- OTHERS: Do not grant access. Be careful not to allow read access to Others.

Creation mask example:

```
umask u=rwx,g=rwx,o=  
or  
umask 0007
```

NFS Sharing

The root folder must be exported via NFS on the server side (meaning the computer that hosts the file system) and mounted (via NFS) on the client side. Once the root folder has been properly shared via NFS, security is enforced via the user and group identifiers of the client (as if the client logged on locally to the server computer). The client in this case is the Content Engine server running on a remote computer.

Root Account access to the NFS Share

Make sure you prevent non-local root access to a folder that has been exported via NFS, as allowing non-local root access has serious implications for security. For example, in Linux the `no_root_squash` option must not be included in the export options.

Content Engine on UNIX, file storage on Windows

This is a supported configuration with the Windows-based file storage area being made available to the UNIX Content Engine Server via NFS.

You must configure your NFS Gateway to share any Windows file shares it has access to. Note that these Windows file shares do not have to reside on the same machine as the NFS Gateway. They can be on other Windows file shares on the network. The important point is that it is the NFS Gateway that is making these shares available via NFS to any NFS client, such as Content Engine running on UNIX.

Once the NFS Gateway is set up, you have to mount these on the UNIX machines where your Content Engine Servers are running. Note that if you have a cluster of Content Engine servers on different UNIX boxes, the mount point on each UNIX box must be the same.

Content cache area security

Security for content cache areas is the same as for file storage areas. Create the shared root directory for the content cache area using the same procedures and principles as for file storage areas before running the Enterprise Manager Create a Content Cache wizard.

Content cache areas can be on UNIX or Windows. They can be remote from the Content Engine server, although they are intended to be on a network that is geographically close. It must be on a network-accessible path that all Content Engine servers in the FileNet P8 domain can access.

Target access required

Target Access Required is a feature of object-valued properties that allows an application designer to specify the access rights that are necessary on the target object in order to assign it to a property. This feature is implemented as a property named Target Access Required on the PropertyDefinitionObject class. Property Definition Objects are used to define object valued properties as part of Class Definitions. The Target Access Required property specifies the access level that the caller must be granted by the target object in order to assign it as the value of the object value property being defined.

The default value for Target Access Required is View properties, that is, if the Target Access Required is not specified then the caller must have at least Read access on the target object to refer to it using an object-valued property.

Example 1

Suppose there are two classes, A and B. Class A includes an object-valued property named OVP_B. The definition for OVP_B specifies that its required class is 'B' and the Target Access Required is Modify All Properties. This means that the value of A.OVP_B, when it is set, must always be an instance of class B. It also means that in order assign a value to A.OVP_B the caller must be granted Modify All Properties by the instance of B that is the target of the assignment.

Consider the following code snippet that illustrates assigning the value of an instance of class B to an instance of class A's OVP_B property:

```
Dim IA as A
Dim IB as B
Set IA = ObjectStore.ConnectObject("A", guidIA)
Set IB = ObjectStore.ConnectObject("B", guidIB)
IA.OVP_B = IB
```

In this example, the target object is IB; therefore the caller must have Modify All Properties access on IB or the assignment will fail.

Example 2

Suppose there are two additional classes, C and D. Class C includes an object-valued property named OVP_D. The definition for OVP_D specifies that its required class is 'D'; however the Target Access Required is not specified. This means that the value of C.OVP_D, when it is set, must always be an instance of class D; however, since the Target Access Required on the definition for OVP_D is not specified, the caller only needs to be granted View properties by the instance of D.

Consider the following code snippet that illustrates assigning the value of an instance of class D to an instance of class C's OVP_D property:

```
Dim IC1 as C
Dim ID1 as D
Set IC1 = ObjectStore.ConnectObject("C", guidIC1)
Set IC1 = ObjectStore.ConnectObject("B", guidIB)
IC.OVP_D = ID
```

In this example, the target object is IC. There is no Target Access Required on the property definition for OVP_D therefore it defaults to View properties which means that the caller must have Read access to IB or the assignment will fail.

Understanding security inheritance

General description

Security inheritance is the passing of permissions from a parent object to a child object. For instance, a folder could be the parent of a sub-folder or a document, a document could be the parent of an annotation, or a class could be the parent of a subclass. The security administrator can apply security updates to many subclasses or objects in one operation by setting the permissions at the parent level.

Some important characteristics of inherited permissions are:

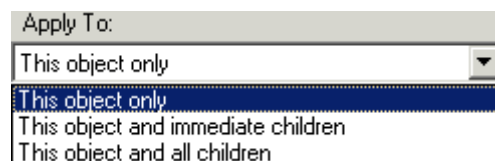
- **Source type:** Inherited permissions have a source type of Inherited. Other source types include Default, Direct, and Template. For more information on the different source types, see [About access rights](#).
- **Not modifiable:** You cannot modify inherited permissions on a child object. You must modify the permissions on the parent object. Inherited permissions are displayed as disabled in security interfaces like the Enterprise Manager's security editor. The modifications will be automatically applied to the inherited permissions on the child objects, after the next metadata refresh.
- **Delete behavior:** When you delete a security parent, its inherited permissions will be deleted from all of its child objects.
- **Document version update behavior:** When you change the inheritable permissions on a security parent, the change will take place on all versions of a security child document which is inheriting those permissions.
- **Inherit only:** The Default Instance Security ACLs of folder classes, document classes, and custom object classes have several permissions marked (Inherit only). These rights are special in that they do not control access to the object but are intended to be inherited. See the procedures [Configure a folder's security inheritance](#) and the [Configure security inheritance](#).
- To enable inheritance on folders, use the Inherit parent permissions checkbox, on the folder's General tab.

Inheritable depth (Apply to)

Each ACE has an inheritable depth setting that is invoked if the ACE is configured to be inherited by a child object. The inheritable depths are:

- This object only. Such an ACE would not be inherited. Therefore, even if this ACE is in a position to be inherited, it will not be inherited by a child object.
- This object and immediate children. Such an ACE would be inherited by the parent object's child, but not by the child object's children. After inheritance takes place, the child ACE will have an inheritable depth of This object only. This ensures that it will not be inherited again.
- This object and all children. Such an ACE would be inherited by as many generations of child objects as are created. After inheritance takes place, the child object's ACE will have an inheritable depth of This object and all children.

The Enterprise Manager security editor lets you set inheritable depth in its [Apply to](#) section:



See [Configure inheritable depth](#) for more information.

NOTE The setting for inheritable depth does not apply to situations where the ACE is being applied in non-inheritance conditions. For example, an ACE on a Default Instance Security ACL of a class will be applied to an instance of that class, even if the ACE has an inheritable depth of This object only, for the reason that the application of security from a Default Instance ACL is considered default security and not inherited security.

Security Parent, Security Folder, Security Proxy Type

Content Engine (CE) provides several means for configuring the security parent - security child relationship whereby inherited permissions can be applied to many objects while being sourced and controlled from a single object.

Security Parent

Earlier versions of CE provided the Security Parent, which was a folder that had been configured to pass inheritable permissions to the documents and custom objects it contained. This feature has been deprecated, although it continues for backward compatibility. Any applications already making use of this feature will continue to work as before, without need to change. Upgrading to the new CE version will not upgrade existing uses of the Security Parent property.

New uses in Enterprise Manager (EM) of the Security Parent folder feature will appear as before, but in fact will make use of the new Security Folder feature (see next entry). You can see the Security Parent property on all folders: in EM, right-click the folder, select Properties, then click the Properties tab and scroll down the list of properties.

Security Folder

The new Security Folder feature is similar to the Security Parent feature, with the important difference that it does not require that its security children be filed in the folder. You can see the Security Folder property on all folders: in EM, right-click the folder, select Properties, then click the Properties tab and scroll down the list of properties.

Security Proxy Type

CE provides extensible security parent relationships by means of a new Security Proxy Type property placed on the metadata of custom object-valued properties. You can add this property to whatever class of objects your security design requires. An object can have many such properties, that is many security parents, with the inherited ACEs from each being merged together and contributing with equal priority to the access check that produces the final security mask.

You can apply the Security Proxy Type property to objects other than folders. For example, a custom object could be configured to inherit security to a document. You can see the Security Proxy property on all objects: in EM, right-click the object, select Properties, then click the Properties tab and scroll down the list of properties.

See Configure security inheritance to configure these security inheritance features.

Configuring inheritance

For configuring inheritance on **documents** and **custom objects**, see Configure a document's security parent and Configure a folder to be a security parent.

Folders have an **Inherit parent permissions** checkbox on the General tab of their property sheets in EM that governs inheritance between folders:

Inherit parent permissions

When selected, the folder will inherit permissions from its parent folder, if the parent folder has inheritable permissions (see Inheritable depth above). This checkbox is selected by default. Clearing the checkbox

removes any ACEs that were formerly inherited from the parent folder and stops any further security inheritance from the parent folder.

Subclassed permissions are copied, not inherited

There is another condition where security can pass from parent to child that is not considered inheritance although it has some of the same characteristics. When a new subclass is created, class permissions with a source type of Default or Direct are copied to the new subclass. This occurs when no permissions have been provided, and the permissions copied from the super class become the default value of the permission property for the new class definition. This is an exact copy, with no change in source type or inheritable depth.

A child class will take on the ACEs of its parent as described in the following table. Notice that Default permissions with inheritable depth of This object only are not inherited by the child: rather they are copied without change and therefore are modifiable on the child object.

NOTE Permissions on the Default Instance Security tab of the class are passed to the instances of the class, while permissions on the Security tab govern who can access the class itself.

If the ACE on the parent class is marked then the same ACE on the subclass will be marked ...	Inheritance or copy?
Source = Default Inheritable depth = "This object only"	Source = Default Inheritable depth = "This object only"	Copy
Source = Direct Inheritable depth = "This object and immediate children"	Source = Inherited Inheritable depth = "This object only"	Inheritance
Source = Direct or Inherited Inheritable depth = "This object and all children"	Source = Inherited Inheritable depth = "This object and all children"	Inheritance
Source = Direct or Inherited Inheritable depths = "This object only"	Does not appear. Inheritance is stopped by the inheritable depth.	n/a

For a discussion of property inheritance between classes, see [Inheritance between classes](#).

Summary of security sources

The table below summarizes how classes receive default security. Security inheritance takes place only if the source class or object has inheritable permissions:

Object	Initial security comes from...	Inherits additional security from...	Its security can be inherited by ...
Folder	Its class. Security policy, if configured.	Parent folder, if Inherit parent permissions enabled.	Child folders, if Inherit parent permissions is enabled on those child folders, and if there are inheritable ACEs. Documents or custom objects that consider the folder its security folder, if the folder has inheritable ACEs.
Document	Its class. Security policy, if	Security folder, if configured.	Any annotations assigned to the document version, if the document has inheritable ACEs.

	configured.	Custom object-valued properties with Security Proxy Type set. See Configure security inheritance.	
Custom object	Its class. Security policy, if configured.	Same as Document.	<none>
Annotation	Its class.	Document.	<none>
Other Classes	Its parent class.	Any additional parent classes up to the top of the class hierarchy.	Child classes, if there are inheritable ACEs.

Directory service providers

This section describes FileNet P8 directory service provider integration and configuration.

Introduction

The topics in this Security Guide section discuss Content Engine security data retrieval from directory servers for the purpose of authorizing users and groups. They also address how to configure authentication, since both authentication and authorization rely on directory service repository and logically cover the same set of security data. Authentication is covered in the topics located under Authentication.

Content Engine does not implement its own authentication module. Instead, it leverages the J2EE application server's authentication mechanism. Before clients can log onto a Content Engine server, the application server's authentication providers have to be configured to point to specified directory servers.

Content Engine's security objects, such as realms, groups and users, are stored in directory servers. Content Engine retrieves those objects through a "Directory Service Provider" layer. There are different provider implementations for different types of directory servers. Content Engine implements providers for the following directory servers:

- Microsoft Active Directory
- Microsoft Active Directory Application Mode (ADAM)
- Sun ONE directory server
- Novell eDirectory server
- IBM Tivoli directory server

A Directory Server is divided into "partitions", each of which is called a naming context (or sometimes "namespace"). There are different types of naming contexts, such as the configuration naming context that holds configuration information, and data naming context which contains all directory data. Each data naming context is defined as a Content Engine realm. Each realm contains groups and users.

Terminology and basic concepts

Distinguished Name

A name that uniquely defines a directory entry within an LDAP server and locates it within the directory tree. A typical distinguished name might be:
CN=StephenHawking,CN=Users,DC=Filenet,DC=Com. This distinguished name identifies the "Stephen Hawking" user object in the Filenet.com domain.

User Principal Name

A user principal name (UPN) is a friendly name that is short and easy to remember. The user principal name consists of a shorthand name that represents the user and usually the DNS name of the domain where the user object resides, or any other designated name.

The user principal name format consists of the user name, the "at" sign (@), and a user principal name suffix. For example, the user James Smith, who has a user account in the reskit.com domain, might have the user principal name JSmith@reskit.com. The user principal name is independent of the distinguished name of the user object, so a user object can be moved or renamed without affecting the user principal name.

Among the types of directory servers that Content Engine supports, only Active Directory has a UPN attribute. The attribute name is userPrincipalName.

Short Name

Short name is a property in both Content Engine User and Group classes:

- User.ShortName
- Group.ShortName

Both user short name and group short name must be unique across all configured directory servers. Unlike DN (distinguished name) and UPN (user principal name), the value of short name does not contain realm information.

User short name is normally mapped to a user login ID in the LDAP repository (samAccountName attribute in AD, uid attribute in SunOne, cn in both IBM and Novell). Group short name is normally mapped to the samAccountName attribute in AD and cn attribute in other types of LDAP servers.

User short name is also persisted to a Content Engine object store as a property such as Document.Creator. Both User.ShortName and Group.ShortName are configurable through the Content Engine API and Enterprise Manager .

Realm

In this document the term realm describes a base object for searching the directory. When the FileNet P8 services interact with a directory service, most operations are done in the context of a realm. Process Engine APIs use the term security domain to be consistent with the IS domain concept. This section describes how FileNet P8 defines a realm for each type of supported directory server.

FileNet P8 domain

When you install Content Engine you will create a new FileNet P8 domain which provides the security context for authenticating applications. In order to avoid possible confusion between the FileNet P8 domain and Windows domains, these terms are usually spelled out completely. For more information, see Concepts: FileNet P8 Domain.

Authentication Provider

All interaction with the directory server that has been configured during installation as the FileNet P8 authentication provider is read-only and is initiated only from Process Engine and Content Engine servers.

Logon

FileNet P8 lets you configure a logon supporting a number of different parameters. Because authentication and logon attributes are persisted in databases, workflow definitions, and stored searches you cannot change the attribute at a later time, including during upgrade. See the sections describing logging on in the topic that describes your directory server.

Find

Documentation refers frequently to "finding" users and groups. This refers to the activities of Enterprise Manager's Select Users and Groups dialog box, which Enterprise Manager uses to search for accounts to add to the ACL of an object, and also to similar controls used in Workplace and Task Manager. On the Content Engine API level, "finding" refers to the FindUsers and FindGroups methods.

SSL

Customers should configure SSL to avoid passing credentials in clear text between the FileNet P8 servers and the directory server. See the Installation Guide for information about configuring SSL.

Group support

FileNet P8 supports groups that can include any number of users and other nested groups. Also, it honors any account states and restrictions (such as whether disabled and logon hours) defined by the directory server.

Configuration Overview

Directory configuration for Content Engine is conducted in the following two areas: authentication and authorization. Content Engine does not support different types of directory servers in the same Content Engine domain.

Directory Configuration for Authentication

Directory configuration for authentication, including configuring login formats, occurs in the application server's authentication providers.

Content Engine server does not implement its own authentication module. Instead, it leverages a J2EE application server's authentication mechanism. Before clients can log onto a Content Engine server, the application server's authentication providers must be configured to point to specified directory servers.

In WebLogic 8.1 application server, for example, Content Engine can support three types of authentication providers:

- Authentication provider by Distinguished Name (DN): This provider authenticates a user by its DN. In this provider, set your "User Name Attribute" to your LDAP DN attribute. You can set your "Static Group Name Attribute" to another LDAP attribute.
- Authentication provider by User Principal Name (UPN): This provider authenticates a user by its UPN. In this provider, set your "User Name Attribute" to your LDAP UPN attribute. You can set your "Static Group Name Attribute" to another LDAP attribute. UPN takes the format of LoginID@yourDomain.com. Note that among the four types of directory server Content Engine supports, only Active Directory has the UPN attribute. So only Active Directory servers support this type of authentication provider.
- Authentication provider by Short Name: In this provider, set your "User Name Attribute" to your short name attribute. You can set your "Static Group Name Attribute" to another short name attribute.

With WebLogic, you can create more than one authentication provider for each LDAP server. Here is a configuration example for an Active Directory domain named "paperwork2", for WebLogic. In this case, there are two short name providers - samAccountName and cn.

```
DefaultAuthenticator
DefaultIdentityAsserter
Paperwork2_By_samAccountName
Paperwork2_By_cn
Paperwork2_By_UserPrincipalName
Paperwork2_By_DistinguishedName
```

NOTE DefaultAuthenticator should always be first in the list.

When configuring WebLogic authentication providers, the LDAP attributes used for fields "User Name Attribute" and "User From Name Filter" need to be the same. This rule also applies to group configuration. Here is an example that uses samAccountName:

```
User Name Attribute: samAccountName
User From Name Filter: (&(objectClass=user)( samAccountName =%u))
```

If the CN attribute is unique across all Active Directory servers accessed by Content Engine, you can also use it as the short name, but note that CN is normally in the format of <first name + last name> for user objects. Here is an example showing the difference between samAccountName and CN:

```
CN: Stephen Hawking
samAccountName: shawking
```

Directory Configuration for Authorization

A GCD administrator can log on to Enterprise Manager and configure the direct connection between Content Engine and the directory service. See [Create a Directory Configuration wizard](#) for information about how to run this wizard.

Third-party developers can also directly call the Content Engine APIs to automatically configure it.

Sun Java System Directory Server

This topic describes FileNet P8's support for integrating with Sun Java System Directory Server.

General

One instance of Sun Java System Directory Server can have multiple data naming contexts. Because each Sun data naming context is mapped to a Content Engine realm, one Sun Java System Directory Server can be mapped to multiple Content Engine realms.

For each realm, you need to create an application server authentication provider and a DirectoryConfigurationSunOne object, so that there is a one-to-one relationship between Realm object and authentication provider, and also a one-to-one relationship between Realm object and DirectoryConfigurationSunOne object.

For each DirectoryConfiguration object, FileNet P8 extracts the realm name from the specified UserBaseDN property value by comparing it with each data naming context. For example, if the UserBaseDN for this DirectoryConfiguration object is "ou=people, o=isp ", and there are two data naming contexts: "o=isp" and "dc=filenet,dc=com", then you know the realm name for this DirectoryConfiguration object is "o=isp".

NOTE FileNet highly recommends that you configure SSL between your application server that hosts Content Engine and your Sun Java System Directory Server. This will include making changes in the application server to the authentication provider's DirectoryConfigurationSunOne object that was created while running Content Engine Setup. Consult your application server's documentation for instructions.

Support matrix

Use this support matrix as a quick lookup of supported directory features.

SunONE Features	Supported By Content Engine
One-way SSL	Y
Two-way SSL	N
Transport Layer Security (TLS)	N
Static Groups	Y
Nested Groups	Y
Dynamic Groups	N
Universal Groups	N
Supported User type (objectClass)	inetOrgPerson
Supported Static Group types (objectClass)	groupOfUniqueNames
Follow referrals for Search (e.g. User/Group retrieval)	N NOTE Earlier releases of FileNet P8 were able to follow referrals, but this is not supported in the present release due to problems with how SunOne performs sorting. Use SunOne's server chaining instead of referrals.
Support multiple realms	Y
Chaining	Y

Roles	N
Directory aliases	N
Restrict to single realm	Y - Just configure one realm in application server
Configurable user short name attribute	Y - Because the short name does not contain realm information, short names must be unique across all your configured domains and realms.
Configurable group short name attribute	Y
Configurable user display name attribute	Y
Configurable group display name attribute	Y
Multiple authenticating attributes support	Y- Can authenticate against the same SunONE server with multiple attributes, such as uid or distinguishedName. See Configure multiple authenticating attributes.
Sorting	Y- Return users/groups in sorted order: either ascending or descending order.
Paging/Continuation	Y- Return users/groups page by page. Page continuation happens automatically in the back end.

Directory Configuration Properties

The following is an alphabetic list of the properties in the DirectoryConfigurationSunOne class. Use Enterprise Manager to view all properties and modify editable properties.

Property Name	Editable?	Description
ClassDescription	N	A ClassDescription object containing the fixed description of the class from which a given object is instantiated.
DirectoryServerHost	Y	Specifies the name of the host that is running the directory server product.
DirectoryServerPassword	Y	Specifies the user password used to authenticate to a given directory server.
DirectoryServerPort	Y	Specifies the port number of the directory server. The value of this property defaults to port 389 for all supported directory server types.
DirectoryServerProviderClass	Y	Specifies the directory server provider class name: com.filenet.engine.security.SunOneProvider
DirectoryServerType	N	Specifies the type of directory server: SunOne
DirectoryServerUserName	Y	Specifies the username for authenticating to the directory server. Example: "uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot@quot;"
DisplayName	Y	The user-readable, provider-specific name of an object. This property is usually the designated Name property of the object's class.
GroupBaseDN	Y	The base DN for searching for groups in the directory server.
GroupDisplayNameAttribute	Y	Specifies the display name for a Group object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
GroupMembershipSearchFilter	Y	The search filter for group membership queries.
GroupNameAttribute	Y	Defines the directory server attribute to be used as the short name for a group.
GroupSearchFilter	Y	Specifies search filter for groups. Example: "(&(objectclass=group)(samAccountName={0}))", where samAccountName will serve as the short name.
Id	N	An object's globally unique ID (GUID).
IsSSEnabled	Y	Defines whether or not Secure Sockets Layer (SSL) protocol is enabled for a given DirectoryConfiguration object. The default value is false, indicating that SSL is disabled.
UserBaseDN	Y	The base DN for searching for users in the directory server.
UserDisplayNameAttribute	Y	Specifies the display name for a User object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
UserNameAttribute	Y	Defines the directory server attribute to be used as the short name for a user.
UserSearchFilter	Y	Specifies search filter for users. Example:

		“(&(objectclass=user)(samAccountName={0}))”, where samAccountName will serve as the short name.
--	--	---

Realm Configuration (WebLogic)

Authentication Provider Configuration, using WebLogic

This section provides realm configuration instructions for WebLogic. After the authentication providers are configured, you must restart WebLogic.

Content Engine can create at most two authentication providers for each Sun Java System Directory Server data naming context (that is, a realm). These two providers are used to authenticate by short name and distinguished name. Content Engine cannot create an authentication provider for UPN since Sun does not have an LDAP attribute to hold the user name in UPN format.

In the following configuration example, short name is mapped to uid:

```
DefaultAuthenticator
DefaultIdentityAsserter
SunOne_By_uid (short name)
SunOne_By_entrydn (DN)
```

When configuring WebLogic authentication providers, the LDAP attribute used for the fields "User Name Attribute" and "User From Name Filter" must be the same. The same applies to group configuration. Here is an example which uses uid:

- User Name Attribute: uid
- User From Name Filter: (&(objectclass=user)(uid =%u))

For each Sun Java System Directory Server, check the attribute namingContexts in its rootDSE. Then find out how many data naming contexts it has. For each data naming context, create a WebLogic authentication provider by doing the following:

- In WebLogic, navigate to mydomain\Security\Realms\myrealm\Providers\Authentication
- Click "Configure a new iPlanet Authenticator..."
- Name: give a name
- Control Flag: Sufficient
- Click "Create"
- Click the "iPlanet" tab
- Host: your host
- Port: <port>
- Principal: <your principal>
- Credential: <password>
- Confirm Credential: <password>
- Click "Apply"
- Click the "Users" tab
- User Name Attribute: uid
- User Base DN: your base DN. For example, ou=people,o=isp
- User From Name Filter: (&(objectClass=person)(uid=%u))
- Click "Apply"

- Click the "Groups" tab
- Group Base DN: your base DN. For example, ou=groups,o=isp
- Group From Name Filter: (&(objectClass=groupOfUniqueNames)(cn=%g))
- Static Group Name Attribute: cn
- Click "Apply"
- Click the "Membership" tab
- Static Group DN's from Member DN Filter : (&(objectClass=groupOfUniqueNames)(uniqueMember=%M))
- Click "Apply"
- Restart WebLogic
- Navigate to mydomain\Security\Realms\myrealm\Users (this may take awhile and typically will only show a partial list of users).
- Ensure you see users from your SunOne server.
- Navigate to mydomain\Security\Realm\myrealm\Groups (this may take awhile and typically will only show a partial list of groups).
- Ensure you see groups from your SunOne server.

Example: Assume a Sun Java System Directory Server has the following two data naming contexts:

- o=isp
- dc=mycompany,dc=com

You must create two WebLogic authentication providers for it.

Provider Name	Tab	Field Name	Field Value
isp_uid	General	Control Flag	Sufficient
	iPlanet LDAP	Host	sunOne-host
		Port	389
		Principal	uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
		Credential	<password>
	Users	User Name Attribute	Uid
		User Base DN	ou=people,o=isp
		User From Name Filter	(&(objectClass=person)(uid=%u))
	Groups	Group Base DN	ou=groups,o=isp

		Group From Name Filter	(&(objectClass=groupOfUniqueNames)(cn=%g))
		Static Group Name Attribute	cn
	Membership	Static Group DN's from Member DN Filter	(&(objectClass=groupOfUniqueNames)(uniqueMember=%M))
MyCompany_uid	General	Control Flag	Sufficient
		iPlanet LDAP	Host
		Port	389
		Principal	uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
		Credential	<password>
	Users	User Name Attribute	uid
		User Base DN	Ou=users,dc=mycompany,dc=com
		User From Name Filter	(&(uid=%u)(objectclass=person))
	Groups	Group Base DN	Ou=users,dc=mycompany,dc=com
		Group From Name Filter	(&(objectClass=groupOfUniqueNames)(cn=%g))
		Static Group Name Attribute	cn
	Membership	Static Group DN's from Member DN Filter	(&(objectClass=groupOfUniqueNames)(uniqueMember=%M))

GCD Configuration

For the example in the previous section, you must create the following two DirectoryConfigurationSunOne objects - one for each data naming context.

DirectoryConfigurationSunOne object 1:

- DisplayName: isp
- DirectoryServerHost: sunOne-host
- DirectoryServerPort: 389
- DirectoryServerUserName:
uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: ou=people,o=isp
- UserSearchFilter: (&(objectClass=person)(uid={0}))
- UserNameAttribute: uid
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=groups,o=isp
- GroupSearchFilter: (&(objectClass=groupOfUniqueNames)(cn={0}))
- GroupMembershipSearchFilter: (&(objectClass=groupOfUniqueNames)(uniqueMember={0}))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

DirectoryConfigurationSunOne object 2:

- DisplayName: MyCompany
- DirectoryServerHost: sunOne-host
- DirectoryServerPort: 389
- DirectoryServerUserName:
uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: ou=users,dc=mycompany,dc=com
- UserSearchFilter: (&(objectClass=person)(uid={0}))
- UserNameAttribute: uid
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=users,dc=mycompany,dc=com
- GroupSearchFilter: (&(objectClass=groupOfUniqueNames)(cn={0}))
- GroupMembershipSearchFilter: (&(objectClass=groupOfUniqueNames)(uniqueMember={0}))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

Configuration Result

Authentication	FileNet P8 can authenticate any user by uid under: ou=people,o=isp ou=users,dc=mycompany,dc=com
Realm set	Two realms (i.e., all the data naming contexts): o=isp dc=mycompany,dc=com
User/group retrieval	FileNet P8 can retrieve any user under: ou=people,o=isp ou=users,dc=mycompany,dc=com FileNet P8 can retrieve any group under: ou=groups,o=isp ou=users,dc=mycompany,dc=com
Group membership	FileNet P8 can search group memberships within a realm. Note that Sun ONE does not support group membership across naming contexts in static groups.

Operation

Get User or Group by Short Name

Iterate through all realms. For each realm:

1. Connect to corresponding host.
2. Search for the user/group by short name.
3. If found, return.

If more than one user/group is found, Content Engine will log an error and return the first user found.

Get User or Group by DN

1. Resolve the realm name from the DN.
2. Connect to corresponding host.
3. Search for the user/group by DN.

Get User or Group by UPN

SunOne provider does not support these methods.

Get User or Group by SID

Iterate through all realms. For each realm:

1. Connect to corresponding host.
2. Search for the user/group by SID.

3. If found, return.

Search Users or Groups in a Given Realm

1. Connect to the host corresponding to the specified realm.
2. Search for the users or groups by the search criteria.

Novell eDirectory

This topic describes FileNet P8's support for integrating with Novell eDirectory directory server.

General

One instance of Novell eDirectory directory server can have multiple contexts. Because each context immediately under the ROOT DSE (tree Object) is mapped to a Content Engine realm, one eDirectory server can be mapped to multiple Content Engine realms.

For each realm, you must create an application server authentication provider and a DirectoryConfigurationNovell object, so that there is a one-to-one relationship between Realm object and authentication provider, and also a one-to-one relationship between Realm object and DirectoryConfigurationNovell object.

For example:

- If the user base DN is “dc=filenet.com, ou=eng, o=cedev1” then o=cedev1 will be the context for all the objects under it, and it is the first level under the ROOT DSE, which is the name of Content Engine Realm object.
- If the user based DN is “dc=filenet.com, ou=eng, c=US” then c=US will be the context for all the objects under it, and it is the first level under the ROOT DSE, which the name of Content Engine Realm object.

For each DirectoryConfigurationNovell object, FileNet P8 uses the specified UserBaseDN property value to lookup context.

NOTE FileNet highly recommends that you configure SSL between your application server that hosts Content Engine and your Novell eDirectory servers. This will include making changes in the application server to the authentication provider's DirectoryConfigurationNovell object that was created while running Content Engine Setup. Consult your application server's documentation for instructions.

Support matrix

Use this support matrix as a quick lookup of supported directory features.

Novell eDirectory Features	Supported By Content Engine
One-way Secure Sockets Layer (SSL)	Y
Two-way SSL	N
Transport Layer Security (TLS)	N
Container types supported	Country (C), Organization (O), Organizational Unit (OU), Domain (DC)
Static Groups	Y
Nested Groups	N (eDirectory does not support nested groups; a group cannot have siblings or grand parents)
Dynamic Groups	N
Supported User type (objectClass)	Person

Supported Static Group types (objectClass)	groupOfNames
Roles	N
Follow referrals for Search (e.g. User/Group retrieval)	N
Support multiple realms	Y
Chaining	Y
Directory aliases	N
Restrict to single realm	Y - Just configure one realm in application server
Configurable user short name attribute	Y - Because the short name does not contain realm information, short names must be unique across all your configured domains and realms.
Configurable user display name attribute	Y
Configurable group display name attribute	Y
Configurable group name attribute for persisting	Y
Multiple authenticating attributes support	Y - See Configure multiple authenticating attributes for details.
Sorting	Y- Return users/groups in sorted order (ascending only)
Paging/Continuation	Y- Return users/groups page by page. Page continuation happens automatically in the back end.
Typeful and type less name display	Typeful name display only

Directory Configuration Properties

The following is an alphabetic list of the properties in the DirectoryConfigurationNovell class. Use Enterprise Manager to view all properties and modify editable properties.

Name	Editable?	Description
ClassDescription	N	A ClassDescription object containing the fixed description of the class from which a given object is instantiated.
DirectoryServerHost	Y	Specifies the name of the host that is running the directory server product.
DirectoryServerPassword	Y	Specifies the user password used to authenticate to a given directory

server		server.
DirectoryServerPort	Y	Specifies the port number of the directory server. The value of this property defaults to port 389 for all supported directory server types.
DirectoryServerProviderClass	Y	Specifies the directory server provider class name: com.filenet.engine.security.EDirectoryProvider
DirectoryServerType	N	Specifies the type of directory server: Novell
DirectoryServerUserName	Y	Specifies the username for authenticating to the directory server.
DisplayName	Y	The user-readable, provider-specific name of an object. This property is usually the designated Name property of the object's class.
GroupBaseDN	Y	The base DN for searching for groups in the directory server.
GroupDisplayNameAttribute	Y	Specifies the display name for a Group object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
GroupMembershipSearchFilter	Y	The search filter for group membership queries.
GroupNameAttribute	Y	Defines the directory server attribute to be used as the short name for a group.
GroupSearchFilter	Y	Specifies search filter for groups. Example: "(&(objectclass=group)(samAccountName={0}))", where samAccountName will serve as the short name.
Id Property	N	An object's globally unique ID (GUID).
IsSSEnabled	Y	Defines whether or not Secure Sockets Layer (SSL) protocol is enabled for a given DirectoryConfiguration object. The default value is false, indicating that SSL is disabled.
UserBaseDN	Y	The base DN for searching for users in the directory server.
UserDisplayNameAttribute	Y	Specifies the display name for a User object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
UserNameAttribute	Y	Defines the directory server attribute to be used as the short name for a user.
UserSearchFilter	Y	Specifies search filter for users. Example: "(&(objectclass=user)(samAccountName={0}))", where samAccountName will serve as the short name.

Realm Configuration (WebLogic)

Authentication Provider Configuration

This section shows realm configuration instructions for WebLogic. After the authentication provider is configured, you must restart WebLogic server.

When configuring WebLogic authentication providers, the LDAP attribute used for the fields "User Name Attribute" and "User from Name Filter" must be the same. The same applies to group configuration. For example:

- User Name Attribute: cn
- User From Name Filter: (&(objectclass=person)(cn =%u))

For each Novell eDirectory directory server, check the Contexts immediately under its root DSE, in order to find out how many contexts it has. For each context, create a WebLogic authentication provider by doing the following:

- In WebLogic, navigate to mydomain\Security\Realms\myrealm\Providers\Authentication
- Click "configure a new Novell Authenticator....."
- Name: <name>
- Control Flag: Sufficient
- Click "Create"
- Click the "Novell LDAP" tab
- Host: your host running eDirectory
- Port: <port> (389 by default)
- Principal: <your principal>
- Credential: <password>
- Confirm Credential: <password>
- Click "Apply"
- Click the "Users" tab
- User object class : person
- User Name Attribute: cn
- User Base DN: your base DN. For ex: dc=filenet.com, ou=eng, o=cempdev1.
- User search scope: subtree (you can change if you wish to search the user at the passed DN level by changing to one level).
- User From Name Filter: (&(cn=%u)(objectclass=person))
- Click "Apply"
- Click the "Groups" tab
- Group Base DN: your base DN. For ex: ou=eng, o=cempdev1
- Group From Name Filter: (&(cn=%g)(objectclass=groupofNames))
- Static Group Name Attribute: cn
- Click "Apply"

- Click the "Membership" tab
- Static Group DN's from Member DN Filter : (&(uniquemember=%M)(objectclass=groupofnames))
- Click "Apply"
- Restart WebLogic
- Navigate to mydomain\Security\Realms\myrealm\Users (this may take awhile and typically will only show a partial list of users).
- Ensure you see users from your eDirectory server.
- Navigate to mydomain\Security\Realm\myrealm\Groups (this may take awhile and typically will only show a partial list of groups).
- Ensure you see groups from your eDirectory server.

Example: Assume Novell eDirectory directory server has the following two contexts.

- o=cempdev1
- dc=filenet.com

You must create two WebLogic authentication providers.

Provider Name	Tab	Field Name	Field Value	
eng_uid	General	Control Flag	Sufficient	
	Novell LDAP	Host	Ed-host	
		Port	389	
		Principal	cn=principal1	
		Credential	<password>	
	Users	User Name Attribute	cn	
		User Base DN	ou=people, o=cempdev1	
		User From Name Filter	(&(cn=%u)(objectclass=person))	
	Groups	Group Base DN	ou=groups, o=cempdev1	
		Group From Name Filter	(&(cn=%g)(objectclass=groupOfNames))	
		Static Group Name Attribute	cn	
		Membership	Static Group DN's from Member DN Filter	(&(member=%M)(objectclass=groupOfNames))

MyCompany_uid	General	Control Flag	Sufficient
	Novell LDAP	Host	Ed-host
		Port	389
		Principal	cn=principal2
		Credential	<password>
	Users	User Name Attribute	cn
		User Base DN	ou=users,dc=filenet.com
		User From Name Filter	(&(cn=%u)(objectclass=person))
	Groups	Group Base DN	ou=Mygroups, dc=filenet.com
		Group From Name Filter	(&(cn=%g)((objectclass=groupOfNames))
		Static Group Name Attribute	cn
	Membership	Static Group DN's from Member DN Filter	(&(member=%M)(objectclass=groupOfNames))

GCD Configuration

For the example in the previous section, you need to create the following two DirectoryConfigurationNovell objects - one for each data naming context.

DirectoryConfigurationNovell object 1:

- DisplayName: cempdev1
- DirectoryServerHost: Ed-host
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=principal1
- DirectoryServerPassword: <password>
- IsSSLEnabled: false
- UserBaseDN: ou=people, o=cempdev1
- UserSearchFilter: (&(objectClass=person)(cn={0}))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=groups, o=cempdev1

- GroupSearchFilter: (&(cn=%g)(objectclass=groupOfNames))
- GroupMembershipSearchFilter: (&(member=%M)(objectclass=groupOfNames))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

DirectoryConfigurationNovell object 2:

- DisplayName: MyCompany
- DirectoryServerHost: Ed-host
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=principal2
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: ou=users,dc=filenet.com
- UserSearchFilter: (&(cn=%u)(objectclass=person))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=Mygroups, dc=filenet.com
- GroupSearchFilter: (&(cn=%g)((objectclass=groupOfNames))
- GroupMembershipSearchFilter: (&(member=%M)(objectclass=groupOfNames))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

Configuration Result

Authentication	FileNet P8 can authenticate any user by cn under: ou=people, o=cempdev1 ou=users,dc=filenet.com
Realm set	Two realms (i.e., all the data naming contexts): o=cempdev1 dc=filenet.com
User/group retrieval	FileNet P8 can retrieve any user under: ou=people, o= cempdev1 ou=users, dc=filenet.com FileNet P8 can retrieve any group under: ou=Mygroups, o= cempdev1 ou= groups, dc=filenet.com
Group membership	FileNet P8 can search group memberships:

	Within a realm Across realms.
--	----------------------------------

Operation

Get User or Group by Short Name

Iterate through all realms. For each realm:

3. Connect to corresponding host.
4. Search for the user/group by short name.
5. If found, return.

If more than one user/group is found, Content Engine will log an error and return the first user found.

Get User or Group by DN

6. Resolve the realm name from the DN.
7. Connect to corresponding host.
8. Search for the user/group by DN.

NOTE eDirectory does not support nested groups.

Get User or Group by UPN

eDirectory provider does not support these methods.

Get User or Group by SID

Iterate through all realms. For each realm:

9. Connect to corresponding host.
10. Search for the user/group by SID.
11. If found, return.

Search Users or Groups in a Given Realm

12. Connect to the host corresponding to the specified realm.
13. Search for the users or groups by the search criteria.

IBM Tivoli Directory Server

This topic describes FileNet P8's support for integrating with IBM Tivoli Directory Server.

General

One instance of IBM Tivoli Directory Server can have multiple data naming contexts. Because each data naming context is mapped to a Content Engine realm, one IBM Tivoli Directory Server can be mapped to multiple Content Engine realms.

For each realm, you need to create an application server authentication provider and a DirectoryConfigurationIBM object, so that there is a one-to-one relationship between Realm object and authentication provider, and also a one-to-one relationship between Realm object and DirectoryConfigurationIBM object.

For each authentication provider, FileNet P8 extracts the realm name from the specified User Base DN value by comparing it with each data naming context. For example, if this authentication provider's user base DN is "ou=people,o=isp", and if there are two data naming contexts: "o=isp" and "dc=filenet,dc=com", then you know the realm name for this authentication provider is "o=isp".

NOTE FileNet highly recommends that you configure SSL between your application server that hosts Content Engine and your Active Directory servers. This will include making changes in the application server to the authentication provider's DirectoryConfigurationIBM object that was created while running Content Engine Setup. Consult your application server's documentation for instructions.

Support matrix

Use this support matrix as a quick lookup of supported directory features.

IBM Tivoli Directory Server Features	Supported By Content Engine
One-way Secure Sockets Layer (SSL)	Y
Two-way SSL	N
Transport Layer Security (TLS)	N
Static Groups	Y
Dynamic Groups	N
Nested Groups	Y
Supported User type (objectClass)	inetOrgPerson
Supported Static Group types (objectClass)	groupOfUniqueNames, groupOfNames
Roles	N
Follow referrals for Search (e.g. User/Group retrieval)	N
Support multiple realms	Y
Chaining	Y

Directory aliases	N
Restrict to single realm	Y - Just configure one realm in application server
Configurable user short name attribute	Y - Because the short name does not contain realm information, short names must be unique across all your configured domains and realms.
Configurable group short name attribute	Y
Configurable user display name attribute	Y
Configurable group display name attribute	Y
Multiple authenticating attributes support	Y- See Configure multiple authenticating attributes.
Sorting	Y- Return users/groups in sorted order: either ascending or descending order.
Paging/Continuation	Y- Return users/groups page by page. Page continuation happens automatically in the back end.

Directory Configuration Properties

The following is an alphabetic list of the properties in the DirectoryConfigurationIBM class. Use Enterprise Manager to view all properties and modify editable properties.

Property Name	Editable?	Description
ClassDescription	N	A ClassDescription object containing the fixed description of the class from which a given object is instantiated.
DirectoryServerHost	Y	Specifies the name of the host that is running the directory server product.
DirectoryServerPassword	Y	Specifies the user password used to authenticate to a given directory server.
DirectoryServerPort	Y	Specifies the port number of the directory server. The value of this property defaults to port 389 for all supported directory server types.
DirectoryServerProviderClass	Y	Specifies the directory server provider class name: com.filenet.engine.security.IBMTivoliProvider
DirectoryServerType	N	Specifies the type of directory server: IBM
DirectoryServerUsername	Y	Specifies the username for authenticating to the directory server.
DisplayName	Y	The user-readable, provider-specific name of an object. This property is usually the designated Name property of the object's class.

GroupBaseDN	Y	The base DN for searching for groups in the directory server.
GroupDisplayNameAttribute	Y	Specifies the display name for a Group object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
GroupMembershipSearchFilter	Y	The search filter for group membership queries.
GroupNameAttribute	Y	Defines the directory server attribute to be used as the short name for a group.
GroupSearchFilter	Y	Specifies search filter for groups. Example: "(&(objectclass=group)(samAccountName={0}))", where samAccountName will serve as the short name.
Id	N	An object's globally unique ID (GUID).
IsSSLEnabled	Y	Defines whether or not Secure Sockets Layer (SSL) protocol is enabled for a given DirectoryConfiguration object. The default value is false, indicating that SSL is disabled.
UserBaseDN	Y	The base DN for searching for users in the directory server.
UserDisplayNameAttribute	Y	Specifies the display name for a User object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
UserNameAttribute	Y	Defines the directory server attribute to be used as the short name for a user.
UserSearchFilter	Y	Specifies search filter for users. Example: "(&(objectclass=user)(samAccountName={0}))", where samAccountName will serve as the short name.

Realm Configuration (WebLogic)

This section shows Realm configuration instructions for WebLogic. After the authentication providers are configured, you must restart WebLogic server.

When configuring WebLogic authentication providers, the LDAP attribute used for the fields "User Name Attribute" and "User From Name Filter" must be the same. The same applies to group configuration. Here is an example which uses samAccountName:

- User Name Attribute: uid
- User From Name Filter: (&(objectclass=user)(uid =%u))

For each IBM Tivoli Directory Server, check the attribute namingContexts in its root. Then find out how many data naming contexts it has. For each data naming context, create a WebLogic authentication provider by doing the following:

- In WebLogic, navigate to mydomain\Security\Realms\myrealm\Providers\Authentication
- Click "Configure a new Open LDAP Authenticator..." (WebLogic does not have an IBM Tivoli authenticator; use Open LDAP authenticator instead.)
- Name: <name>
- Control Flag: Sufficient
- Click "Create"
- Click the "Open LDAP" tab
- Host: your host running IBM Tivoli Directory Server
- Port: <port>
- Principal: <your principal>
- Credential: <password>
- Confirm Credential: <password>
- Click "Apply"
- Click the "Users" tab
- User Name Attribute: cn
- User Base DN: <your base DN. For example, ou=people,o=isp>
- User From Name Filter: (&(objectClass=person)(cn=%u))
- Click "Apply"
- Click the "Groups" tab
- Group Base DN: <your base DN. For example, ou=groups,o=isp>
- Group From Name Filter:
(&(cn=%g)((objectClass=groupOfNames)(objectClass=groupOfUniqueNames)))
- Static Group Name Attribute: cn
- Click "Apply"
- Click the "Membership" tab
- Static Group DN's from Member DN Filter :
(!(&(objectclass=groupOfNames)(member=%M))&(objectclass=groupOfUniqueNames)(uniqueMember=%M)))
- Click "Apply"
- Restart WebLogic
- Navigate to mydomain\Security\Realms\myrealm\Users (this may take awhile and typically will only show a partial list of users).
- Ensure you see users from your IBM Tivoli Directory Server.
- Navigate to mydomain\Security\Realm\myrealm\Groups (this may take awhile and typically will only show a partial list of groups).
- Ensure you see groups from your IBM Tivoli Directory Server.

Authentication Provider Configuration, using WebLogic

Assume an IBM Tivoli Directory Server has the following two data naming contexts:

- o=isp
- dc=mycompany,dc=com

You need to create two WebLogic authentication providers for it.

Provider Name	Tab	Field Name	Field Value
isp_cn	General	Control Flag	Sufficient
	Open LDAP	Host	MyHost1
		Port	389
		Principal	cn=root
		Credential	<password>
	Users	User Name Attribute	cn
		User Base DN	ou=people,o=isp
		User From Name Filter	(&(objectClass=person)(cn=%u))
	Groups	Group Base DN	ou=groups,o=isp
		Group From Name Filter	(&(cn=%g)((objectClass=groupOfNames)(objectClass=groupOfUniqueNames)))
		Static Group Name Attribute	cn
	Membership	Static Group DN's from Member DN Filter	(!(&(objectclass=groupOfNames)(member=%M))(&(objectclass=groupOfUniqueNames)(uniqueMember=%M)))
	MyCompany_cn	General	Control Flag
Open LDAP		Host	MyHost1
		Port	389
		Principal	cn=root
		Credential	<password>
Users		User Name Attribute	cn

		User Base DN	ou=users,dc=mycompany,dc=com
		User From Name Filter	(&(objectClass=person)(cn=%u))
	Groups	Group Base DN	ou=users,dc=mycompany,dc=com
		Group From Name Filter	(&(cn=%g)((objectClass=groupOfNames)(objectClass=groupOfUniqueNames)))
		Static Group Name Attribute	cn
	Membership	Static Group DN from Member DN Filter	((&(objectclass=groupOfNames)(member=%M))(&(objectclass=groupOfUniqueNames)(uniqueMember=%M)))

GCD Configuration

For the example in the previous section, you must create the following two DirectoryConfigurationIBM objects - one for each data naming context.

DirectoryConfigurationIBM object 1:

- DisplayName: isp
- DirectoryServerHost: MyHost1
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=root
- DirectoryServerPassword: <your password>
- IsSSLEnabled: false
- UserBaseDN: ou=people,o=isp
- UserSearchFilter: (&(objectClass=person)(cn={0}))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=groups,o=isp
- GroupSearchFilter: (&(cn={0})(|(objectClass=groupOfNames)(objectClass=groupOfUniqueNames)))
- GroupMembershipSearchFilter: (|(&(objectclass=groupOfNames)(member={0}))(&(objectclass=groupOfUniqueNames)(uniqueMember={0})))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

DirectoryConfigurationIBM object 2:

- DisplayName: MyCompany
- DirectoryServerHost: MyHost1
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=root
- DirectoryServerPassword: <your password>
- IsSSLEnabled: false
- UserBaseDN: ou=users,dc=mycompany,dc=com
- UserSearchFilter: (&(objectClass=person)(cn={0}))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=users,dc=mycompany,dc=com
- GroupSearchFilter: (&(cn={0})((objectClass=groupOfNames)(objectClass=groupOfUniqueNames)))
- GroupMembershipSearchFilter: ((&(objectclass=groupOfNames)(member={0}))(&(objectclass=groupOfUniqueNames)(uniqueMember={0})))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

Configuration Result

Authentication	FileNet P8 can authenticate any user by uid under: ou=people,o=isp ou=users,dc=mycompany,dc=com
Realm set	Two realms (i.e., all the data naming contexts): o=isp dc=mycompany,dc=com
User/group retrieval	FileNet P8 can retrieve any user or group under: ou=people,o=isp ou=users,dc=mycompany,dc=com FileNet P8 can retrieve any group under: ou=groups,o=isp ou=users,dc=mycompany,dc=com
Group membership	FileNet P8 can search group memberships: Within a realm Across realms.

Operation

Get User or Group by Short Name

Iterate through all realms. For each realm:

14. Connect to corresponding host.
15. Search for the user/group by short name.
16. If found, return.

If more than one user/group is found, Content Engine will log an error and return the first user found.

Get User or Group by DN

17. Resolve the realm name from the DN.
18. Connect to corresponding host.
19. Search for the user/group by DN.

Get User or Group by UPN

IBM Tivoli provider does not support these methods.

Get User or Group by SID

Iterate through all realms. For each realm:

20. Connect to corresponding host.
21. Search for the user/group by SID.
22. If found, return.

Search Users or Groups in a Given Realm

23. Connect to the host corresponding to the specified realm.
24. Search for the users or groups by the search criteria.

Windows Active Directory

This topic describes FileNet P8's support for integrating with Windows Active Directory.

NOTE FileNet highly recommends that you configure SSL between your application server that hosts Content Engine and your Active Directory servers. This will include making changes in the application server to the authentication provider's DirectoryConfigurationAD object that was created while running Content Engine Setup. Consult your application server's documentation for instructions.

Support matrix

Use this support matrix as a quick lookup of supported directory features.

Active Directory Features	Supported By Content Engine
One-way SSL	Y
Two-way SSL	N
Universal Groups	Y
Security Groups	Y
Distribution Groups	Y
Nested Groups	Y
Builtin Groups	N
Users and groups belonging to custom Active Directory objects	Y
Supported User type (objectClass)	user
Supported Static Group types (objectClass)	group
Follow referrals for Search (for User/Group retrieval)	N
Roles	N
Directory aliases	N
Native Mode Active Directory	Y
Mixed Mode Active Directory	Y – No support for NT4.
Restrict to single realm	Y - By configuring just one realm.
Support multiple realms/domains	Y

Support multiple forests	Y
Support users/groups migrate from domain to domain within a forest	N
Support domains across multiple forests	Y
Configurable user short name attribute	Y - Because the short name does not contain realm information, short names must be unique across all your configured domains and realms.
Configurable group short name attribute	Y
Configurable user display name attribute	Y
Configurable group display name attribute	Y
Configurable principal Name - Boolean flag	Y If true: shortname@authentication.domain If false: full DN
DNS Site	Y – Resolve domain controllers in a given DNS site.
Multiple authenticating attributes support	Y – Can authenticate against the same Active Directory server using multiple attributes, such as samAccountName, userPrincipalName, or distinguishedName. See Configure multiple authenticating attributes.
Sorting	Y – Return users/groups in sorted order: either ascending or descending order.
Paging/Continuation	Y – Return users/groups page by page. Page continuation happens automatically in the back end.
Windows NT domains (versions 4.0 and earlier).	N
Group search returns Domain Local Groups	Y

Directory Configuration Properties

The following is an alphabetic list of the properties in the DirectoryConfigurationAD class. Use Enterprise Manager to view all properties and modify editable properties.

Property Name	Editable?	Description
ClassDescription	N	A ClassDescription object containing the fixed description of the class from which a given object is instantiated.
ConnectionTimeout	Y	Specifies the Active Directory Service provider connection timeout in milliseconds. The default is 500 milliseconds. If the connection is across a WAN, consider increasing the value.
DirectoryServerHost	Y	Specifies the name of the host that is running the directory server product.
DirectoryServerPassword	Y	Specifies the user password used to authenticate to a given directory server.
DirectoryServerPort	Y	Specifies the port number of the directory server. The value of this property defaults to port 389 for all supported directory server types.
DirectoryServerProvider Class	Y	Specifies the directory server provider class name: com.filenet.engine.security.ActiveDirectoryProvider
DirectoryServerType	N	Specifies the type of directory server: AD
DirectoryServerUserName	Y	Specifies the username for authenticating to the directory server. Example: "CN=test1,CN=Users,DC=myCompany,DC=com"
DisplayName	Y	The user-readable, provider-specific name of an object. This property is usually the designated Name property of the object's class.
GroupBaseDN	Y	The base DN for searching for groups in the directory server.
GroupDisplayNameAttribute	Y	Specifies the display name for a Group object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
GroupMembershipSearchFilter	Y	The search filter for group membership queries.
GroupNameAttribute	Y	Defines the directory server attribute to be used as the short name for a group.
GroupSearchFilter	Y	Specifies search filter for groups. Example: "(&(objectclass=group)(samAccountName={0}))", where samAccountName will serve as the short name.
Id	N	An object's globally unique ID (GUID).
IsSSEnabled	Y	Defines whether or not Secure Sockets Layer (SSL) protocol is enabled for a given DirectoryConfiguration object. The default value is false, indicating that SSL is disabled.

ReturnNameAsDN	Y	Specifies whether to return the user or group name in Distinguished Name (DN) format for Active Directory Service provider. By default, the Active Directory Service provider returns the user and group names in UPN format. If set to true, the service provider returns the names in DN format, which is consistent with other types of directory service providers.
SearchCrossForestGroupMembership	Y	Specifies whether the Active Directory Service provider performs cross-forest group membership searches. The default is false. To enable cross-forest group membership searches, set this property to true.
UserBaseDN	Y	The base DN for searching for users in the directory server.
UserDisplayNameAttribute	Y	Specifies the display name for a User object generated by the authentication provider. The default property value is dependent on the authentication provider and is specified by the provider's configuration.
UserNameAttribute	Y	Defines the directory server attribute to be used as the short name for a user.
UserSearchFilter	Y	Specifies search filter for users. Example: "(&(objectclass=user)(samAccountName={0}))", where samAccountName will serve as the short name.

Realm Configuration (WebLogic)

Active Directory consists of forests, each forest consisting of trees, and each tree consisting of domains. Each domain has one data naming context, called the default naming context. Each Active Directory domain equate to one Content Engine realm, and the value of the default naming context attribute is the name of the Content Engine realm. For example:

DefaultNamingContext: DC=paperwork2,DC=eng,DC=filenet,DC=com
 Content Engine realm name: DC=paperwork2,DC=eng,DC=filenet,DC=com

For an Active Directory forest, all its domain controllers register to one DNS server. This DNS server normally runs on a domain controller of the root domain. Make sure the machine on which your application server is running points to this DNS server.

Active Directory service provider can be configured in three different levels: single-realm, multi-realm, and multi-forest. Each is explained in the following sections.

Single-Realm Configuration

This section demonstrates LDAP configuration for a domain named "paperwork2" and uses WebLogic 8.1 for application server examples. Any field that is not listed in the following table takes its default value.

Authentication Provider Configuration for single-realm, using WebLogic

Provider Name	Tab	Field Name	Field Value
domain_samAccountName	General	Control Flag	Sufficient
	Active	Host	Host_name

	Directory	Port	389
		Principal	CN=test1,CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
		Credential	<password>
	Users	User Name Attribute	sAMAccountName
		User Base DN	CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
		User From Name Filter	(&(objectclass=user)(sAMAccountName=%u))
	Groups	Group Base DN	CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName
domain_userPrincipalName	General	Control Flag	Sufficient
	Active Directory	Host	
		Port	389
		Principal	
		Credential	<password>
	Users	User Base Attribute	userPrincipalName
		User Base DN	CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
		User From Name Filter	(&(objectclass=user)(userPrincipalName=%u))
	Groups	Group Base DN	CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName

GCD Configuration for single-realm

com.filenet.api.admin.DirectoryConfigurationAD

- DisplayName: paperwork2
- DirectoryServerHost: Fileclerk2
- DirectoryServerPort: 389
- DirectoryServerUserName: CN=test1,CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: false

Configuration Result for single-realm

Authentication	FileNet P8 can authenticate any user in this domain under CN=Users,DC=domain,DC=eng,DC=filenet,DC=com. FileNet P8 can authenticate the users by either samAccountName or userPrincipalName.
Realm set	One realm: DC=paperwork2,DC=eng,DC=filenet,DC=com
User/Group retrieval	FileNet P8 can retrieve any user or group in this realm under CN=Users,DC=paperwork2,DC=eng,DC=filenet,DC=com
Group Membership	FileNet P8 can search any group membership in this local domain. But if the membership involves other domains, an error is returned.

Multi-Realm Configuration

This section configures all or some domains in a single forest.

In a single forest, if a cross-domain group membership involves any Domain Local Group, the group membership is one-way, and it is not replicated to the Global Catalog. The forward group membership search (given a group, find all its member groups and member users) is easy. Given a group in one domain, you can easily find all its members in other domains. However, the backward group membership search (given a group or user, find all parent groups that it belongs to) is expensive. You must iterate

through all other domains to find all the parent groups to which this group belongs. Since large enterprises may have 30 or more domains in one forest, this iterative approach for cross-domain backward group membership search is too expensive and so is not acceptable.

As a result, you should split this cross-domain membership and add an extra group in between. Let's say a domain local group in Domain A contains a global group in Domain B. You can create a universal group in Domain A, make the domain local group containing the universal group its member, and then make the universal group in Domain A to contain the global group in Domain B. This is because:

- Group membership in the same domain is two-way, regardless the group scope.
- If the cross-domain group membership does not involve domain local group, it is two-way in the Global Catalog.

Entire Forest Configuration

This section assumes Content Engine accesses every domain in a forest. This is a quick and easy configuration.

Authentication Provider Configuration for entire forest, using WebLogic

Create one Active Directory authentication provider in WebLogic, and then make the following changes:

1. In "Active Directory" tab, for the "Host" field, pick a domain controller which has the Global Catalog (GC) server running on it.
2. Change the "Port" to 3268 which is the port number of Global Catalog server on that host.
3. For the "Principal" field, make sure you pick a user who can be authenticated in every domain in the forest, and has read-access to any group and user in the forest. This principal will be used to access every domain in the forest. Normally, a user in "Domain Admins" or "Enterprise Admins" group is a good choice.
4. In the "Users" tab, set "User Base DN" field to blank. This sets the user base DN to the root of the GC. Therefore, this provider can authenticate any user in this forest.
5. In the "Group" tab, set "Group Base DN" field to blank. This sets the group base DN to the root of the GC.

Example: assume you have a forest consisting of the following three domains:

- mycompany.com
- ca.mycompany.com
- wa.mycompany.com

"mycompany.com" is the root domain of this forest. "mycompany-dc1" is the host name of a domain controller for the root domain, and has global catalog server running on it. Here is the provider for this entire forest:

Provider Name	Tab	Field Name	Field Value
MyCompany_samAccountName	General	Control Flag	Sufficient
	Active Directory	Host	<i>Host_name</i>
		Port	3289
		Principal	CN=Administrator,CN=Users,DC=mycompany,DC=com

		Credential	<password>
Users		User Name Attribute	sAMAccountName
		User Base DN	<Set to blank>
		User From Name Filter	(&(objectclass=user)(sAMAccountName=%u))
Groups		Group Base DN	<Set to blank>
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName

GCD Configuration for entire forest

Here is the GCD configuration for the example in the above section.

com.filenet.api.admin.DirectoryConfigurationAD

- DisplayName: MyCompany
- DirectoryServerHost: mycompany-dc1
- DirectoryServerPort: 3268
- DirectoryServerUserName: CN=Administrator,CN=Users,DC=mycompany,DC=com
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: null
- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: null
- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: false

When the Active Directory service provider sees that this DirectoryConfiguration object is pointing to a global catalog server of a forest (by checking the port), it will pick up this one and ignore all other DirectoryConfiguration objects configured for the same forest. It will search in the GC for all domains in this forest, and treat each domain as a Content Engine realm. The domain distinguished name will be used as the realm name, the user base DN, and group base DN for this realm.

When searching for domains, Active Directory provider does not check any trust relationship. By definition, if a domain joins any forest, it has to establish two-way transitive trust relationships with other domains in the forest.

Configuration Result for entire forest

Authentication	FileNet P8 can authenticate any user by samAccountName in any domain within this single forest.
Realm set	Three realms (that is, all the domains in this forest): DC=mycompany,DC=com DC=ca,DC=mycompany,DC=com DC=wa,DC=mycompany,DC=com
User/Group retrieval	FileNet P8 can retrieve any user or group in any domain within this forest.
Group Membership	FileNet P8 can search the following two types of group membership. Local-domain group membership. Cross-domain group membership (within this forest) If the membership involves other forests and flag SearchCrossForestGroupMembership is enabled, make sure you have other forests configured as well. Otherwise, an error is returned.

Partial Forest Configuration

In some cases, you may not want every domain in the forest to be accessed by Content Engine. For example, a company might have 30 domains in its forest, but only 3 of them should be accessed by Content Engine. In this case, you can create 3 authentication providers in the WebLogic server. Each provider points to one of the domains accessed by Content Engine. At the same time, you create three DirectoryConfiguration objects to point to the same set of domains.

Authentication Provider Configuration for partial forest

The steps are the same as mentioned in single realm, except you create multiple providers here. Make sure none of them points to a Global Catalog server.

Example: using the example in the previous section, among all the domains in the forest, you want only the following two to be accessed by Content Engine system:

- mycompany.com
- ca.mycompany.com

So you create two WebLogic authentication providers as shown below. Assume "ca-dc1" is the host name of a domain controller for domain "ca.mycompany.com".

Provider Name	Tab	Field Name	Field Value
MyCompany_samAccountName	General	Control Flag	Sufficient
	Active	Host	mycompany-dc1

	Directory	Port	389
		Principal	CN=Administrator,CN=Users,DC=mycompany,DC=com
		Credential	<password>
	Users	User Name Attribute	sAMAccountName
		User Base DN	DC=mycompany,DC=com
		User From Name Filter	(&(objectclass=user)(sAMAccountName=%u))
	Groups	Group Base DN	DC=mycompany,DC=com
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName
CA_sAMAccountName	General	Control Flag	Sufficient
	Active Directory	Host	ca-dc1
		Port	389
		Principal	CN=Administrator,CN=Users,DC=ca,DC=mycompany,DC=com
		Credential	<password>
	Users	User Name Attribute	sAMAccountName
		User Base DN	DC=ca,DC=mycompany,DC=com
		User From Name Filter	(&(objectclass=user)(sAMAccountName=%u))
	Groups	Group Base DN	DC=ca,DC=mycompany,DC=com
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name	sAMAccountName

		Attribute	
--	--	-----------	--

GCD Configuration for partial forest

The steps are the same as mentioned in GCD configuration for single realm, except you create multiple DirectoryConfiguration objects. Make sure none points to a Global Catalog server.

Here is the GCD configuration for the example in the above section:

DirectoryConfigurationAD object 1:

- DisplayName: MyCompany
- DirectoryServerHost: mycompany-dc1
- DirectoryServerPort: 389
- DirectoryServerUserName: CN=Administrator,CN=Users,DC=mycompany,DC=com
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: DC=mycompany,DC=com
- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: DC=mycompany,DC=com
- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: false

DirectoryConfigurationAD object 2:

- DisplayName: ca
- DirectoryServerHost: ca-dc1
- DirectoryServerPort: 389
- DirectoryServerUserName: CN=Administrator,CN=Users,DC=ca,DC=mycompany,DC=com
- DirectoryServerPassword: <password>
- IsSSEnabled: false
- UserBaseDN: DC=ca,DC=mycompany,DC=com
- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: DC=ca,DC=mycompany,DC=com

- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: false

Configuration Result for partial forest

Authentication	FileNet P8 can authenticate any user in any of these two domains.
Realm set	Two realms: DC=mycompany,DC=com DC=ca,DC=mycompany,DC=com
User/group retrieval	FileNet P8 can retrieve any user or group in these two domains.
Group membership	FileNet P8 can search the following two types of group membership. Local-domain group membership. Group membership across these two domains. If the membership involves other domains in this forest, make sure you have other domains configured as well. Otherwise, an error is returned. If the membership involves other forests and flag SearchCrossForestGroupMembership is enabled, make sure you have other forests configured as well. Otherwise, an error is returned.

Multi-Forest Configuration

Let's say you have two forests. You can follow steps specified in Entire Forest to create two entire forest configurations. Or you can follow the steps in Partial Forest to create the partial forest configuration for each forest. Or you can create a mixture of these two configurations.

Example: assume you have the following two forests. Note that forest name is the root domain name.

Forest	Domain	Domain Controller
mycompany.com	This forest consists of the following domains. mycompany.com (root domain) ca.mycompany.com wa.mycompany.com	"mycompany-dc1" is the host name of a domain controller for the root domain, and has global catalog server running on it.
mynewcompany.com	This forest consists of the following domains. mynewcompany.com (root domain) hr.mynewcompany.com	"mynewcompany-dc1" is the host name of a domain controller for the root domain, and has global catalog server running on it.

Authentication Provider Configuration for multi-forest

Below are the authentication providers for these two forests. Each provider covers all domains in the corresponding forest.

Provider Name	Tab	Field Name	Field Value
MyCompany_samAccountName	General	Control Flag	Sufficient
	Active Directory	Host	mycompany-dc1
		Port	3268
		Principal	CN=Administrator,CN=Users,DC=mycompany,DC=com
		Credential	<password>
	Users	User Name Attribute	sAMAccountName
		User Base DN	<Set to blank>
		User From Name Filter	(&(objectclass=user)(sAMAccountName=%u))
	Groups	Group Base DN	<Set to blank>
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName
	MyNewCompany_samAccountName	General	Control Flag
Active Directory		Host	mynewcompany-dc1
		Port	3268
		Principal	CN=Administrator,CN=Users,DC=mynewcompany,DC=com
		Credential	<password>
Users		User Base Attribute	sAMAccountName
		User Base DN	<Set to blank>
		User From	(&(objectclass=user)(sAMAccountName=%u))

		Name Filter)
	Groups	Group Base DN	<Set to blank>
		Group From Name Filter	(&(sAMAccountName=%g)(objectclass=group))
		Static Group Name Attribute	sAMAccountName

GCD Configuration for multi forest

Here are the two DirectoryConfiguration objects for the above two forests:

DirectoryConfigurationAD object 1:

- DisplayName: MyCompany
- DirectoryServerHost: mycompany-dc1
- DirectoryServerPort: 3268
- DirectoryServerUserName: CN=Administrator,CN=Users,DC=mycompany,DC=com
- DirectoryServerPassword: <password>
- IsSSLEnabled: false
- UserBaseDN: null
- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: null
- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: true

DirectoryConfigurationAD object 2:

- DisplayName: MyNewCompany
- DirectoryServerHost: mynewcompany-dc1
- DirectoryServerPort: 3268
- DirectoryServerUserName: CN=Administrator,CN=Users,DC=mynewcompany,DC=com
- DirectoryServerPassword: <password>
- IsSSLEnabled: false
- UserBaseDN: null

- UserSearchFilter: (&(objectClass=user)(samAccountName={0}))
- UserNameAttribute: samAccountName
- UserDisplayNameAttribute: cn
- GroupBaseDN: null
- GroupSearchFilter: (&(objectClass=group)(samAccountName={0}))
- GroupMembershipSearchFilter: null
- GroupNameAttribute: samAccountName
- GroupDisplayNameAttribute: cn
- SearchCrossForestGroupMembership: true

Configuration Result for multi forest

Authentication	FileNet P8 can authenticate any user by samAccountName in any of these two forests.
Realm set	Five realms (that is, all the domains in these two forests): <ul style="list-style-type: none"> • DC=mycompany,DC=com • DC=ca,DC=mycompany,DC=com • DC=wa,DC=mycompany,DC=com • DC=mynewcompany,DC=com • DC=hr,DC=mynewcompany,DC=com
User/group retrieval	FileNet P8 can retrieve any user or group in any of these two forests
Group membership	FileNet P8 can search the following three types of group membership. <ul style="list-style-type: none"> • Local-domain group membership. • Cross-domain group membership. • Cross-forest group membership.

Operation

Get User or Group by Short Name

Iterate through all forests. For each forest:

Connect to GC and search for DN by the short name.

Resolve the domain name from the DN.

Connect to the domain and search for the user/group by DN.

If group membership is asked for, FileNet P8 searches for it in the local domain first; then searches for it again in the GC. In the end, it combines the results.

If the multi-forest support flag is on and the group membership is asked for, FileNet P8 searches for it in all other forests.

If more than one user/group is found, Content Engine logs an error and returns the first user found.

Get User or Group by DN

- Resolve the domain name from the DN.
- Connect to the domain and search for the user/group by DN.
- If group membership is asked for, FileNet P8 searches for it in the local domain first; then searches for it again in the GC. In the end, FileNet P8 combines the results.
- If the multi-forest support flag is turned on and the group membership is asked for, FileNet P8 searches all forests.

Get User or Group by UPN

- Resolve the domain name from the UPN.
- Get short name from UPN.
- Connect to the domain and search for the user/group by short name.
- If group membership is asked for, FileNet P8 search it in the local domain first; then search it again in GC. In the end, FileNet P8 combines the results.
- If the multi-forest support flag is turned on and the group membership is asked for, FileNet P8 will search for it in all other forests.

Note: Microsoft defines the user principal name (UPN) format to consist of the user name, the "at" sign (@), and a user principal name suffix. In Content Engine, the user name part is always the short name, and the suffix part is always the DNS domain name of the domain the user belongs to.

Get User or Group by SID

- Resolve the domain name from the SID. Part of user/group SID is its domain SID. FileNet P8 maintains a mapping between domain SID and domain name.
- Connect to the domain and search for the user/group by SID.
- If group membership is asked for, FileNet P8 searches for it in the local domain first; then searches for it again in the GC. In the end, FileNet P8 combines the results.
- If the multi-forest support flag is turned on and the group membership is asked for, FileNet P8 searches all forests.

Search Users or Groups in a Given Realm

- Connect to the domain specified by the realm name.
- Search for the users or groups by the search criteria.
- For each user/group, if group membership is asked for, FileNet P8 searches for it in the local domain first; then searches for it again in the GC. In the end, FileNet P8 combines the results.
- If the multi-forest support flag is turned on and the group membership is asked for, FileNet P8 searches all forests.

Windows Active Directory Application Mode (ADAM)

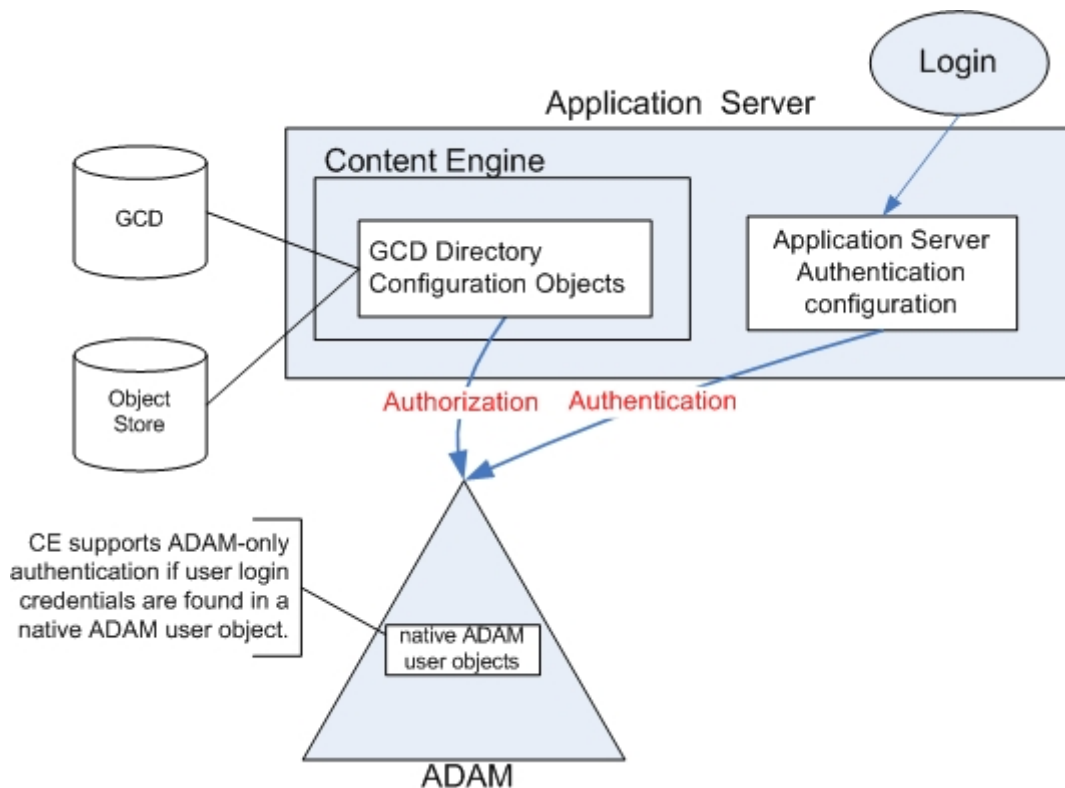
This topic describes FileNet P8's support for integrating with Windows Active Directory Application Mode (ADAM).

One instance of ADAM can have multiple application partitions, each of which can be mapped to a Content Engine (CE) realm. Therefore one instance of ADAM can be mapped to multiple CE realms.

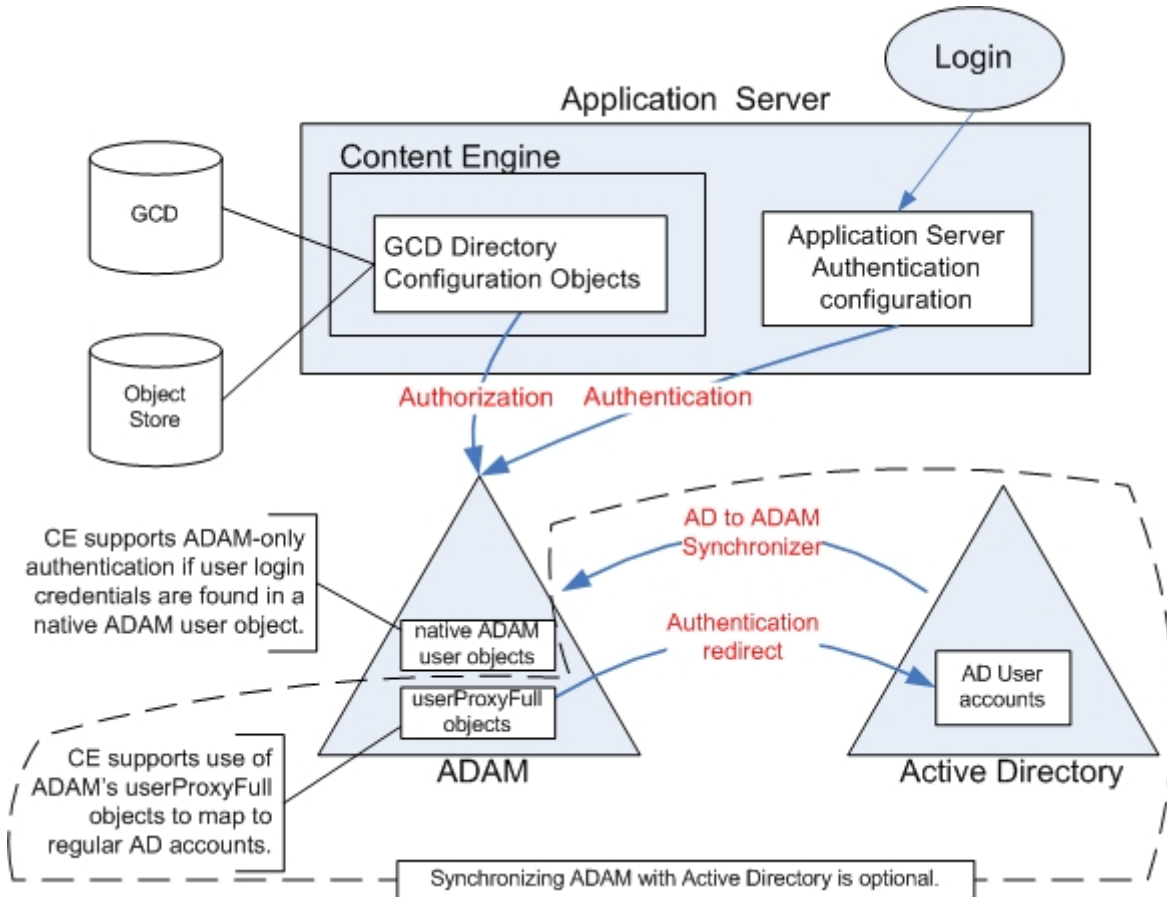
For each realm, you must create an application server authentication provider and a DirectoryConfigurationADAM object, to establish a one-to-one relationship between Realm object and authentication provider, and also a one-to-one relationship between Realm object and DirectoryConfigurationADAM object. The initial set of these objects is created during CE installation.

For each DirectoryConfiguration object, FileNet P8 extracts the realm name from the specified UserBaseDN property value by comparing it with each application partition. For example, if the UserBaseDN for this DirectoryConfiguration object is "ou=people, o=isp ", and there are two application partitions: "o=isp" and "dc=filenet,dc=com", the realm name for this DirectoryConfiguration object is "o=isp".

The following graphic shows CE authenticating with ADAM:



The next graphic shows the optional configuration of CE authenticating with ADAM configured for proxy login and search to Active Directory. When a user logs in using an ID found in a userProxyFull object, ADAM redirects authentication to Active Directory.



You can optionally use the Synchronizer tool, a built-in feature of ADAM, to pull user account information from Active Directory. In this scenario, ADAM user accounts are represented using the userProxyFull object, which stores the user ID while the account password remains in Active Directory. When properly configured this provides one-way data flow from Active Directory to ADAM. You could continue to provision ADAM-only accounts in ADAM, and both types of accounts could authenticate to a FileNet P8 application, following normal configuration of CE classes' Default Instance Security tabs in Enterprise Manager. The application does not need to be aware of this Active Directory interaction.

Consult your ADAM documentation for how to use the userProxyFull object. Content Engine does not support ADAM's userProxy object.

NOTE FileNet highly recommends that you configure SSL between your application server that hosts CE and your ADAM servers. This will include making changes in the application server to the authentication provider's DirectoryConfigurationADAM object that was created while running CE Setup. Consult your application server's documentation for instructions.

Support matrix

Use this support matrix as a quick lookup of supported directory features.

ADAM Features	Supported By Content Engine
One way SSL	Y
Two way SSL	N
Static Groups / Security Groups	Y

Nested Groups	Y
Dynamic Groups	n/a
Universal Groups	n/a
Roles	N Roles are not used by FileNet P8 services and are not part of the LDAP standard. Do not confuse this "Roles" with the ADAM "Roles" container which is just a container of groups.
Referrals for Logon	N
Referrals for Search (for User and Group retrieval)	N
Chaining	N
Directory aliases	N
Native Mode Active Directory	n/a
Mixed Mode Active Directory	n/a
Support multiple realms	Y - Each realm corresponds to one ADAM application partition. P8 4.0.x support for multiple realms depends on the application server. As of the 4.0.x release, multiple realm support is available for WebLogic, JBoss, and WebSphere 6.1 (by way of federated user repositories), but not for WebSphere 6.0 or 5.1. See Configure multiple realms.
Restrict to single realm	Y By configuring just one authentication provider and one directory configuration.
Support domains across multiple forests	n/a
Logon to any W2k domain in the forest (implies 2-way trust)	n/a
Logon to NT 1 way trust domains in the forest	n/a
Configurable username for logon	Y The short or common name does not contain realm information. Short names must be unique across all of your configured application partitions and realms.

Configurable user display name	Y
Configurable group display name	Y
Configurable group name for persisting	Y Group names are not persisted in the CE database, even though they are persisted in stored searches and workflow definitions.
Support ADAM users (for logon and Search)	Y
Support use of userProxyFull class (for logon and Search)	Y You can create a special user proxy object in ADAM that maps to a regular Active Directory user account. The user proxy does not have an actual password stored in the ADAM object itself. When the application performs its normal bind operation, it checks the ID locally, but checks the password against Active Directory. Content Engine requires the use of the ADAM userProxyFull class, rather than UserProxy.
Support Windows (domain & local) users (logon and Search)	N
Users in Application Partitions	Y
Users in Configuration and Schema partitions	N There is a patch from Microsoft that allows ADAM users to reside in the Configuration partition. However, FileNet P8 does not support this.

Directory Configuration Properties

The following is an alphabetic list of the properties in the DirectoryConfigurationADAM class with default values. Use Enterprise Manager to view all properties and modify editable properties. See FileNet P8 domain properties (Directory config tab) for information.

Property Name	Editable?	Description
ClassDescription	N	A ClassDescription object containing the fixed description of the class from which a given object is instantiated.
DirectoryServerHost	Y	Specifies the name of the host that is running the directory server product.
DirectoryServerPassword	Y	Specifies the user password used to authenticate to a given directory server.
DirectoryServerPort	Y	Specifies the port number of the directory server. The value

		of this property defaults to port 389 for all supported directory server types.
DirectoryServerProviderClass	Y	Specifies the directory server provider class name: com.filenet.engine.security.AdamDirectoryProvider
DirectoryServerType	N	Specifies the type of directory server: ADAM
DirectoryServerUserName	Y	Specifies the username for authenticating to the directory server. Example: cn=ceadmin,ou=people,o=isp
DisplayName	Y	The user-readable, provider-specific name of an object. This property is usually the designated Name property of the object's class.
GroupBaseDN	Y	The base DN for searching for groups in the directory server. Example: ou=people,o=isp
GroupDisplayNameAttribute	Y	Specifies the display name for a Group object generated by the authentication provider: cn
GroupMembershipSearchFilter	Y	The search filter for group membership queries: (&(objectClass=group)(member={0})).
GroupNameAttribute	Y	Defines the directory server attribute to be used as the short name for a group: cn
GroupSearchFilter	Y	Specifies search filter for groups. Example: (&(objectclass=group)(cn={0}))
Id	N	An object's globally unique ID (GUID).
IsSSLEnabled	Y	Defines whether or not Secure Sockets Layer (SSL) protocol is enabled for a given DirectoryConfiguration object. The default value is false, indicating that SSL is disabled.
UserBaseDN	Y	The base DN for searching for users in the directory server. Example: ou=people,o=isp
UserDisplayNameAttribute	Y	Specifies the display name for a User object generated by the authentication provider: cn
UserNameAttribute	Y	Defines the directory server attribute to be used as the short name for a user: cn
UserSearchFilter	Y	Specifies search filter for users: (&(objectClass=person)(cn={0})) This filter find both native ADAM accounts and Active Directory accounts referenced by the userProxyFull object.

Realm Configuration

Authentication Provider Configuration

This section shows Realm configuration instructions for WebLogic server. After the authentication providers are configured, you need to restart WebLogic server.

When configuring WebLogic authentication providers, the LDAP attributes used for fields “User Name Attribute” and “User From Name Filter” must be the same. This rule also applies to group configuration. Here is an example which uses cn:

- User Name Attribute: cn
- User From Name Filter: (&(objectclass=person)(cn =%u))

By default, the User From Name Filter takes the form of: (&(objectclass=person)(cn =%u)). This supports both native Adam users and Adam UserProxyFull users. If you have only native Adam users, you can safely change the User From Name Filter to: (&(objectclass=user)(cn =%u)). This rule applies for both the J2EE application server authenticator and the GCD configurations described below.

For each ADAM directory server, determine how many application partitions it has. Each application partition corresponds to one data naming context. For each application partition, create a WebLogic authentication provider by performing the following steps. Since WebLogic doesn't have an ADAM Authenticator template built in, you will modify its OpenLdap Authenticator in the following examples:

- In WebLogic, navigate to mydomain\Security\Realms\myrealm\Providers\Authentication
- Click “Configure a new Open LDAP Authenticator...”
- Name: <name>
- Control Flag: Sufficient
- Click “Create”
- Click the “OpenLdap” tab
- Host: your host running ADAM
- Port: <port>
- Principal: your ldap search user. For example, cn=ceadmin,dc=dev1,dc=filenet,dc=com
- Credential: <password>
- Confirm Credential: your password
- Click “Apply”
- Click the “Users” tab
- User Name Attribute: cn
- User Base DN: your base DN. For example, dc=dev1,dc=filenet,dc=com
- User From Name Filter: (&(objectClass=person)(cn=%u))
- Click “Apply”
- Click the “Groups” tab
- Group Base DN: your base DN. For example, dc=dev1,dc=filenet,dc=com Group From Name Filter: (&(objectClass=group)(cn=%g))
- Static Group Name Attribute: cn
- Click “Apply”

- Click the “Membership” tab
- Static Group DNs from Member DN Filter: (&(objectClass=group)(member=%M))
- Click “Apply”
- Restart WebLogic
- Navigate to mydomain\Security\Realms\myrealm\Users (this could take awhile and typically will only show a partial list of users).
- Ensure you see users from your ADAM server.
- Navigate to mydomain\Security\Realm\myrealm\Groups (this could take awhile and typically will only show a partial list of groups).
- Ensure you see groups from your ADAM server.

Example: Assume an ADAM server has the following two application partitions:

- o=isp
- dc=mycompany,dc=com

You would need to create two WebLogic authentication providers for this configuration.

Provider Name	Tab	Field Name	Field Value
Adam1	General	Control Flag	Sufficient
	Adam LDAP	Host	<Adam-host>
		Port	389
		Principal	cn=ceadmin,ou=people,o=isp
		Credential	<password>
	Users	User Base Attribute	cn
		User Base DN	ou=people,o=isp
		User From Name Filter	(&(objectClass=person)(cn=%u))
	Groups	Group Base DN	ou=people,o=isp
		Group From Name Filter	(&(objectClass=group)(cn=%g))
		Static Group Name Attribute	cn
	Membership	Static Group DNs from Member DN Filter	(&(objectClass=group)(member=%M))
	Adam2	General	Control Flag
Adam LDAP		Host	<Adam-host>
		Port	389

		Principal	cn=ceadmin,ou=users,dc=mycompany,dc=com
		Credential	<password>
	Users	User Base Attribute	cn
		User Base DN	ou=users,dc=mycompany,dc=com
		User From Name Filter	(&(uid=%u)(objectclass=user))
	Groups	Group Base DN	ou=users,dc=mycompany,dc=com
		Group From Name Filter	(&(objectClass=group)(cn=%g))
		Static Group Name Attribute	cn
	Membershi p	Static Group DNs from Member DN Filter	(&(objectClass=group)(member=%M))

GCD Configuration

For the example in the previous section, you must now create two DirectoryConfigurationAdam objects – one object for each data naming context. See Administer directory configurations for information.

DirectoryConfigurationAdam object 1:

- DisplayName: Adam1
- DirectoryServerHost: <adam1-host>
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=ceadmin,ou=people,o=isp
- DirectoryServerPassword: <password>
- IsSSLEnabled: false
- UserBaseDN: ou=people,o=isp
- UserSearchFilter: (&(objectClass=person)(cn={0}))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=people,o=isp
- GroupSearchFilter: (&(objectClass=group)(cn={0}))
- GroupMembershipSearchFilter: (&(objectClass=group)(member={0}))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

DirectoryConfigurationAdam object 2:

- DisplayName: Adam2

- DirectoryServerHost: <adam2-host>
- DirectoryServerPort: 389
- DirectoryServerUserName: cn=ceadmin,ou=users,dc=mycompany,dc=com
- DirectoryServerPassword: <password>
- IsSSLEnabled: false
- UserBaseDN: ou=users,dc=mycompany,dc=com
- UserSearchFilter: (&(objectClass=person)(cn={0}))
- UserNameAttribute: cn
- UserDisplayNameAttribute: cn
- GroupBaseDN: ou=users , dc=mycompany,dc=com
- GroupSearchFilter: (&(objectClass=group)(cn={0}))
- GroupMembershipSearchFilter: (&(objectClass=group)(member={0}))
- GroupNameAttribute: cn
- GroupDisplayNameAttribute: cn

Configuration Result

Authentication	FileNet P8 can authenticate any user by cn under: ou=people,o=isp ou=users,dc=mycompany,dc=com
Realm set	Two realms (that is, all the data naming contexts): o=isp dc=mycompany,dc=com
User/Group retrieval	FileNet P8 can retrieve any user under: ou=people,o=isp ou=users,dc=mycompany,dc=com FileNet P8 can retrieve any group under: ou=groups,o=isp ou=users,dc=mycompany,dc=com

Configuration with SSL

Here are the steps for configuring the SSL connection between WebLogic server and ADAM directory server.

- Install server certificates into your directory server, and enable its SSL port. Check ADAM documentation for instructions.
- For CE authentication configuration. Configure your application server to use a trust key store that contains the CA certificate for your directory server. This is application server type specific.

For WebLogic, it is under the “Keystores & SSL” tab of your application server instance in admin console.

- WebLogic → Authentication Provider → openLdap tab → Host field : If the host name in your server certificate is fully qualified host name, you have to change this field be fully qualified host name as well.
- WebLogic → Authentication Provider → openLdap tab → Port field : Change it to the SSL port number where your directory server is listening.
- WebLogic → Authentication Provider → openLdap tab → SSL Enabled : Select it.
- For CE authorization configuration. Configure your Java standard Trust key store (JAVA_HOME\jre\lib\security\cacerts) to include the CA certificate for your directory server. You can use keytool.exe (under JAVA_HOME/bin/ folder).
- If you are no using java standard Trust key store, then you need to specify the trust store info as following in the Java command line of your application server startup script.

```
- Djavax.net.ssl.trustStore=<path_to_trust_store_file>  
- Djavax.net.ssl.trustStorePassword=<password_of_the_trust_store>
```

At the same time, make the following changes for your DirectoryConfigurationAdam objects.

- CEMP → DirectoryConfigurationAdam → DirectoryServerHost: If the host name in your server certificate is the fully qualified host name, change this property to the fully qualified host name as well.
- CEMP → DirectoryConfigurationAdam → DirectoryServerPort: Change it to the SSL port number where your directory server is listening.
- CEMP → DirectoryConfigurationAdam → IsSSLEnabled: Set it to true.

Restart your application server. Now the SSL connections between your application server and your ADAM servers are established.

Operation

Get User or Group by Short Name

Iterate through all realms. For each realm:

1. Connect to corresponding host.
2. Search for the user/group by short name.
3. If found, return.

If more than one user/group is found, Content Engine will log an error and return the first user found.

Get User or Group by DN

1. Resolve the realm name from the DN.
2. Connect to corresponding host.
3. Search for the user/group by DN.

Get User or Group by UPN

ADAM provider does not support these methods.

Get User or Group by SID

Iterate through all realms. For each realm:

1. Connect to corresponding host.
2. Search for the user/group by SID.
3. If found, return.

Search Users or Groups in a Given Realm

1. Connect to the host corresponding to the specified realm.
2. Search for the users or groups by the search criteria.

Directory service performance

This topic provides suggestions for improving the performance of a FileNet P8-initiated search of the directory service for users and groups for Sun Java System Directory Server and Novell eDirectory.

Sun Java System Directory Server

This document recommends certain configurations of your Sun Java System Directory Server to better support FileNet P8, especially in the area of sorting.

Glossary definitions

Here are some Sun Java System Directory Server glossary definitions, for easy reference:

All IDs Threshold. A size limit which is globally applied to every index key managed by the server. When the size of an individual ID list reaches this limit, the server replaces that ID list with an All IDs token.

browsing index. Otherwise known as the virtual view index, speeds up the display of entries in the Directory Server Console. Browsing indexes can be created on any branchpoint in the directory tree to improve display performance.

default index. One of a set of default indexes created per database instance. Default indexes can be modified, although care should be taken before removing them, as certain plug-ins may depend on them.

dn (distinguished name). String representation of an entry's name and location in an LDAP directory.

equality index. Allows you to search efficiently for entries containing a specific attribute value.

filter. A constraint applied to a directory query that restricts the information returned.

nsslapd-sizelimit (Size Limit). Specifies the maximum number of entries to return from a search operation. If this limit is reached, ns-slapd returns any entries it has located that match the search request, as well as an exceeded size limit error.

nsLookthroughLimit. Specifies the maximum number of entries that the directory will check when examining candidate entries in response to a search request.

substring index. Allows for efficient searching against substrings within entries. Substring indexes are limited to a minimum of two characters for each entry.

virtual list view index. Otherwise known as a browsing index, speeds up the display of entries in the Directory Server Console. Virtual list view indexes can be created on any branchpoint in the directory tree to improve display performance.

Additional information

For information about the Sun Java System Directory Server features mentioned here, see:

Sun Java System Directory Server 5.2 Installation and Tuning Guide, "Chapter 7 Tuning Indexing."

Sun Java System Directory Server 5.1 or 5.2 Administration Guide, "Chapter 10 Managing Indexes."

Recommendations

To ensure proper sorting of a returned sublist in a Sun Java System Directory Server environment containing a large number of entries, you should make one or more of the following configuration changes:

- Create substring indexes for the uid and cn fields that typically map to Content Engine's UserShortNameAttribute, GroupShortNameAttribute, UserDisplayNameAttribute, and GroupDisplayNameAttribute properties. (To view these properties in Enterprise Manager, right-click the Root Domain node, select Properties, and then click the Directory Config tab.)

Note If you implement this recommendation, users should avoid entering single-character search strings. See the topics under "More information about the recommendations" below for information about supporting single-character searches while using substring indexes.

- Increase the All Ids Threshold from its default of 4000, as appropriate. For example, if its value is 4,000 and if there are more than 4,000 entries in the Directory Server that start with "sys", then the Directory Server will no longer maintain that index. This means that Directory Server will return an error if you do a sorted query for "StartsWith sys", even if you set the maximum results parameter to less than 4,000. According to Sun's documentation, you should try to maintain this threshold at no less than 5% of your total directory. That is, the default of 4,000 assumes an authentication directory containing approximately 80,000 entries. If your directory contained 150,000 entries, you would probably want to increase the All Ids Threshold to at least 7500.
- Increase the Look Through Limit from its default of 5000 to -1 (no limit) so that there is no maximum number of accounts that will be checked in any given search. An alternative would be to set this value to such a large number that virtually no search would reach the maximum.
- Note The Directory Server may return an error when the Directory Server entry count is over this limit, regardless of whether the query is sorted.
- Increase the Size Limit parameter (default of 2000) to a value larger than the maximum-results entry set for any associated FileNet P8 user interface (for example, the FileNet P8 Workplace Site Preference setting called Group and User Maximum Filter Size). This ensures that the expected maximum number of results will be returned. Otherwise, users could get back the number of records determined by the size limit parameter and not be aware that there are in fact more entries that meet the search criteria.
- Set a browsing index in your Sun Java System directory to improve the performance of a no-pattern query.

Note Queries without a search pattern are used to retrieve all entries.

More information about the recommendations

Simply increasing the All IDs Threshold by itself is not advisable, given the Sun Java System Directory Server recommendation to keep the All IDs Threshold value at about 5% of the Directory Server total entry count. Also, handling single-character searches without using a substring index could incur performance problems. Therefore it is recommended that you select an approach combining browsing index, substring index, and manipulating the All IDs Threshold. The following sections describe some of the issues to be resolved while implementing the recommendations.

Create a substring index for sorting attributes

For each attribute to be sorted, create a substring index in order to support the FileNet P8 pattern search. Remember that the Directory Server will still return an error if the entry count for a specific index is over the setting of All IDs Threshold.

You should enable a substring index for each attribute you assigned as Content Engine's UserShortNameAttribute, GroupShortNameAttribute, UserDisplayNameAttribute, and GroupDisplayNameAttribute. They are shown on the Directory Config tab of the Root Domain node in Enterprise Manager. For Content Engine, these attributes are normally uid and cn. For Process Engine, the sort attribute is cn (display attribute in peboot.ini.)

The substring index does not support the case of querying all users (which is done by leaving the FileNet P8 search string empty). This problem can be resolved by creating a browsing index, because the filter is static so only a few browsing entries are needed.

Notes

- iPlanet Directory Server 5.1: Substring indexing does not support a single-character pattern. Users should enter at least two characters. Otherwise, searches may return inconsistent results for single-character vs. multi-characters pattern searches, and error conditions may show up in FileNet P8 user interfaces. This problem is resolved by upgrading to the Sun Java System Directory Server 5.2 release, which supports single-character search strings.
- Sun Java System Directory Server 5.2: Substring indexing does support a single-character search pattern as long as you use the FileNet P8 "Starts with" search type.

Change the All IDs Threshold value

All queries will run as expected provided the All IDs Threshold is greater than the Directory Server entry count. The All IDs Threshold is a factor only when sorting is requested.

With the substring index enabled, a multi-character pattern search works as long as the Query entry count is less than All IDs Threshold.

Change the All IDs Threshold to a number greater than the Directory Server entry count to ensure that single-character pattern searches work. There are, however, some costs of doing so. For example, when an index for a specific attribute value is over the All IDs Threshold value, the Directory Server will not maintain the index list for that value. In order to resolve this problem, the All IDs Threshold should be increased. The Sun recommendation is to keep the All IDs Threshold value at about 5% of the Directory Server total entry count, but even this percentage may have to be adjusted.

Example Assume the Directory Server has 80,000 entries and the threshold is 4000. Also assume there are 5000 entries that start with "pw" and 1000 entries that start with "au". When you query for "au*" you get 1000 entries, no error. But when you query "pw*" you would get an error because there are more than 4000 entries that start with "pw" (in fact there are 5000 entries) and Sun Java System Directory Server stops sorting for this case. You would have to increase the threshold to a number over 5000 in order to get back all entries that fulfill the query's specifications.

Effects of changing the All IDs Threshold without enabling a substring index

FileNet conducted tests to show the effects of the All IDs Threshold without enabling a substring index.

Conclusions

- Directory Server returns an error when the Directory Server entry count is over the setting of All IDs Threshold.
- All queries works as long as the All IDs Threshold is greater than the Directory Server entry count.
- You may have performance problems if you create no index.
- The Sun documentation recommends setting the All IDs Threshold value as 5% of the Directory Server entry count, but this may or may not be adequate, depending on your particular Directory Server.

Recommendation

- Change sizeLimit to -1 (no limit).
- Change nsslapd-lookthroughlimit to -1 (no limit).
- Set the All IDs Threshold to a value no less than 5% of the Directory Server entry count, but monitor and test user and group account searches carefully to determine whether a higher value may be more appropriate.

How to create a browsing index

The substring index does not help queries without pattern. A query without pattern returns an error when the Directory Server entry count is over the All IDs Threshold value. To resolve this problem, you should create a browsing index for a static condition.

Here are steps to create a browsing index, (assuming the Short name and Display name of user entries is uid, the Short name and Display name of group entries is cn, the base DN is dc=eng,dc=filenet,dc=com):

- Create a file BrowsingIndex.txt with the following contents:

```
dn: cn="dc=eng,dc=filenet,dc=com:(objectClass=person)",  
cn=userRoot,cn=ldbm database,cn=plugins,cn=config  
objectClass: top  
objectClass: vlvSearch  
cn: "dc=eng,dc=filenet,dc=com:(objectClass=person)"  
vlvbase: dc=eng,dc=filenet,dc=com  
vlvscope: 2  
vlvfilter: (objectClass=person)
```

```
dn: cn=sort uid,cn="dc=eng,dc=FileNet,dc=com:(objectClass=person)",cn=userRoot,cn=ldbm  
database,cn=plugins,cn=config
```

```
objectClass: top  
objectClass: vlvIndex  
cn: sort uid  
vlvSort: uid
```

```
dn: cn=rev sort uid,cn="dc=eng,dc=filenet,dc=com:(objectClass=person)",cn=userRoot,cn=ldbm  
database,cn=plugins,cn=config
```

```
objectClass: top  
objectClass: vlvIndex  
cn: rev sort uid  
vlvSort: -uid
```

```
dn: cn="dc=eng,dc=filenet,dc=com:(objectClass=groupOfUniqueNames)",cn=userRoot,cn=ldbm  
database,cn=plugins,cn=config
```

```
objectClass: top  
objectClass: vlvSearch  
cn: "dc=eng,dc=filenet,dc=com:(objectClass=groupOfUniqueNames)"  
vlvbase: dc=eng,dc=filenet,dc=com  
vlvscope: 2  
vlvfilter: (objectClass=groupOfUniqueNames)
```

```
dn: cn=sort cn,cn="dc=eng,dc=filenet,dc=com:(objectClass=groupOfUniqueNames)",cn=userRoot,cn=ldbm  
database,cn=plugins,cn=config  
objectClass: top  
objectClass: vlvIndex  
cn: sort cn  
vlvSort: cn
```

```
dn: cn=rev sort cn,cn="dc=eng,dc=filenet,dc=com:(objectClass=groupOfUniqueNames)",cn=userRoot,cn=ldbm  
database,cn=plugins,cn=config  
objectClass: top  
objectClass: vlvIndex  
cn: rev sort cn
```

vlvSort: -cn

```
dn: cn=sort_users_cn, cn="dc=eng,dc=FileNet,dc=com:(objectClass=person)",
cn=userRoot,cn=ldbm database, cn=plugins, cn=config
objectClass: top
objectClass: vlvIndex
cn: sort_users_cn
vlvSort: cn
```

```
dn: cn=rev_sort_users_cn ,
cn="dc=eng,dc=filenet,dc=com:(objectClass=person)",cn=userRoot,cn=ldbm
database,cn=plugins,cn=config
objectClass: top
objectClass: vlvIndex
cn: rev_sort_users_cn
vlvSort: -cn
```

The first three entries are browsing-index entries for querying users. The next three entries are browsing- index entries for querying groups. The last two are sort indexes on objectClass=person, based on cn instead of uid. Each vlvSearch entry is tied to a specific base DN and a specific filter. Note that both vlvSearch and vlvIndex entries should be named. In this example, the vlvSearch name is formed by concatenating the base DN and the filter as a unique name. For the user case, one vlvIndex is named sort uid and the second one is named rev sort uid. It is assumed that the user Short name attribute and Display name attribute are the same attribute: uid. If they are not the same, then two more vlvIndex entries should be added for another attribute. In this example, these vlvSearch and vlvIndex entries are specific to the base DN dc=eng,dc=filenet,dc=com. If there are more realms on the server, a new set of entries should be added for each realm.

- Run the ldapmodify utility to create vlvSearch and vlvIndex entries using the above file as input:

```
<SUNONE_INSTALL_DIR>\shared\bin\ldapmodify -a -h <SUNONE_HOST_NAME> -p
<SUNONE_PORT_NUMBER> -D <USER_ID> -w <PASSWORD> -f <full path of
BrowsingIndex.txt>
```

- Create browsing indexes:

```
<SUNONE_INSTALL_DIR>\bin\slapd\server\slapd db2index -D
"<SUNONE_INSTALL_DIR>\slapd-<SUNONE_SERVER_NAME>" -n <DB_NAME> -T "sort uid"
```

```
<SUNONE_INSTALL_DIR>\bin\slapd\server\slapd db2index -D
"<SUNONE_INSTALL_DIR>\slapd-<SUNONE_SERVER_NAME>" -n <DB_NAME> -T "rev sort
uid"
```

```
<SUNONE_INSTALL_DIR>\bin\slapd\server\slapd db2index -D
"<SUNONE_INSTALL_DIR>\slapd-<SUNONE_SERVER_NAME>" -n <DB_NAME> -T "sort cn"
```

```
<SUNONE_INSTALL_DIR>\bin\slapd\server\slapd db2index -D
"<SUNONE_INSTALL_DIR>\slapd-<SUNONE_SERVER_NAME>" -n <DB_NAME> -T "rev sort
cn"
```

Repeat the process above if there are more indexes added to the BrowsingIndex.txt file.

- Run the Sun Java System Directory Server Console, and open a server window.
- Click the Directory tab.
- Expand "config" in the left pane, then expand "plugins" and "ldbm database."

- Click userRoot and find the vlvSearch entries (for both user and group) created by the earlier step.
- Right-click the vlvSearch entry and select "Set Access Permissions."
- Add a new ACI as anonymous user. This will allow all users to access these vlvSearch entries.

Configuring sorting for Novell eDirectory

FileNet recommends that customers configure directory sorting options to ensure that users and groups are returned in a meaningful order.

Create an index for each user/group short name and display name attribute (follow the Novell documentation). For example, add an index for cn and choose rule: substring.

Use ConsoleOne

- In ConsoleOne, right-click the Server object.
- Click Properties > Indexes > Add.
- Enter the Index Name.

If you do not enter an index name, the attribute is automatically assigned as the index.

- Select an attribute.
- Select the index rule.

Value matches the entire value or the first part of the value of an attribute. For example, value matching could be used to find entries with a LastName that is equal to "Jensen" and entries with a LastName that begins with "Jen".

Presence requires only the presence of an attribute rather than specific attribute values. A query to find all entries with a Login Script attribute would use a presence index.

Substring matches a subset of the attribute value string. For example, a query to find a LastName with "der" would return matches for Derington, Anderson, and Lauder.

A substring index is the most resource intensive index to create and maintain.

- Click OK to update the index table.
- Click Apply to restart Limber as a background process and initiate the change.

Users and groups required by FileNet P8 Platform

This topic describes the users, groups, security access roles, and security-related responsibilities required to install, configure, and administer FileNet P8 and its family of applications.

Required for installation - summary

Role/responsibility/tasks	Description
Content Engine system user	Application server administrative account that is used to create the GCD and launch Content Engine.
Application server administrator: WebLogic, WebSphere	Administrator who configures the application server that will contain Content Engine. NOTE There is no requirement for a JBoss administrative account.
Install Content Engine	(Content Engine) Administrator who installs Content Engine on Windows/Unix.
Install Process Engine	(Process Engine) Administrator who installs Process Engine.
Accounts required by Process Engine Setup (Windows)	(Process Engine) Several users and groups required by Process Engine/Windows Setup.
Accounts required by Process Engine Setup (Unix)	(Process Engine) Several local users and groups required by Process Engine/UNIX Setup.
Install Application Engine	(Application Engine) Administrator who installs the Application Engine.
Publishing user account	(Rendition Engine) A Windows domain user account that must be added to the local Administrators group on all Rendition Engine servers.

Required for database creation and connectivity - summary

Role/responsibility/tasks	Description
Content Engine SQL Server account	Creates and maintains connectivity for Content Engine using SQL Server.
Content Engine Oracle alias	Owns and maintains connectivity for Content Engine using Oracle.
Content Engine DB2 account	Creates and maintains connectivity for Content Engine using DB2®.

Process Engine database administrators group: Windows and Unix	Database Administrators Group (default = dba) required by Process Engine Setup for Oracle databases.
Process Engine ORA_DBA group	(Process Engine, Windows authentication only) ORA_DBA group is required to install Process Engine on Windows, using Oracle.
Process Engine Oracle account	Oracle User (default = oracle) is required by Process Engine Setup for Oracle.
Rendition Engine SQL Server account	(Rendition Engine, SQL Server only) Required by Rendition Engine setup.

Required for administration - summary

Role/responsibility	Description
GCD administrators	(Content Engine) Administrators with Full Control on the FileNet P8 domain object.
Object store administrators	(Content Engine) Administrators with Full Control to an object store. Has "Set Any Owner" privileges to any object.
Content Engine directory service user	(Content Engine) Used by Content Engine to connect to Sun, Novell, IBM, Active Directory, or ADAM.
Content Engine operating system account	(Content Engine) Required to configure the shared root directory of a file storage area.
Autonomy K2 server accounts: <ul style="list-style-type: none"> • K2 security group • K2 security user • K2 operating system user 	(Content Engine/Autonomy K2) User and group accounts used by Content Engine to connect to the Autonomy K2 server for CBR, and the operating system account that K2 services run as.
Process Engine service user	(Process Engine) Account used by Process Engine when connecting to Content Engine.
Process Engine administrators group	(Process Engine) Group whose members have administrative privileges for Process Engine.
Process Engine configuration group	(Process Engine) Group whose members have configuration privileges for workflows on Process Engine.

Application Engine administrator role	Role whose members have administrative privileges for the Application Engine.
CFS for IS user	(Content Federation Services) Account that Content Engine uses to log into the Image Services system.

Required for Workplace and Workplace XT applications - summary

Access Roles	Description
PSConsole, PSDesigner, PWAdministrator, PWConfiguration, PWDDesigner	Optional access roles. Application Engine administrator role members can define new access roles to limit access to application features.

Required for internal operations - summary

Role/responsibility	Description
#AUTHENTICATED-USERS	(Content Engine) A logical group principal whose members are all authenticated users.
CREATOR-OWNER	(Content Engine) The special account granted to the user who creates an object.
Process Engine internally required accounts created by Setup: <ul style="list-style-type: none"> • f_maint, f_sw • SysAdmin, FieldService, Operator 	Accounts used internally by Process Engine for specific administration or troubleshooting activities. The accounts are automatically created by PE Setup. FileNet recommends resetting the initial passwords to maintain system security.

Required for installation - details

Content Engine system user

Description	An application server administrative account that is stored in the CEMPBoot.properties file and is used first to create the GCD and thereafter is the account that Content Engine uses to run as.
Additional info	This account is used to login to the application server, access the datasources named in the GCDConnection property, and create the GCD. Content Engine will not be able to start if this user is not able to log on to the application server. This account should be used only for this purpose and should not be considered an all-purpose admin account.

	<p>Any deployments of the EAR file for the same FileNet P8 domain must have the same property values. FileNet also recommends that you exempt this account from policies requiring periodic password change.</p> <p>If you change your system's login parameters so that the Content Engine system user's credentials are no longer valid, the result would be that Content Engine will not be able to start. For example, if you modified the User Short Name Attribute or User Search Filter, in the application server's authentication provider and in Enterprise Manager's P8 Domain Properties > Modify Directory Configuration > User property sheet, from "samAccountName" to "distinguishedName", you would also need to make the same change to the com.filenet.gcd.Username property in the CEMPBoot.properties file. For information on how to change the ID or password for the Content Engine system user, see CE Bootstrap properties.</p> <p>Content Engine system user is the creator of the new P8 domain:</p> <ul style="list-style-type: none"> • Content Engine Setup uses the credentials of the Content Engine system user to create the new P8 Domain, and also makes it a GCD administrator by granting it Full Control over the new FileNet P8 domain object. • The first time you launch Enterprise Manager following initial installation, you logon as the Content Engine System user. During this initial launch, you will run the Configure New Domain Permission wizard. This wizard prompts you to add at least one user or group account to the list of GCD administrators (that is, accounts with Full Control access to the GCD). The Content Engine system user will remain on the list of GCD administrators unless some other GCD administrator explicitly removes it. However, FileNet recommends that you do remove it, and that you keep the Content Engine system user (in its role as application server administrator) for the sole purpose as the account that Content Engine will run as.
Minimum required permissions	This is an LDAP account that is also an application server administrator. The important permission it needs on the application server is access to the GCD datasources. These permissions are granted by Content Engine Setup, which grants it the Administrator role on the application server.
When is it created and when is it needed?	<p>These credentials are captured and configured by Content Engine Setup, in its Setup Content Engine Bootstrap Properties panel.</p> <p>The account must remain in the CEMPBoot.properties file as it is required to launch Content Engine.</p>
See also	Refer to CE Bootstrap properties. See also the entry for GCD administrator.

WebLogic administrator

Description	<p>Administrator who configures WebLogic before running Content Engine Setup as well as deploys the Content Engine EAR file during installation and subsequently to additional Content Engine servers. This administrator also logs on the WebLogic in order to create:</p> <ul style="list-style-type: none"> Authentication provider objects needed for multi-realm authentication or to add additional login parameters to existing realms. Datasources required for each object store. <p>This administrator will be the same as the CE system user, if you enter the</p>
-------------	---

	same credentials into the Specify WebLogic Information panel and also the Setup Content Engine Bootstrap Properties panel while running Content Engine Setup.
Minimum required permissions	Full administrative control over the WebLogic domain that will contain Content Engine.
When is it created and when is it needed?	Enter the credentials of this account while installing Content Engine. The account is required on an ongoing basis.
See also	"Configure WebLogic for Content Engine" task in the Installation Guide. For information about creating datasources, see the "Create an object store" task in the Installation Guide.

WebSphere administrator

Description	<p>Administrator who configures WebSphere before running Content Engine Setup as well as deploys the CE EAR file during installation and subsequently to additional Content Engine servers. This administrator also logs on to WebSphere in order to create datasources required for each object store.</p> <p>The WebSphere administrator is an alias to an actual account in the configured directory server. If WebSphere global security is turned on, the account must reside in the configured directory server.</p> <p>This administrator will be the same as the CE system user, if you enter the same credentials into the Specify WebSphere Administrator Account panel and also the Setup Content Engine Bootstrap Properties panel while running Content Engine Setup.</p>
Minimum required permissions	Full administrative control over the WebSphere domain that will contain Content Engine.
When is it created and when is it needed?	Enter the credentials of this account while installing Content Engine. The account is required on an ongoing basis.
See also	"Configure WebSphere for Content Engine" task in the Installation Guide. For information about creating datasources, see the "Create an object store" task in the Installation Guide.

Install Content Engine

Description	Permissions needed to run Content Engine installation program.
Minimum required	On a Windows server: logon as a Local administrator.

permissions	On AIX®: logon as root. On Unix servers except AIX: you must have read/write/execute permissions on the FileNet, application server instance, and temp directories.
When is it needed?	Required while running Content Engine setup program.
See also	"Install and Deploy Content Engine" in the Installation Guide, including information about installing on Windows as a Power User.

Publishing user account

Description	(Content Engine, if installing the Rendition Engine) A Windows domain user account that must be added to the local Windows Administrators group on all Rendition Engine servers.
Minimum required permissions	Domain privileges in the Windows domain that contains the FileNet P8 domain and local administrator privileges on the machine where the Rendition Engine is installed.
Additional info	Installation instructions refer to this user as FNRE_Admin, but you could use any Windows domain account. On all servers where you intend to install Publishing components (Publishing Plug-in servers and Rendition Engine servers that will contain the publishing software), log on as a local administrator and add the domain_name\FNRE_Admin account to the local Administrators group. You will then log on as this user to run the Rendition Engine setup program.
When is it created?	"Install and configure FileNet Publishing components" in the Rendition Engine Installation and Upgrade Guide.
When is it needed?	Required on an ongoing basis if your Content Engine installation includes a Rendition Engine.

Logon requirements for running Process Engine Setup (all platforms)

Description	The user who runs Process Engine Setup program.
Minimum required permissions	Windows: A Windows administrator for the local machine. This user must also have administrator privileges on the database if you plan to run the database scripts using OS authentication. You can log on as the local Windows Administrator or create a user (with the specified permissions) expressly for the purpose of running Process Engine Setup. UNIX: The root user, running in either the Bourne or Korn shell.
When is it needed?	(Windows only) If you created a user specifically to run Process Engine Setup, you can delete that user after Process Engine Setup completes.

See also	"Install Process Engine" in the Installation Guide.
----------	---

Accounts required by Process Engine Setup (Windows)

Description	<p>Several local operating system users and groups, or their assigned aliases, required by Process Engine Setup. Requirements vary depending on the database:</p> <p>Groups (Oracle): fnadmin, fnusr, fnop, <database administrators group; default group name = dba>, ORA_DBA. Users (Oracle): fnsw, <Oracle user; default user name = oracle></p> <p>Groups (MS SQL Server): fnadmin, fnusr, fnop User (MS SQL Server): fnsw</p> <p>Groups (DB2): fnadmin, fnusr, fnop User (DB2): fnsw</p>
Minimum required permissions	<p>The fnsw user is added to the following groups during installation (unless you predefine these users and groups, in which fnsw must be manually added to these groups before running PE setup): fnadmin, fnusr, fnop, ORA_DBA and <dba> groups. If the database is Oracle, the fnsw user can be removed from the ORA_DBA group when the installation is complete.</p> <p>Windows: The fnsw user is added to the local Windows Administrator group, and is given local security user rights of:</p> <ul style="list-style-type: none"> • Log on as a service. • Act as part of the operating system. • Increase quotas. • Replace a process level token. <p>If the database is Oracle, the <Oracle> user is a member of the ORA_DBA group. The <Oracle> user can be removed from the ORA_DBA group when the installation is complete. If you have a remote database and are not using the Oracle user for Oracle database activities, then the Oracle user can also be removed.</p>
Additional info	<p>fnadmin: Members have all privileges on Process Engine files and databases. fnusr: Members have non-administrator privileges on Process Engine files and databases. fnop: Members have non-administrator operator privileges on PE Oracle database administrators group: Members act as database administrators. fnsw: Primary Process Engine user. Used to execute Process Engine software and Services. FileNet strongly recommends resetting the initial password to maintain system security. See "Install Process Engine (Windows)" in the Installation Guide for further considerations in resetting the fnsw password.</p>
When is it created?	These accounts are defined automatically during Process Engine Setup (unless predefined by the user); automatic definition includes setting the minimum required permissions described above.
When is it	Required on an ongoing basis by Process Engine.

needed?	
See also	"Install Process Engine (Windows)" in the Installation Guide for descriptions of the users and groups, and for further information about resetting the fnsw password.

Accounts required by Process Engine Setup (UNIX)

Description	Several operating system users and groups, or their assigned aliases, required by Process Engine Setup. Groups: fnadmin, fnusr, fnop, <database administrators group; default group name = dba> Users (Oracle): fnsw, <Oracle user; default user name = oracle> Users (DB2): fnsw
Minimum required permissions	For the fnsw user, the primary group must be fnusr, and fnop, fnadmin and <dba> must be secondary groups. The Korn shell must be the default shell for the fnsw user. If the database is Oracle, the fnsw user can be removed from the <database administrators group> when the installation is complete. For the root user, fnadmin and fnusr must be secondary groups. The Korn shell must be the default shell for the root user. For the <Oracle> user, the primary group must be <dba> and fnusr must be a secondary group.
Additional info	fnadmin: Members have all privileges on Process Engine files and databases. fnusr: Members have non-administrator privileges on Process Engine files and databases. fnop: Members have operator non-administrator privileges on Image Services utilities. Oracle database administrators group: Members act as database administrators. fnsw: Primary Process Engine user. Used to execute Process Engine software and Services.
When is it created?	Manually created prior to running Process Engine Setup.
When is it needed?	Required on an ongoing basis by Process Engine.
See also	"Install Process Engine" in the Installation Guide for descriptions of the users and groups, and for instructions to create them manually.

Install Application Engine

Description	The administrator who installs the Application Engine.
Minimum required permissions	Unix: Administrator must have: <ul style="list-style-type: none"> • write access to the /bin directory

	<ul style="list-style-type: none"> read, write, and execute access to the directory where you plan to install Application Engine. <p>Windows: Administrator must belong to the local Administrators group or a user with equivalent permissions.</p>
Additional info	The FileNet P8 WebDAV provider is installed along with the Application Engine and is administered by the administrator who installs the Application Engine.
When is it needed?	To run the Application Engine setup program.
See also	"Install the Application Engine" in the Installation Guide.

Required for database creation and connectivity - details

Content Engine SQL Server account

Description	Creates and maintains connectivity between Content Engine and Microsoft SQL Server databases containing the GCD and object stores.
Minimum required permissions	<p>SQL Server Roles: System Administrators, Server Administrators, and Database Creators.</p> <p>Database Access: public and db_owner for GCD database and all object store databases.</p> <p>Also add this user to SQL Server's master database and grant it the SqlJDBCXAUser role, along with the public role, so it can participate in distributed transactions with the JDBC driver.</p>
Additional info	You can use the same account for the GCD and object store databases. This account may optionally reside in the configured directory service.
When created	Must be created and configured in SQL Server before running Content Engine setup (for the GCD) and before running Enterprise Manager's object store wizard.
See also	Tasks in the Installation and Upgrade Guide: Specify FileNet P8 Accounts, Install and Deploy Content Engine, Configure Content Engine Application Server Database Connectivity.

Create Oracle tablespaces for Content Engine

Description	Creates and maintains connectivity between Content Engine and Oracle databases containing the GCD and object stores.
Minimum required permissions	CREATE SESSION, CREATE TABLE, and CREATE SEQUENCE roles
Additional	The Oracle roles CONNECT and RESOURCE combine to include the minimal

info	<p>privileges required by Content Engine, which are CREATE SESSION, CREATE TABLE, and CREATE SEQUENCE. Because these two roles include other privileges as well, FileNet recommends that you design your own roles if you prefer to grant only the minimal privileges required by Content Engine.</p> <p>You must create a different Oracle user for each Content Engine database. That is, multiple object stores must not share the same Oracle user, because otherwise the objects you intend to add to one object store will show up in all those object stores that share the same Oracle user.</p> <p>This account may optionally reside in the configured directory service.</p>
When created	Must be created and configured in Oracle before running Content Engine setup (for the GCD) and before running Enterprise Manager's object store wizard.
See also	Tasks in the Installation and Upgrade Guide: Specify FileNet P8 Accounts, Install and Deploy Content Engine, Configure Content Engine Application Server Database Connectivity.

Content Engine DB2 account

Description	Creates and maintains connectivity between Content Engine and DB2 databases containing the GCD and object stores.
Minimum required permissions	<p>CREATETAB and CONNECT ON DATABASE.</p> <p>USE OF TABLESPACE on User and User Temp DB2 tablespaces used for GCD and object stores.</p>
Additional info	<p>Content Engine DB2 account is created on the AIX operating system first. It can be the instance owner, the fenced user, or a completely separate user with the above permissions granted to it. FileNet recommends the latter configuration.</p> <p>This account may optionally reside in the configured directory service.</p>
When created	Must be created and configured in DB2 before running Content Engine setup (for the GCD) and before running Enterprise Manager's object store wizard.
See also	Tasks in the Installation and Upgrade Guide: Specify FileNet P8 Accounts, Install and Deploy Content Engine, Configure Content Engine Application Server Database Connectivity.

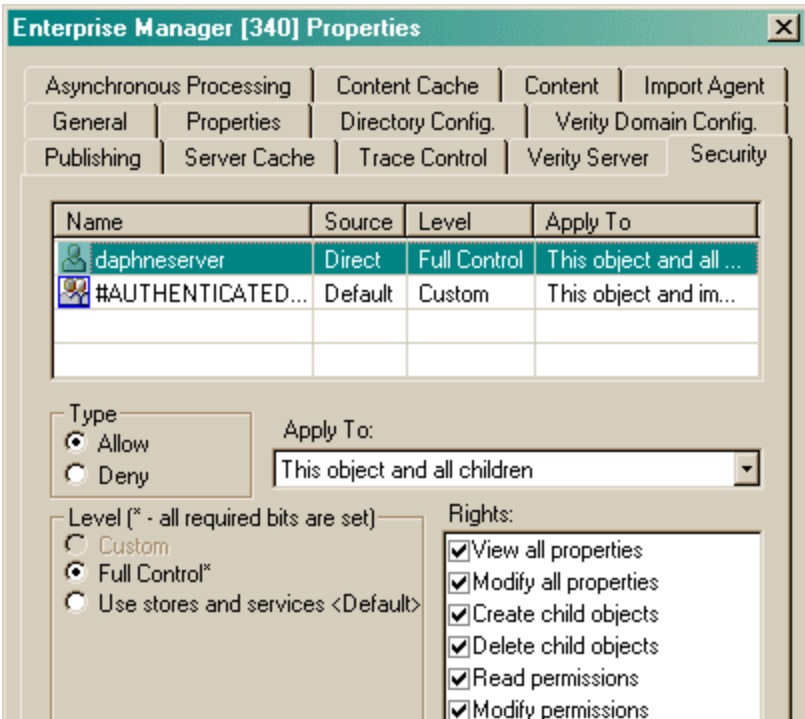
Rendition Engine SQL Server account

Description	Creates and maintains connectivity for Rendition Engine installations that use SQL Server.
Minimum required permissions	<p>Any user with SQL Server's "Create Database" permission.</p> <p>Any SQL Server user with SQL Server's "Database Creators" server role. (In SQL Server Enterprise Manager, navigate to Security > Logins. Right-click the user's name and select Properties. Click the Server Roles tab and select at least Database Creators.)</p>
Additional	This user is a safer alternative to using SQL Server's SA or SYS users, which

info	will also satisfy the configuration requirements. (This user is not the same as the FNRE_Admin user.)
When created	As part of installing FileNet Publishing components (Vista configuration).
See also	"Install and configure FileNet Publishing components" in the Rendition Engine Installation and Upgrade Guide.

Required for administration - details

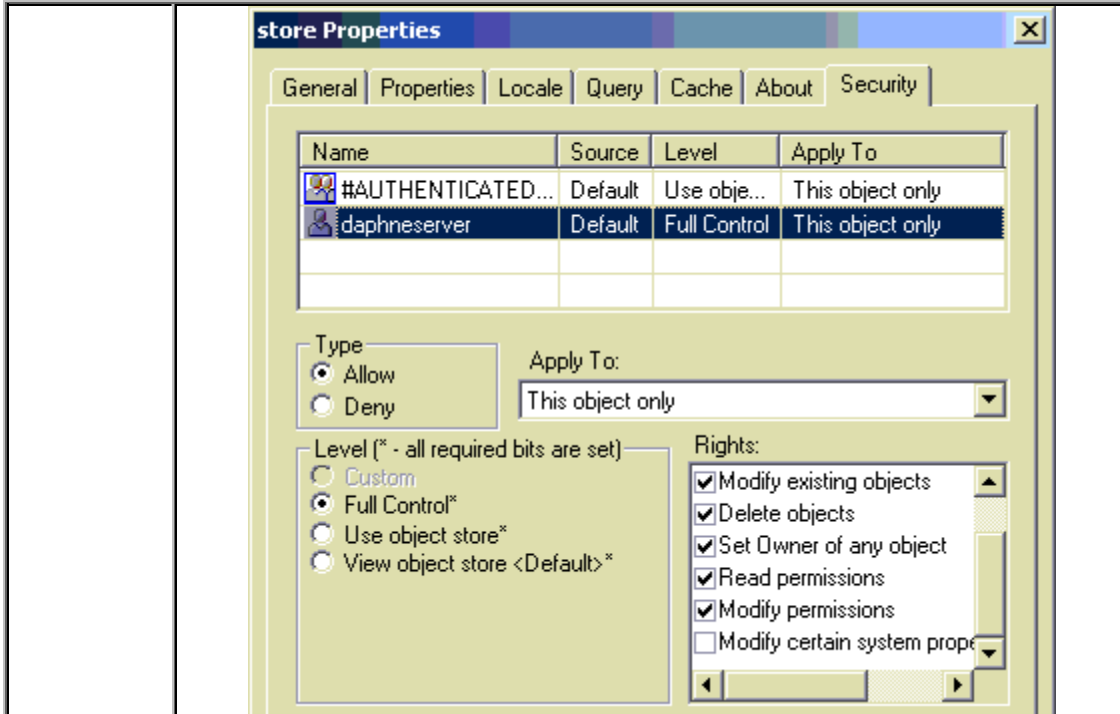
GCD administrator role

Description	(Content Engine) Users and groups who have been granted Full Control over the FileNet P8 domain object.
Additional info	<p>Full Control over the FileNet P8 domain object is a security level comprised of the following access rights:</p> <ul style="list-style-type: none"> • View and modify all properties • Create and delete child objects • Read and modify permissions <p>In the following screen shot the selected user has Full Control access to Enterprise Manager's Domain Root Properties > Security property sheet and is therefore a GCD administrator. A GCD administrator can grant Full Control rights to additional users and groups, thereby making them GCD administrators as well.</p>  <p>Being a GCD administrator does automatically make you an object store</p>

	administrator which is a role assigned on the object store's own property sheet.
When created	<p>Content Engine's installation program, in its Setup Content Engine Bootstrap Properties step, requests the credentials of the initial GCD administrator. This account must reside in the directory service specified during Content Engine installation.</p> <p>When Enterprise Manager is launched for the first time after Content Engine installation, running as the GCD Administrator, you will:</p> <ol style="list-style-type: none"> 1. Complete the Directory Configuration Wizard which automatically launches the first time you launch Enterprise Manager. This Wizard connects Content Engine to the directory server for authorization. See Directory service provider integration for more information. 2. Complete the Domain Permissions Wizard which assigns additional users and groups to the role of GCD administrator. <p>See the entry for the CE System user for a different account of this process.</p> <p>You can add or remove users or groups from this list at any time later on. See Add or remove a GCD administrator.</p>
When is it needed?	Whenever you create or change a P8 domain resource.
See also	<p>For descriptions of FileNet P8 domain resources, see Concepts: FileNet P8 Domain in Help for Content Engine Administration.</p> <p>For descriptions of the wizards mentioned above, see Administer directory configurations and Configure New Domain Permissions Wizard.</p>

Object store administrator role

Description	(Content Engine) Users and groups who have been granted Full Control access to an object store.
Additional info	<p>Full Control over an object store is a security level comprised of the following access rights:</p> <ul style="list-style-type: none"> • Connect to store, and create new and modify existing objects (not including P8 domain objects) • Delete objects • Set owner of any object • Read and modify permissions. <p>In the following screen shot any accounts with Full Control permission to the object store (named "store" in this example) are its object store administrators.</p>



NOTE See Object store access rights for information about why Modify certain system properties is not included in Full Control.

Each time a GCD administrator runs the Object Store Wizard, you are asked to specify the users and groups who should have administrative access to the object store. Each object store could therefore have a different set of administrators. Conversely, if you want the same groups to administer all object stores in the FileNet P8 domain, you must add them while creating each new object store using the Object Store Wizard. By default, the GCD administrator creating the object store also becomes an object store administrator.

Object store administrative rights do not include the ability to add, move, or remove object stores, fixed content devices, content cache areas, or any of the other FileNet P8 domain resources. These permissions are granted only to GCD administrators.

An object store administrator is not also a GCD administrator unless also specifically granted those permissions. This means that an object store administrator who is not also a GCD administrator would have to request that a GCD administrator create a new domain resource like an object store. Once these objects are created by the GCD administrator, however, the object store administrator can populate the object store with new classes and folders, store content in the file storage area, assign markings, and so on.

When is it created?

Specified each time a GCD administrator runs the Object Store Wizard to create a new object store. Any user or group with Full Control on this security page is an administrator of that object store. The list of object store administrators is available for viewing and modifying in Enterprise Manager's Object Store Properties > Security property page. You can add or remove users or groups from this list at any time later on. See Add or remove an object store administrator.

NOTE FileNet recommends that you keep the number of accounts assigned as object store administrators or object store end users as small as possible. This will improve performance and simplify administration. The best way to do this is to use group accounts instead of large numbers of individual users. Groups can

	have as many members as you wish and can be nested (that is, contain other groups).
When is it needed?	Required for ongoing administration of the object store.
See also	Logging on and using Enterprise Manager in Help for Security.

Directory service user (Content Engine)

Description	Used by Content Engine to connect to the directory server.
Minimum required permissions	<p>The directory service user must be configured with the following minimum rights for each security realm that will be configured for your FileNet P8 domain:</p> <p>Sun Java System Directory Service: Read, Search, Compare Novell eDirectory: Compare, Read IBM Tivoli: Read, Search, Compare Active Directory: Must belong to the Pre-Windows 2000 Compatible Access Group in each desired domain in the Active Directory forest. ADAM: Ability to see the other users in the partition. To configure this, do the following:</p> <ol style="list-style-type: none"> 1. Start ADAM ADSI Edit under Start > All Programs > ADAM. 2. Connect to the partition. Expand partition in left-hand pane and click the CN=Roles node.)Be sure you have selected the CN=Roles container in the partition not under the CN=Configuration.) 3. In the right-hand pane right-click the CN=Readers group and select Properties. 4. In the Attributes list double-click the "member" attribute. 5. Click Add ADAM Account.... 6. Enter the full DN of the user to be designated as the service user while running Content Engine Setup, and click OK. 7. Click OK and click OK again.
Additional info	<p>When retrieving information from the directory service, Content Engine connects using this account. For example, in Enterprise Manager this occurs when you launch the Select Users and Groups dialog box to search for and add accounts to an object's ACL.</p> <p>The Directory Service User cannot be accessed using referrals.</p>
When is it created?	Any time before installing Content Engine. Available for viewing and modifying in Enterprise Manager: see Directory configuration properties (General tab) in Help for Content Engine Administration.
When is it needed?	Whenever Content Engine accesses the directory service.

Operating system account (Content Engine)

Description	<p>Account used to create and configure the shared root directory of a file storage area or content cache area.</p> <p>For Windows-based Content Engine and file storage areas, the operating system (OS) account must reside in the same Windows domain or in trusted Windows domains as the servers that host Content Engine and the file storage area.</p> <p>For Unix-based Content Engine and file storage areas, configuring security requires the use of NFS.</p>
Additional info	<p>The operating system user who logs on to the Content Engine server and starts the local application server process is the account that must be used to secure the folders and files in a file storage area. From a practical standpoint, the account that is used to install the application server should be the same account that is used to start the application server process. As an administrator, you will always log on using the same OS account to secure the folders and files in the file system that Content Engine will use for a file storage area.</p> <p>Optionally, you can use an OS group account. All OS user accounts would have to be members of this group.</p>
See also	<p>For a more complete description of the security requirements for creating file storage areas, see Storage area security. See also "Create a File Storage Area" in the Installation Guide.</p>

K2 security group account (Content Engine/Autonomy K2)

Description	<p>A directory service group account used by Content Engine to connect to the Autonomy K2 server for content-based retrieval (CBR).</p>
Additional info	<p>K2 security user accounts must be members of the K2 security group. Only members of this group will have access to the collection.</p> <p>The credentials assigned to the K2 security user and K2 security group are available for viewing and modifying in Enterprise Manager. See "FileNet P8 domain root properties (Verity Domain Config tab)" in Help for Content Engine Administration.</p> <p>Content Engine automatically places the value assigned to the K2 security group on each Verity Collection.</p>
When is it created?	<p>Assigned during initial installation.</p>
When is it needed?	<p>For the ongoing functioning of CBR.</p>
See also	<p>For more security information, see Security for Autonomy K2 server and the subtopic File storage area security in the Storage area security. See also "Install and Configure Content Search Engine" in the Installation Guide.</p>

K2 security user account (Content Engine/Autonomy K2)

Description	A directory service user account used by Content Engine to connect to the Autonomy K2 server for content-based retrieval (CBR).
Minimum required permissions	The K2 security user must be granted the following permissions: <ul style="list-style-type: none"> • Administrator rights on the server where Autonomy K2 is installed. • Assigned as a "Dashboard Administrator". This takes place during K2 installation and configuration.
Additional info	Content Engine logs on to Autonomy K2 server using the credentials of the K2 security user account, which must be a member of the K2 security group. Only members of this group will have access to the collection. The credentials assigned to K2 security user and K2 security group are available for viewing and modifying in Enterprise Manager. See "FileNet P8 domain root properties (Verity Domain Config tab)" in Help for Content Engine Administration.
When is it created?	Assigned during initial installation.
When is it needed?	For the ongoing functioning of CBR.
See also	For more security information, see Security for Autonomy K2 server and the subtopic File storage area security in the Storage area security. See also "Install and Configure Content Search Engine" in the Installation Guide.

K2 operating system user (Autonomy K2)

Description	The operating system account that Autonomy K2 services run as.
Minimum required permissions	The K2 operating system user must be an operating system administrator on the machine where Autonomy K2 Master Administration server is installed.
Additional info	The K2 operating system user is used to secure file system access to collections: <ul style="list-style-type: none"> • Windows-based file storage areas: the K2 operating system user must have Read access to the file storage area root directory. • Unix-based file storage areas: one way to grant access to the K2 operating system user is to set the Set UID bit of all the K2 program files, and then also set the owner of each program file to the K2 operating system user, which must also have Read/Execute access to the file storage area root directory.
When is it needed?	For the ongoing functioning of CBR.
See also	For more security information, see Security for Autonomy K2 server and the subtopic File storage area security in the Storage area security. See also "Install

	and Configure Content Search Engine" in the Installation Guide.
--	---

Service user (Process Engine)

Description	Account used by Process Engine when connecting to Content Engine.
Minimum required permissions	The Process Engine service user must be a member of the Process Engine Administrator Group.
When is it created?	Assigned using the Process Task Manager.
See also	Configure the Process Engine security connection

Process Engine administrator group

Description	Group whose members have administrative privileges for Process Engine.
When is it created?	Assigned using the Process Task Manager.
When is it needed?	By Process Engine administrators to administer the workflow database.
See also	About workflow security Configure the Process Engine security connection

Process Engine configuration group

Description	Group whose members have configuration privileges on the Process Engine workflow database.
Additional info	The Process Engine configuration group is optional. If specified, members of this group or Process Engine administrator group can make configuration changes to the workflow database. If not assigned, anyone can make these changes.
When is it created?	Assigned using the Process Task Manager.
When is it needed?	By Process Engine administrators to administer the workflow database.
See also	About workflow security Configure the Process Engine security connection

Application Engine administrator role

Description	Role whose members have administrative privileges for the Application Engine.
Minimum required permissions	None: this is an internally managed role.
Additional info	Managed by Access Roles preferences. The user who configures the bootstrap preferences on initial sign-in to the Workplace or Workplace XT application is automatically added to this role. Other users can be added to the role as needed.
When is it created?	Created while configuring Site Preferences.
When is it needed?	Required for access to Site Preferences, for modifying membership of specific roles.
See also	Access Roles preferences in Help for Site Preferences.

CFS for IS User

Description	Image Services account used by Content Engine to log on to and access Image Services resources as part of Content Federation Services configuration.
When is it created?	Any time prior to creating a CFS fixed content device.
When is it needed?	Used whenever Content Engine accesses the Image Services system as part of Content Federation Services configuration.
See also	See FileNet P8 Content Federation Services for Image Services Guidelines for full reference information.

Required for Workplace and Workplace XT applications - details

PSConsole

Description	Access role whose members can access Simulation Console from the Advanced Author page.
Minimum required permissions	None. Access roles are stored as values in custom objects in the object store.
Additional info	You can restrict the ability to run Simulation Console by assigning a group to the corresponding role in the Access Roles Site Preferences. If a group is assigned to the role, only members of the group can run Simulation Console.

When is it created?	While defining site preferences.
When is it needed?	By users running Simulation Console.
See also	Application security in Process Engine Reference.

PSDesigner

Description	Access role whose members can access Simulation Designer from the Advanced Author page.
Minimum required permissions	None. Access roles are stored as values in custom objects in the object store.
Additional info	You can restrict the ability to run Simulation Designer by assigning a group to the corresponding role in the Access Roles Site Preferences. If a group is assigned to the role, only members of the group can run Simulation Designer.
When is it created?	While defining site preferences.
When is it needed?	By users running Simulation Designer.
See also	Application security in Process Engine Reference.

PWAdministrator

Description	Access role whose members can access Process Administrator from the Admin page in Workplace or the Tools menu in Workplace XT.
Minimum required permissions	None. Access roles are stored as values in custom objects in the object store.
Additional info	You can restrict the ability to run Process Administrator by assigning a group to the corresponding role in the Access Roles Site Preferences. If a group is assigned to the role, only members of the group can run Process Administrator.
When is it created?	While defining site preferences.
When is it needed?	By users running Process Administrator.
See also	Application security in Process Engine Reference.

PWConfiguration

Description	Access role whose members can access Process Configuration Console from the Admin page in Workplace or the Tools menu in Workplace XT.
Minimum required permissions	None. Access roles are stored as values in custom objects in the object store.
Additional info	You can restrict the ability to run Process Configuration Console by assigning a group to the corresponding role in the Access Roles Site Preferences. If a group is assigned to the role, only members of the group can run Process Configuration Console.
When is it created?	While defining site preferences.
When is it needed?	By users running Process Configuration Console.
See also	Application security in Process Engine Reference.

PWDesigner

Description	Access role whose members can access Process Designer and the Workflow Subscription wizard from the Advanced Author page in Workplace or the Tools menu in Workplace XT..
Minimum required permissions	None. Access roles are stored as values in custom objects in the object store.
Additional info	You can restrict the ability to run Process Designer by assigning a group to the corresponding role in the Access Roles Site Preferences. If a group is assigned to the role, only members of the group can run Process Designer.
When is it created?	While defining site preferences.
When is it needed?	By users running Process Designer.
See also	Application security in Process Engine Reference.

Required for internal operations - details

#AUTHENTICATED-USERS

Description	(Content Engine) A logical group whose members are any authenticated user principal. Any user account that who can successfully login belongs to this
-------------	---

	group.
Additional info	<p>#AUTHENTICATED-USERS is similar to the special groups "Everyone" in Windows NT 4 and "Authenticated Users" in Windows 2000. It does not have specific memberships that you can modify, and it does not include anonymous users or guests.</p> <p>If you specify #AUTHENTICATED-USERS to be a default user/group of an object store, then all users who log onto the FileNet P8 domain are automatically made members of this group. It will appear on the Default Instance Security ACL of all classes. Therefore each instance of the class will include #AUTHENTICATED-USERS on its own ACL. If you do not change the default, the net effect will be that any user who can logon to the FileNet P8 domain will be able to:</p> <ul style="list-style-type: none"> • View all object stores (default level = "Use stores and services") • View all folders (default level = "View properties") • View all documents, both properties and content (default level = "View content") • View all custom objects (default level = "View properties") <p>If this is not what you want, you could:</p> <ul style="list-style-type: none"> • Remove #AUTHENTICATED-USERS from the particular class or classes. (You can, of course, remove it from individual objects, but this is not a recommended method for efficiently administering security across many classes and objects.) • Add "deny groups" to the class' Default Instance ACL; this will effectively remove the members of the deny group from the #AUTHENTICATED-USERS group. • Use a non-security method such as exploiting the "IsHiddenContainer" property which Workplace and other applications use to hide a folder. <p>#AUTHENTICATED-USERS and #CREATOR-OWNER are referred to as "Special Accounts" in Enterprise Manager's Select Users and Groups dialog box.</p>
When is it created?	Automatically created and maintained by Content Engine.

#CREATOR-OWNER

Description	(Content Engine) The special account granted to the user who creates an object.
Additional info	<p>#CREATOR-OWNER is a placeholder in an access control entry (ACE) and is used for copying a defined set of permissions to the individual user who is creating a new object. This copying takes place:</p> <ul style="list-style-type: none"> • When applying default instance security from a class to an instance of the class. • Whenever a security template places ACEs on an object.

	<ul style="list-style-type: none"> When performing inheritance propagation to a target ACE (such as from a parent folder to a child folder). <p>By default, #CREATOR-OWNER appears on the Security and Default Instance Security tabs of all instantiable classes, and is granted Full Control, with an inheritable depth of "This object only". This account functions just like a normal user account, and its default permissions can be edited according to normal rules (that is, by users with appropriate permission).</p> <p>When the ACE is inherited, the permissions granted to the #CREATOR-OWNER become the permissions granted to the object's current owner. For example, when a user creates a document based on a document class, that user takes on the #CREATOR-OWNER's permissions.</p> <p>Actually, two target ACEs result whenever the #CREATOR-OWNER is copied onto an object - a substituted ACE and a non-substituted ACE:</p> <ul style="list-style-type: none"> The substituted ACE is always created but is forced to be non-inheritable (its inheritable depth becomes "This object only" regardless of the source value). The unsubstituted ACE is a complete copy of the source ACE except that if performing inheritance propagation the inheritable depth value may be decremented (if it is not 0 or -1), and in all cases the unsubstituted ACE will be suppressed if the (resulting) inheritable depth is zero. <p>Windows Authentication: the user attribute used is the samAccountName. Sun Java System Directory Server and eDirectory: the user attribute is configurable to the LogonAttribute in the GCD.</p> <p>#AUTHENTICATED-USERS and #CREATOR-OWNER are referred to as "Special Accounts" in Enterprise Manager's Select Users and Groups dialog box.</p>
When is it created?	Automatically created and maintained by the Content Engine.
See also	Take or change ownership Mapping security levels to individual access rights for information about the Modify Owner permission.

Accounts required by Process Engine to access the workflow database

Description	Database accounts used by Process Engine to access the workflow database. You can instruct setup to automatically create these accounts or to assign an alias of your choice. By default, the following database users are created: <ul style="list-style-type: none"> f_maint: Used for database maintenance. f_sw: Required for runtime access to Process Engine database.
Minimum required permissions	Process Engine Setup grants various database permissions to these users. For a complete list, see "Process Engine SQL Scripts" in the Installation Guide.
When is it created?	Manually created before installation or automatically created during Process Engine installation.

When is it needed?	Required on an ongoing basis.
See also	Because Process Engine Setup creates each of these users with a default password, FileNet strongly recommends resetting the passwords to maintain system security. See "To set the f_maint and f_sw passwords (Oracle and SQL Server)" in Installation Guide for instructions.

Accounts required by Process Engine for internal use (created by Process Engine Setup)

Description	Process Engine Setup automatically creates several accounts (SysAdmin, FieldService, Operator) required when using the underlying Image Services (IS) tools. The users are created in the SEC database; they are not operating system, directory service, or database users.
Minimum required permissions	<ul style="list-style-type: none"> • SysAdmin: Logs on to the IS XApex Utility. Primary administrator user for FileNet IS tools. • FieldService: Used internally by Process Engine software. • Operator: Used internally by Process Engine software.
When is it created?	Automatically created during Process Engine installation.
When is it needed?	Required on an ongoing basis.
See also	Because Process Engine Setup creates each of these users with a default password, FileNet strongly recommends resetting the passwords to maintain system security. See "To reset administrative user passwords" in the appropriate platform section of Installation Guide for instructions.

Security tools and procedures

This topic lists common security procedures and the FileNet P8 Platform tool you would use to carry out the procedure.

To do this...	Use this tool...	Comments
Define users and groups	The resources of your directory server.	Users and groups reside in the directory service of your authentication provider, and are cached on Content Engine servers.
Add users and groups to object stores, classes, and security policies.	Enterprise Manager	See Start Enterprise Manager.
Define security policies	Security Policy wizard, on either Enterprise Manager or in the Workplace or Workplace XT applications	In Enterprise Manager, use the Security Policy Wizard. To assign the security policy to a document class you must use Enterprise Manager. Workplace or Workplace XT users can use the Security Policy Wizard.
Assign access rights to workflow queues and rosters	Process Configuration Console	Start Process Configuration Console from the Workplace Admin page or the Workplace XT Tools menu.
Set site preferences	Site Preferences	Start Site Preferences from the Workplace Admin page or the Workplace XT Tools menu.
Create document classes and folders, configure folders to serve as security parents to contained documents	Enterprise Manager	By default, all users can create top-level folders. As the administrator, you can modify the root folder's security if you wish to prevent users from creating top-level folders.

CE Bootstrap properties

Content Engine needs bootstrap information in order to create the GCD, and thereafter to provide the resources it needs to boot up. CE Setup creates and configures the bootstrap file in its "Setup Content Engine Bootstrap Properties" panel. Once CE Setup is finished and the new FileNet P8 domain is created and functioning, the bootstrap file will continue to provide the information below to allow CE to load. This file is named **CEMPBoot.properties**, contained in the CE EAR file.

There are two reasons why you would edit the bootstrap file:

- The value of the Username property has to be changed. (The account defined in the Username property is referred to in documentation as the "Content Engine system user".
- The JNDI datasources are no longer available.

With planning and normal precautions, these situations can likely be avoided, meaning you would never need to change the bootstrap file. However, if these situations do occur, use the Bootstrap Configuration Utility (BCU) to edit the file, as described below.

All deployments of the EAR file, for the purpose of adding additional CE servers to the FileNet P8 domain, must use identical values for the bootstrap properties. Therefore, any changes you make to the EAR file for a system in production must be made to all such EAR files. Depending on how your J2EE application server is configured, these changes could be made part of an automated deployment process.

Sample CEMPBoot.properties file

The following is a sample bootstrap file showing default values for all properties except that EncryptedPassword has already been set and programmatically encrypted by the Master Key:

```
com.filenet.gcd.CipherKeyLength=128
com.filenet.gcd.Username=CEMPAdmin
com.filenet.gcd.DigestAlgorithm=SHA
com.filenet.gcd.GCDConnection=jndiname\=Domain1DS;jndinamexa\=Domain1DSXA
com.filenet.gcd.EncryptedPassword=8dd56a9d9331b9cbe43536a42ce8146d
com.filenet.gcd.CipherAlgorithm=AES
```

These properties are defined in the following table:

CEMPBootstrap properties	Definition
CipherKeyLength	Default length of the cipher key that will be used to encrypt GCD credentials.
Username	A directory service account that is granted the role of application server administrator by CE Setup. This account will be used to logon to the application server and access the datasources named in the GCDConnection property. CE runs as this account, and it is therefore referred to in documentation as the "CE system user". The default value CEMPAdmin is only a suggestion and will be changed to whatever you enter into CE setup. See the entry for "CE system user" in Users and Groups for information about this account.
DigestAlgorithm	Default digest algorithm used to encrypt the Master Key. See Content Engine Encryption for information about the Master Key.
GCDConnection	The two datasource names that will be used in the creation of the GCD. Entered while running CE setup.

EncryptedPassword	The encrypted password of the user identified by the Username property. Entered while running CE setup. The encryption was carried out using the Master Key.
CipherAlgorithm	Default algorithm used to encrypt the Master Key.

Edit CE Bootstrap properties with the Bootstrap Configuration Utility

The Bootstrap Configuration Utility is a tool that edits the CEMPBoot.properties file. The bootstrap tool is contained in the BootstrapConfig.jar file, which is installed by the CE setup program into the Program Files\FileNet\ContentEngine\lib folder.

Usage

```
java -jar BootstrapConfig.jar ...
```

```
-h
-v
-e file -l
-e file -rf
-e file -j file
-e file [-fnq] [-b bits] [-c algorithm] ...
[-g name] [-i name] [-k key] [-m algorithm] ...
[-p password] [-s name] [-u name] [-x name] ...
[-y class] [-o boolean] [-j file]
```

```
-b,--keylength <bits> Cryptographic key length (in bits)
-c,--cipher <algorithm> Cryptographic cipher algorithm
-e,--ear <file> Filename and optional path of the EAR file
-f,--force Forces the utility to ignore warnings
-g,--dprovider <name> Cryptographic message digest provider
-h,--help Displays this help message
-i,--cprovider <name> Cryptographic cipher provider
-j,--targetear <file> File path of the EAR file to be patched with bootstrap info
-k,--key <key> Master cryptographic key
-l,--list Lists the current configuration
-m,--digest <algorithm> Cryptographic message digest algorithm
-n,--forcetext Forces the utility to store a plaintext password
-o,--outside <boolean> Forces master key safe mode
-p,--password <password> Password associated with username
-q,--quiet Suppresses text output
-r,--reset Restores the configuration to default values
-s,--datasource <name> JNDI datasource name (non-XA)
-t,--insert <filepath> Inserts the specified file into the EAR
-u,--username <name> Username of an app server administrator
-v,--version Displays version and copyright information
-x,--xdatasource <name> JNDI datasource name (XA)
-y,--handler <class> Keystore handler class name (with package)
```

Example

The following example shows how you would upgrade a new CEMPBoot.properties file by copying CEMPBoot.properties from a source (old) EAR's props.jar to a target (new) EAR. The properties in the target will be overwritten:

```
java -jar BootstrapConfig.jar -e /opt/<path>/Engine-ws.ear -j Engine-ws.ear
```

(-e introduces the source (old) EAR; -j introduces the target (new) EAR; ws denotes WebSphere; wl would denote WebLogic; jb denotes single JBoss, and jbc denotes the cluster installation of JBoss.)

NOTE If props.jar or CEMPBoot.properties do not exist in the target, they will be created based on the source. This will also fix some cases of malformed target EARs.

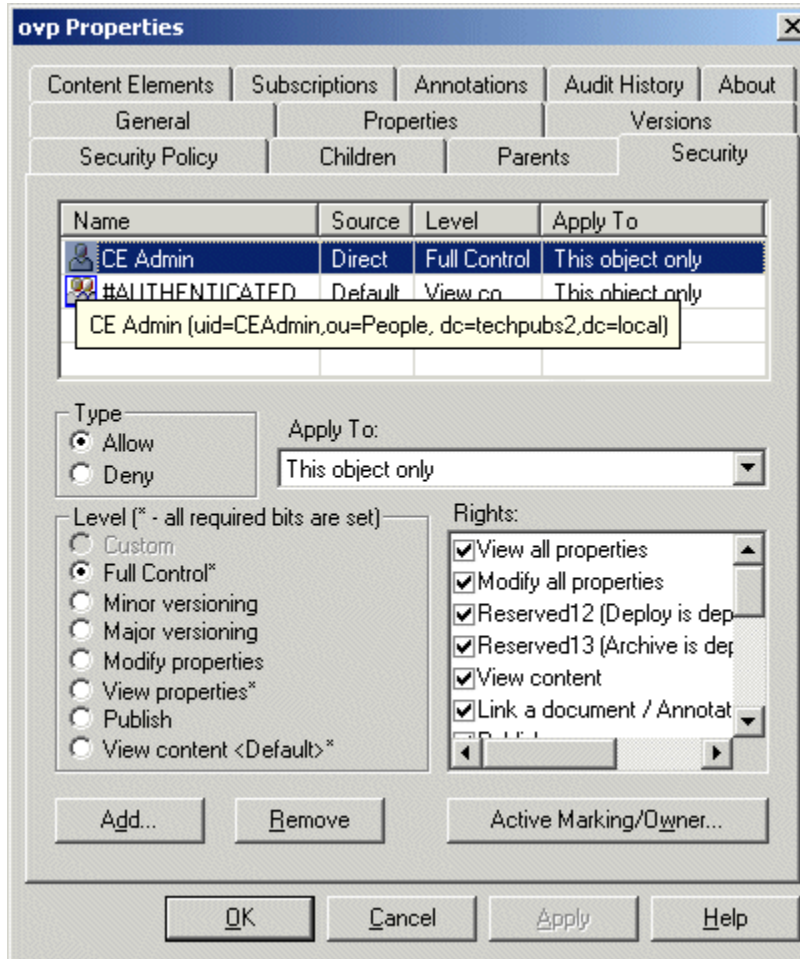
Digital signing

The Java applets are digitally signed using Verisign. When downloading an applet, the user must confirm that he wants to run the signed FileNet application. The digital sign expires (in approximately one year) and must be refreshed by installing an Application Engine fix pack.

Enterprise Manager's security editor

Using the Enterprise Manager, an administrator can view and modify an object's security by opening its property sheet and going to the Security tab. (About access rights has full definitions of these fields.)

The following shows the Security tab for a document named "ovp". This document's security tab (technically called its "ACL", or Access Control List) contains 2 names (technically called "ACEs", or Access Control Entries), each identified by several fields:



- Name: authentication provider's Display Name. If you hover your mouse above the name, you will see the following:
 - Sun Java System Directory Server and Novell eDirectory: full DN. (Example: uid=shawking,cn=users,dc=filenet,dc=com)
 - Active Directory: the display is constructed by the Content Engine by taking the first part of the samAccountName, adding "@", and then adding the domain from the actual UPN (for example, shawking@filenet.com). (This same construction appears in the results of a search for an account when using the Select Users and Groups dialog box.)
- Source: each ACE in this example has a different source type. The selected ACE is editable, which you can tell because the various regions (allow/deny, apply to, etc.) are not grayed out, which is because Direct permissions are editable. The ACEs whose Source is Template and Inherited are not editable, and when selected the rest of the security editor becomes grayed out.
- Level: the possible levels for the object type (in this case a document object) are listed with radio buttons lower down. The users and groups who are specified as object store administrative

groups while running the object store wizard appear on all ACLs with Full Control. You can change the level by selecting one of the radio buttons associated with the Levels listed below.

- Apply to: also called "inheritable depth, you can change the value using the Apply to control box if the ACE is editable.
- Type: Displays whether the ACE is allowed or denied, and also lets you change the value if the ACE is editable.
- (list of) Levels: List of security levels appropriate to the object. Different objects have different sets of security levels. The list shown is appropriate to documents, as it includes such things as the ability to publish and to create minor and major versions. A folder would have a different set of security levels. Notice that when Full Control is selected all the other "lower" levels are marked with an asterisk. The asterisk next to a Level means that it is included in the Level currently selected; this is the meaning of "all required bits are set".
- (list of) Rights: In this example, with Full Control selected as the Level, all Rights will be selected. If you were to deselect just one of them, "View all properties," for example, the Level would automatically be changed to Custom, which simply means that the collection of all selected Rights do not exactly match the requirements of the predefined Levels. If you were to reselect "View all properties" so that all the Rights were selected, the Full Control level would again be automatically selected.
- Add: Click to add or remove users and groups.
- Remove: Click to remove the selected ACE. (This does not remove the user or group from the authentication provider or from any other ACLs the ACE may be present on.)
- Active Marking/Owner: Click to view or edit the ownership of this object.

See About access Rights for a description of the various parts of the security editor, and Security tab for how to use it.

Enterprise Manager's Select Users and Groups dialog box

The Enterprise Manager uses the following dialog box to search for accounts in the P8 domain's configured directory service:

Display Name	Short Name	Principal Name
#CREATOR-OWNER		
#AUTHENTICATED-USERS		
CE Admin	CEAdmin	uid=CEAdmin,ou=People, dc=techpubs2,dc=local

See Select users and groups dialog box for a description of how to use this dialog box.

In this graphic, all three "object types" are selected: groups, users, and "special accounts" which are the #CREATOR-OWNER and #AUTHENTICATED-USERS accounts that are comparable to the "builtin" accounts used by Windows. By selecting or multi-selecting the accounts found and appearing in the Display Name column, and then by clicking OK, you will add those accounts to the ACL of the object from whose Security property page you launched this dialog box.

How Workplace applications display security

What Workplace and Workplace XT users see

The default level of access that users of an object store get when they are added while running the object store wizard (in its Specify Initial User Groups page), is View Properties on folders and and View Content documents. A user must have View Properties access rights to see an item in an object store. If the user can see the item, the user can also see the item's properties and security.

To open an object store, users must have View Properties access to the root folder.

The following illustration from Workplace shows the Security View for a document called Timesheet. Note that a Security Policy is currently attached to the document and controls the default security for Timesheet. Check marks identify the Allow permissions that apply to each User and Group in the current security list.

The example is a representation of an access control list (ACL). Each row, called an access control entry (ACE), lists a user or group (in the Title column) and that user's or group's Allow access rights to the object.

Document: **Timesheet** (Version 1.0, Released)

Security Policy: **Wade's Policy**

Assign Policy Unassign Policy

Domain: *filenet.com*

Users Groups Add New

Title	Owner Control	Promote Version	Modify Content	Modify Properties	View Content	View Properties	Publish	Remove
#AUTHENTICATED-USERS					<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>
HR_Managers		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
QSAdmins	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
PWDesigner					<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>
john.smith			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>
suser				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

uid=jsmith,ou=Shared, ou=Engineering, dc=filenet,dc=com

To save your changes, click [Apply](#)
 To exit Information, click [Exit](#)

TIP Move your cursor over any user or group display name to see the complete Distinguished Name for the specific user or group.

Permissions

The following illustration shows you the current permission settings for HR Managers. Both permission for Owner Control and Publish are set at Implicit Deny. Check marks in the System Allow column show that the currently assigned Security Policy set these permissions to the allow mode.

Security - Settings

Current Settings for: **HR Managers**

Distinguished Name: **cn=HR Managers,ou=groups,dc=filenet,dc=com**

Permission	Allow	System Allow	Deny	System Deny	System Notes
Owner Control	<input type="checkbox"/>		<input type="checkbox"/>		Implicit Deny
Promote Version	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Allow due to security policy
Modify Content	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Allow due to security policy
Modify Properties	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Allow due to security policy
View Content	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Allow due to security policy
View Properties	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Allow due to security policy
Publish	<input type="checkbox"/>		<input type="checkbox"/>		Implicit Deny

To proceed with changes, click [Continue](#)
To exit Settings, click [Exit](#)

System Notes are described in Manage Security.

Site and user preferences

Site preferences configure the appearance, behavior, and connectivity of the Workplace and Workplace XT applications. For example, a site preference determines whether users can view and modify security while adding a new document. Application users can optionally configure their own user preferences which will override the corresponding site preferences. Not all site preferences have a matching user preference.

The first user to access the application after installation must be a member of the Application Engine Administrators group. When this user signs in for the first time, the bootstrap initialization dialog displays. Settings entered during bootstrap initialization can be changed later in Site Preferences. Bootstrap initialization creates a preference file for the site and saves it in the /Preferences folder of the designated object store. It also updates the bootstrap.properties file and saves it to Application Engine on a network file server. The table below describes the preferences that have security implications.

Site Preference	Security options
Allow guests	Identify a guest account. Enter any valid user name (but not the domain Guest account or, if authenticating with iPlanet/Sun ONE Directory Server, a user with a blank password) and password to identify a guest account. When defined, the sign-on page allows a user to sign in by clicking Sign In as Guest.
Security info	Redirect sign-ins to an SSL server to secure user IDs and passwords between the Application Engine and the client.
Security page visibility	Show or hide the security page when application users add folders, add documents, or check in documents. If shown, users can see and modify security settings when performing these tasks. If using an Entry Template which specifies whether to show or hide the security page, it will override the preference setting.
Access Roles preferences	Assign Access Roles.
Keep-alive interval	Set the Session Timeout to limit the length of an inactive session.

Logging on and using Enterprise Manager

Enterprise Manager is the system administrator's primary tool for working with FileNet P8 security. By design, FileNet P8 applications like Workplace or Application Integration expose a subset of the full security interface required to completely understand and maintain FileNet P8 security.

Launching the Enterprise Manager

Enterprise Manager installs and runs only on machines running the Windows operating system. Anyone who can log on to the machine containing Enterprise Manager can launch Enterprise Manager. But EM itself will check the user's credentials by presenting a logon dialog box. See Logon to a domain for more information.

Logging on to an object store

Each object store has its own ACL, comprised of the administrative users added to the Object Store Wizard's Specify Administrators page. Since each object store could have a different set of administrators, each object store requires a logon before the Enterprise Manager will make its features available, as explained below.

However, each object store displayed in the Enterprise Manager will check the user's access rights before making its full feature set available, as follows:

To access these menu items you need this access right
Expand object store tree	Object store: Full Control
Object Store > <i>Object Store name</i> > Logon menu	Object store: Connect to store
Object Stores > New Object Store menu Object Store > <i>Object Store name</i> > New Storage Area menu Sites > <i>Site_name</i> > Content Cache Area > New Content Cache area menu	Domain root: Create new stores
Object Store > Delete menu	Domain root: Delete child objects
Storage Areas > Delete menu Content Cache Areas > Delete menu	Domain root: Delete child objects

See Logon to object stores for information on how to logon.

Network security

The security of your FileNet P8 installation on your corporate network is the responsibility of the security administrator and is not the subject of the topics in this Help section.

The topics in this section are only intended to explain what the relationship of FileNet P8 technology is in regard to issues like firewalls, digital signing, and SSL, and do not fully explain how to configure and maintain these important security components.

Content Engine Encryption

This topic provides information about the data and credentials encrypted by Content Engine. See also Edit CE Bootstrap properties.

Content Engine credential encryption

CE encrypts the following credentials that are stored in the GCD:

- Verity passwords.
- CFS-IS passwords.
- CFS-CS+/IICE passwords.
- Third-party fixed storage device credentials (Centera, Snaplock).
- Liquent passwords.
- Passwords used in publishing to password protect PDF files.
- Directory service user.

NOTE You can view and modify these IDs and passwords in Enterprise Manager. See View and modify FileNet P8 domain properties.

Any attempt to retrieve these password fields via any public API will return zero length binary data. This will result in a null value being emitted when any object containing a password field is exported. You will need to reset the password before you can use the imported object.

Encrypting credentials with the Master Key

CE uses a single 128-bit Master Key for encrypting and decrypting all credentials. The Master Key is generated during CE installation by using a one-way hash on a phrase supplied while running the CE setup program, in the Specify GCD Master Key panel. You should choose a phrase that follows these guidelines:

- It should not be a word that can be found in a dictionary or be easily guessable. You should use a non-obvious and preferably unique phrase.
- It must be at least 8 characters long. At least two characters, but not more than four, must be numeric. At least one character must be "special" (not alphanumeric; you can use any Java character that is not a letter or number.)
- There is no maximum length.
- Using mixed case also provides a measure of strength.

At install time, CE programmatically edits the EAR file and places the generated Master Key into an extremely safe place where it can be retrieved programmatically, but is not otherwise available for retrieval or inspection. Because the phrase used to create the Master Key is a string used to generate an encryption key and not a password, there is no need for the phrase to adhere to any policy for changing passwords. You should take appropriate precautions protecting the original phrase

You can regenerate a new EAR file with the same key by running the Bootstrap Configuration Utility, and supplying the original Master Key. The passwords stored in the database would be unaffected by the corrupted EAR file/Master Key, barring any serious disk/file system corruption problems on the the host machine. If you enter the same password for the Master Key, you will get the same encryption key, and you will not need to re-enter all the passwords stored in the database.

An EAR file with the same Master Key will have to be used across all servers in the FileNet P8 domain. It would be possible to support different versions of the EAR file with the same generated Master Key (rolling upgrade case).

Procedure for resetting keys

If the Master Key is lost or corrupted, the credentials mentioned above would have to be reset by the GCD administrator if any of the encrypted information in the GCD had to be changed or reset.

Resetting the keys must be done by someone who knows the existing passwords (for Verity/Liquent/Directory service user/fixed storage devices, etc.) or who has the ability to reset those passwords. When the Master Key is lost, corrupted or changed, you will have to follow these basic steps.

- Re-run the bootstrap utility (BCU) to re-encrypt the JNDI bootstrap user credentials that are stored in the CEMPBoot.properties file.
- Re-deploy this EAR file to each CE server in the FileNet P8 domain.
- Logon to Enterprise Manager as a GCD administrator. Re-enter/reset the passwords for each of the following:
 - Each Directory Service Configuration.
 - Each IsolatedRegion.
 - Each CFS-IS or CFS-CS fixed device.
 - Each Verity domain configuration.
 - For each PublishStyleTemplate defined on the system for PDF documents, that makes use of PDF passwords, have the person who created the template go into the Publishing Style Template Manager application and re-enter/reset the PDF passwords.

Contact FileNet Customer Service and Support web site for assistance.

Symmetric encryption of sensitive data at rest in the GCD

CE uses symmetric key encryption to encrypt sensitive data at rest. It uses a single encryption algorithm and strength using 128-bit AES encryption. The CE credentials encryption module will easily leverage an alternate encryption standard defined in any JCE (Java Cryptographic Extension).

Process Engine server piggybacks on the CE API for the purpose of authenticating clients. Users of the PE Java API will perform a JAAS logon, and then make a call to the CE server. The CE server will authenticate the caller, establishing the validity of their JAAS credentials, and return an identity token. The PE API client will then pass this identity token to the PE server. The PE server will rely on this token to establish the client's identity. This mechanism relies on the use of several symmetric encryption keys. Refer to Process Engine Authentication for details about these keys and their usage.

Asymmetric encryption of sensitive data sent over the wire

PKI encryption is used to encrypt sensitive data sent over the wire between clients and CE. Enterprise Manager provides out-of-the-box support for setting the passwords used to access Verity collection, and various fixed content provider implementations (CFS-IS, Centera, SnapLock, and others).

The public key of the pair is a property of the FileNet P8 Domain object. EM will use the public key to encrypt password fields that are to be sent to the server.

The private key of the pair will also be stored in the GCD, encrypted with the CE Master Key. The private key will not be accessible via any client API; only CE server code has access to it. Asymmetric encryption is done using 1024-bit keys with the RSA algorithm. The server will decrypt incoming password properties with the private key. These values will then be encrypted using the FileNet P8 Domain's Master Key prior to being persisted. The PKI key pair used for asymmetric encryption will be generated as a part of creating a FileNet P8 domain, via calls to the standard Java Cryptography Extension (JCE) provider. Any time CE generates a new Master Key for a FileNet P8 Domain object, it will re-generate this PKI key pair as well.

Content encryption

CE does not provide features for encrypting the data contained in content files being transmitted between CE and the fixed content storage device or file storage area. You should therefore consider whether and how to provide this type of security.

Content files are secure while in storage. See File storage area and content cache area access rights.

SSL and data privacy

Both HTTP and LDAP can use Secure Sockets Layer (SSL) to secure authentication credentials and data sent over a network:

- SSL can be used to verify the identity of one or both of the parties engaged in a network connection.
- SSL can also provide data integrity (proof that the data has not been modified since the sender sent it).
- SSL can provide data confidentiality (obscuring the data from third parties observing the dialogue over the connection).

For information on using SSL with FileNet P8, refer to the following topics.

FileNet P8 engine	SSL documentation
Application Engine	<p>"Configure Application Engine for Symmetric Encryption" and "Set up Application Engine SSL Security" in the Installation Guide.</p> <p>About Application Engine Security (Application Engine Administration)</p> <p>Bootstrap preferences (Application Engine Administration)</p> <p>"Set Up Content Engine and Client Transport SSL Security" in the IBM FileNet Workplace XT Installation Guide.</p> <p>NOTE Where data privacy is not important, it is sufficient to redirect logins to SSL to protect user names and passwords.</p> <p>NOTE You can run FileNet P8 applications, such as Workplace and Workplace XT, under SSL to secure both properties and content. A web site running under SSL has a URL beginning with https://. If you use SSL for these applications, you must support an additional non-SSL site for the designer applications, which are Java applets.</p>
Content Engine	"Set up Content Engine and Client Support SSL Security" in the FileNet P8 Platform Installation Guide.
Process Engine	"Set up Process Engine SSL Security" in the FileNet P8 Platform Installation Guide
Verity	Verity documentation. (Verity has an option to use SSL when communicating with directory servers.)

How to...

In addition to the security procedures in this section, the following security-related procedures are covered elsewhere in Help:

- Creating security policies
- Using the Enterprise Manager's security editor
- Specifying object store administrators and users and administrators in the Object Store wizard

Add users and groups to a class

Classes that are instantiable, like the document class, have two security tabs:

- The **Default Instance Security** tab contains the ACL that will be applied to new instances of the class. (For example, a document is an instance of a particular document class.) New users and groups who will be users of the objects based on a class should be added to the Default Instance Security tab of the class: they will need the Create Instance permission that is contained on the Security tab.
- The **Security** tab governs the security of the class itself (who can subclass it, delete it, change its properties, etc.). New users and groups who will be administrators of the class should be added to the Security tab of the class.

To add new users/groups to the Default Instance Security ACL of a class

1. Use Enterprise Manager to open the property sheet of the class and go to the **Default Instance Security** tab. Use the Add button and use the Select Users or Groups dialog box. Click OK to close the dialog box.
2. Use the security editor to examine and if necessary change the default permissions of the new grantees.

NOTE The default security level of a new user or group is View Properties. Full administrative control is conferred by having the Full Control level. Make sure you should grant the new user or group the permissions they will need.

3. Click OK when you are done. New instances of the class will include the newly added grantee with the default access permissions you selected.

NOTE Modifying a class does not affect existing objects that are instances of that class. For example, adding users to the Default Instance Security ACL of a document class does not add those new users to existing documents based on that class.

Add users and groups to a single object

Securable objects like documents and folders get their initial security from the Default Instance Security ACL of their class and possibly also from the class' default security policy, if there is one. Authorized users can add to or change this initial security. Directly changing a single object's security means that only that one object will receive the changes. If the object is a document, these direct changes to a single version will continue to appear on later versions of the same document.

To add new users and groups to a single object

1. Ensure that the user or group already exists in the authentication Realm that was configured while installing and configuring the FileNet P8 domain.
2. Use the Enterprise Manager to open the property sheet of the object and go to the Security tab. Use the Add button and add the users or groups in the Select Users or Groups dialog box. Click OK to close the dialog box.
3. Use the security editor to examine and if necessary change the default permissions of the new grantees.
4. Click OK when you are done.

Add or remove a GCD administrator

Immediately following installation Enterprise Manager launches the Configure New Domain Permissions wizard so you can assign the GCD administrative role to additional additional users and groups. You can view and edit this list at any time later on using the procedure below.

For more information, see the entry for GCD administrator in Users and Groups.

To add or remove GCD administrators

1. Logon to Enterprise Manager as a GCD administrator.
2. Right-click the FileNet P8 domain node and select Properties.
3. To add GCD administrators, select the Security tab and click Add. Use the Select Users and Groups dialog box to find and add users and groups. Click OK to return to the Security tab. Apply the Full Control permission level to all the accounts you just added. Click OK when you are done
4. To remove GCD administrators, select the Security tab. Select the user or group you want to remove and click Remove.

NOTE Since there must always be at least one GCD administrator, Enterprise Manager will prevent you from removing all entries.

5. Click OK when you are done.

Add or remove an object store administrator

Each object store has its own list of object store administrators, initially created while running the object store wizard. You can view and edit this list at any time later on. For more information, see the entries for object store administrator and GCD administrator in Users and Groups.

To add or remove object store administrators

1. Logon to Enterprise Manager as a GCD administrator.
2. Right-click the object store whose list you want to edit and select Properties.
3. To add object store administrators, select the Security tab and click Add. Use the Select Users and Groups dialog box to find and add users and groups. Click OK to return to the Security tab. (To remove object store administrators, select the Security tab. Select the user or group you want to remove and click Remove. Click OK when you are done.)
4. Grant the Full Control permission level to all the accounts you just added. Click OK when you are done. These new users and groups will have the rights of an object store administrator for all new objects created in the object store going forward.
5. Run the Security Script Wizard to provide these new object store administrators access to existing objects. See Update object store with new users and groups.

Allow or disallow security inheritance

Use the procedure below to enable or disable inheritance from a parent folder to a child folder.

To allow or disallow inheritable parent permissions to a child folder, using the Enterprise Manager

1. Right-click the folder, select Properties, and select the General tab.
2. Select (to allow) or clear (to disallow) the checkbox labeled "Inherit parent permissions".
 - Changing to selected: allows the inherited permissions from the parent folder to propagate to this folder.
 - Changing to clear: prevents the object from inheriting permissions from the parent in the future.

NOTE Only those permissions in the parent folder that are inheritable will actually be inherited by the child folder. See Understanding inheritance for information about inheritance and inheritable depth.

NOTE For the Root folder, which by definition has no parent folder, checking or unchecking the check box has no effect.

See Configure a document's security parent folder for configuring security inheritance for documents and custom objects.

Allow users to add items to a folder

Allow users to add subfolders to a folder

1. Display the folder's security.
2. Add the groups and grant them the Modify Properties security level.
3. Refresh the folder.

Allow users to add documents to a folder

4. Display a folder's security.
5. Add the groups and grant them the Add to Folder security level.
6. Refresh the folder.

Configure a folder's security inheritance

This topic explains how to configure a folder to be a security folder to documents or custom objects. For information on how to configure those objects to inherit permissions from the folder, see [Configure security inheritance](#).

Run this procedure as part of your initial security configuration, after you have added document classes and configured their Default Security Instance tabs. It is best to establish security inheritance before putting an object store into production.

For more information about inherited security see:

- Understanding security inheritance
- About access rights

Use Enterprise Manager to configure a security folder

1. Logon to Enterprise Manager as object store administrator.
2. Open the Object Store node and select the Root Folder.
3. Right-click the folder whose security will be inherited and select Properties. Click the Security tab. The users or groups whose security you want to be inherited should appear on the list of names.
 - a. Make sure the ACEs whose security will be inherited have an Apply to setting of either This object and immediate children or This object and all children.
 - b. If the users or groups are not already on the folder's list of names, click the Add button to open the Select Users and Groups dialog box. Use the Find capability to find the users or groups you want and then add them to the folder's security list of names.
4. Select the user or group whose permissions you want to be inherited and make one or more of the following changes:
 - a. Choose Allow or Deny.
 - b. In Apply to, select This object and immediate children (for only one level of inheritance) or This object and all children (for infinite levels of inheritance).
 - c. Choose the security level or individual rights that will be inherited. In most cases you will choose from the rights marked (Inherit only).
5. Click Apply or OK.
6. If necessary, configure the target documents and custom objects to inherit the Inherit only permissions from this folder. See [Configure security inheritance](#).

Configure security inheritance

This topic describes the following ways to configure security inheritance between objects:

- Using the deprecated (but still active) Security Parent property (first procedure below)
- Using the class property Security Folder (second procedure below)
- Creating one or more custom object-valued properties and setting its Security Proxy Type property to Inherited, and then assigning that property to a class (third procedure below).

Run these procedures as part of your initial security configuration, after you have created document classes and configured their default security instance tabs. It is best to establish security inheritance before putting an object store into production.

For more information about inherited security see:

[Understanding security inheritance](#)

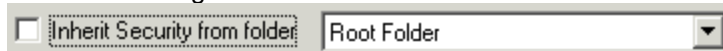
[About access rights](#)

NOTE You can use this procedure for custom objects also. Just substitute "custom object" wherever you see "document".

Use Enterprise Manager to designate a folder as a security parent, using Security Parent

You can configure a document to inherit permissions from a folder. The folder may contain the document, but this is not required. (Earlier versions of Content Engine did require that the security parent folder contain the document, but this requirement has been removed with the introduction of the SecurityFolder property.)

1. Logon to Enterprise Manager as object store administrator.
2. Open the Object Store node, select the Root Folder, and navigate to the folder containing the document whose security inheritance you are configuring.
3. Click the folder's Security tab. Make sure the folder has ACEs whose Apply to setting is either This object and immediate children or This object and all children. See [Configure a folder's security inheritance](#) for reference.
4. Click OK to close the folder's property sheet.
5. Right-click the document, select Properties, and then select the property sheet's General tab. You will see the following:



6. Select the Inherit Security from folder checkbox. Then select a folder from the drop-down list. (All folders containing the document will appear in the list. If the folder you want is not in the list you will need to stop this procedure, file the document in the folder, and then start again.)

NOTE After the upgrade to CE 4.0.1, this checkbox appears exactly as it did in earlier releases. However, selecting it actually sets the new SecurityFolder property and not the SecurityParent property as it did formerly and which is being deprecated. However, because the SecurityParent feature is still supported, the dropdown box will display only those folders that contain the document. This Enterprise Manager behavior therefore mimics the SecurityParent behavior which depends on containment, even though it is in fact using the SecurityFolder property which does not require containment of the document by the folder. Custom applications that have been coded using SecurityParent will continue to function without change.

7. Click Apply or OK.

8. Click the document's Security tab and confirm that it has inherited ACEs from the security folder. The inherited ACEs will show a Source type of Inherited. If the required rights do not appear, make sure that they are configured to be inheritable on the folder. See [Configure a folder's security inheritance](#).

Use Enterprise Manager to designate a folder as a security folder, using Security Folder

Similar to the SecurityParent procedure above, this procedure uses the Security Folder property, a standard property of every document and custom object class.

1. Logon to Enterprise Manager as object store administrator.
2. Open the Object Store node, select the Root Folder, and navigate to the folder that will serve as the Security Folder.
3. Right-click the folder and select Properties. Click the folder's Security tab. Make sure the folder has ACEs whose Apply to setting is either This object and immediate children or This object and all children. See [Configure a folder's security inheritance](#) for reference. Click Cancel or OK to close the folder's property sheet. You should see the folder's icon listed in Enterprise Manager's tree view.
4. Right-click the folder and select Copy Object Reference.
5. Now navigate to the folder containing the document whose security inheritance you are configuring.
6. Right-click the document and select Properties. Select the property sheet's Properties tab.
7. Scroll down the list of properties and find Security Folder. Its Property Value cell will display <Value Not Set> if there is no value yet for this property.
8. Click the Property Value column. The Set Object Value dialog box will appear. Click OK to set the value. The Select Object from Paste Buffer dialog box will appear and will list the object reference you copied earlier under the Object Name column.
9. Select the appropriate Object Name and click OK. You will see the name of the folder appear as the Property Value for the Security Folder property.
10. Click Apply to apply the changes you just made and keep the document's property sheet open.
11. Click the Security tab, and confirm that the Security Folder's inheritable ACEs appear, with Source type of Inherited.

Use Enterprise Manager to configure security inheritance using a custom object-valued property

In addition to the methods explained above, you can also create pairs of security-passing and security-inheriting objects, as follows:

1. Logon to Enterprise Manager as object store administrator.
2. Copy the object reference of the object whose security will be inherited. (This object will become a security parent as a result of this procedure.) This object must have at least one inheritable ACE (one whose Apply to setting is either This object and immediate children or This object and all children). For reference, see [Copy object reference](#).
3. Launch the Create a Property Template Wizard to create the property that will establish the connection between the two objects. For reference, see [Create a property template](#).
 - a. Give the new template a name.
 - b. Select Object for the data type.
 - c. On the Single or Multi-Value? step of the wizard, select Single. Click the More... button.

- d. On the More tab of the dialog box that displays, for Security Proxy Type select Inherited. (This value will appear as the integer 2 when viewed in the document's property grid.) Click OK.
 - e. Click Next and Finish to complete the wizard.
4. Assign the new property template to a new or existing class. The following procedure assumes the class already exists. For reference see Assign properties to a class.
- a. Right-click the class and select Add Properties to Class. This launches the Add Properties to a Class Wizard. Click Next.
 - b. In the Select Properties panel, select the Show Object Type checkbox and in the Available column select the property you just created above. Click Add>> to add the property to the Selected column. Click Next.
 - c. In the Select Property Attributes panel, select the property you just added to the class and then click More. The property template's property sheet opens.
 - d. In the property template's property sheet, click the More tab.
 - i. For Required Class, use the drop-down menu to select the class of the object whose object reference you copied above. For example, if that proxying object is a document, you would select its exact class or subclass.
 - e. Click Next and then click Finish to finish the Wizard.
5. (Optional) Assign a default value to the object-valued property. This step is optional but can be used, if appropriate, to automate the process of establishing the connection to the object providing security inheritance. If you do not set a default value, Enterprise Manager will request an object reference each time you create a new object that references that object-valued property for its inherited security. (This step assumes that there is a single inheritance-providing object for this particular custom property.)
1. Right-click the class you used in the step above and select Properties. Select the Properties tab.
 2. Scroll down and find the Property Definitions row. (This is not the same as selecting the Property Definitions tab of the property sheet.)
 3. Click the down arrow in the Property Value column. The list of all custom properties drops down.
 4. Select the object-valued property you just created. Its property sheet will display. Click the Properties tab of that property sheet.
 5. Scroll down and find the Property Default Object row and click its Property Value cell. If you have not yet set the value, you will get a dialog box asking you to select OK to set the value. Click OK and the Select Object from Paste Buffer will appear.
 6. Select the object that will be supplying the inherited security and click OK. Click Close and OK to close the class property sheet.
- If the object you need is not in the list, click Cancel and start this procedure again, being careful to follow the step describing how to copy the object reference of the object whose security will be inherited.
- If the Propagate Metadata Changes dialog box opens, you must decide, based on the requirements of your security design, whether the new property you just added to a class should be propagated down to all subclasses. We will not propagate for this procedure; therefore in the Updated Property Definitions box do not select the property definition we just created. Click OK to return to Enterprise Manager.
6. Create a new document using the class we have been using in this procedure. (If you have not assigned a default value as optionally described above, you will be prompted for an object reference. Set the reference using the object reference you copied.)

7. Examine the new document's Security tab and confirm that it has inherited ACEs from the security parent object. The inherited ACEs will show a Source type of Inherited. In order to change the access rights of this inherited ACE, you would change it on the source document; the changes will automatically be updated on the target document.
8. Repeat this procedure as many times as required by your security design.

Configure inheritable depth (Apply to)

For each user and group listed in a security parent's ACL, the "Apply to" setting determines the inheritable depth of the ACE.

1. In Enterprise Manager, display an object's Security page.
2. Select the user or group whose inheritable depth you wish to edit.
3. Make a selection from the Apply to pull-down menu and click Apply or OK when you are done.

Configure multiple authenticating attributes

The following procedure provides a general list of steps to follow for configuring your application server so that users can login using both shortname and distinguished name. You must first configure the Content Engine (CE) application server's authentication parameters, then CE's authorization parameters. Then - in some cases - you must also configure the Application Engine's (AE) authentication parameters.

NOTE You can carry out this procedure before or after installing CE and AE. If you have already installed CE, and if you selected to install the "Application Server Authentication Provider" component, the CE setup program has already configured your application server's authentication parameters for one authenticating attribute, for example, using shortname (cn).

The following procedures use the terms *shortname* and *longname* which typically map to the following specific LDAP attributes:

Directory Server	typical shortname equivalent	typical longname equivalent
Active Directory	sAMAccountName	userPrincipalName or DN
Sun	uid	DN
Novell	cn	DN
IBM	cn	DN

To configure for multiple authenticating attributes (shortname and distinguished name)

1. Logon to CE's application server as an administrator.
2. Do the following, depending on your application server:
 - a. **WebSphere** - In the profile containing Content Engine:
 - i. Set the user filter to:
 (&!(*shortname*=%v)(*longname*=%v))(objectcategory=user)
 - ii. Set the User ID Map to:
 user:*shortname*;user:*longname*
 - b. **WebLogic** - In the domain containing CE:
 - i. Create two authentication providers, one using shortname and another using longname.
 - c. **JBoss** - In the JBoss server containing CE:
 - i. Edit login-config.xml to allow both types of login. The following example provides a general idea. Notice, in the two versions of the <authentication> section, the different entries for baseFilter and roleFilter:

```
- <application-policy name="ibm">
- <authentication>
- <login-module code="org.jboss.security.auth.spi.LdapExtLoginModule" flag="sufficient">
<module-option name="java.naming.provider.url">ldap://yourURL:389</module-option>
<module-option name="java.naming.security.authentication">simple</module-option>
<module-option name="allowEmptyPasswords">>false</module-option>
<module-option name="bindDN">cn=test1,CN=Users,DC=yourDC</module-option>
<module-option name="bindCredential">test1</module-option>
<module-option name="baseCtxDN">CN=Users,DC=yourDC</module-option>
<module-option name="baseFilter">(longname={0})</module-option>
<module-option name="rolesCtxDN">CN=Users,DC=yourDC</module-option>
<module-option name="roleFilter">(longname={0})</module-option>
<module-option name="roleAttributeID">memberOf</module-option>
<module-option name="roleAttributeIsDN">>true</module-option>
<module-option name="roleRecursion">-1</module-option>
</login-module>
```

```
- <login-module code="org.jboss.security.auth.spi.LdapExtLoginModule" flag="sufficient">
<module-option name="java.naming.provider.url">ldap://yourURL:389</module-option>
<module-option name="java.naming.security.authentication">simple</module-option>
<module-option name="allowEmptyPasswords">false</module-option>
<module-option name="bindDN">cn=test1,CN=Users,DC=yourDC</module-option>
<module-option name="bindCredential">test1</module-option>
<module-option name="baseCtxDN">CN=Users,DC=yourDC</module-option>
<module-option name="baseFilter">({shortname={0}})</module-option>
<module-option name="rolesCtxDN">CN=Users,DC=yourDC</module-option>
<module-option name="roleFilter">({shortname={0}})</module-option>
<module-option name="roleAttributeID">memberOf</module-option>
<module-option name="roleAttributeIsDN">>true</module-option>
<module-option name="roleRecursion">-1</module-option>
</login-module>
</authentication>
</application-policy>
```

CAUTION When using JBoss 4.0.5, if CN=Users is missing from the rolesCtxDN tag, you will not be able to log on to Enterprise Manager, which will throw an incorrect user name/password exception.

3. Logon to EM as a GCD administrator.
 - a. Right-click the EM Root Folder, and then click Properties.
 - b. Click the Directory Configuration tab.
 - c. Select the directory configuration entry and click Modify if you changed an existing authentication configuration,
 - d. Click Add if you added a new authentication configuration and complete the Create a Directory Configuration Wizard using the same values you just entered into the application server's authentication configuration.
 - e. Make the same changes you made in your application server.
4. If your authentication design requires that AE's application server's authentication parameters exactly match those of CE's application server, logon to AE's application server as an administrator.
 - a. Make the same authentication changes on your AE's application server that you made for CE.

If your AE is installed on a different application server type than CE (only supported when using Web Services transport) between AE and CE, achieving an exact match of multiple login configuration might require experimentation and careful testing.

Configure multiple realms

FileNet P8 support for multiple realms depends on the features of the Content Engine's J2EE application server. Multiple realm support is available for WebLogic, JBoss, and WebSphere 6.1, but not for WebSphere 6.0 or 5.1.

The Installation and Upgrade Guide describes how to configure an initial authentication realm as part of Content Engine (CE) setup. This topic explains how to add additional realms, and assumes you have already successfully installed Content Engine.

See Configuring authentication in the Security Overview for a high level picture of the separate processes of authentication and authorization.

To configure multiple realms with Active Directory

To configure all Active Directory Windows domains in a particular forest, create one authentication provider in your J2EE application server that references the Windows Domain Controller hosting the Global Catalog. (This configuration does not depend on the application server type and version.)

To configure multiple realms using WebLogic

WebLogic supports multiple security realms and multiple authentication "providers" per realm. In WebLogic 8, CE is installed into the default security realm with the default name of "myrealm". In WebLogic 9, Content Engine is installed into the default security realm; if there is more than one, the **Summary of Security Realms** in the WebLogic Administration Console displays the default.

This procedure assumes:

- You have successfully installed CE with WebLogic and that you selected to install the CE's Application Server Authentication Provider component.
 - Or that you configured WebLogic authentication yourself before installing CE and did not install the Application Server Authentication Provider component.
1. Log on as WebLogic administrator to the WebLogic domain that contains CE. In the realm in use by CE, create a new authentication provider using properties that point to the additional naming context on your directory server. Restart the application server.
 2. Log on to Enterprise Manager (EM) as GCD administrator. Run the Create Directory Configuration wizard. Enter the same directory service configuration property values that you just added to the authentication provider.
 3. Repeat steps 1 and 2 for each additional directory server naming context that you want to configure as FileNet P8 realms.
 4. Grant the new users and groups access to objects, for example, by logging on to EM as object store administrator and adding the new accounts to document classes.
 5. Test the new configuration by logging in to a client application with an account residing in the newly configured realm.

For Weblogic configuration details, see the Realm Configuration sections of the topics describing your directory service provider.

To configure multiple realms using WebSphere 6.1

Multi-realm configuration is not possible using WebSphere 5.1 or 6.0, due to limitations in those WebSphere versions as well as with Tivoli Access Manager's SSO solution. WebSphere 6.1 supports multi-realm configuration through its new federated user repository feature. Because federated user repositories must be configured before installing Content Engine, this procedure is explained in the "Configure an Application Server for Content Engine (WebSphere)" task of the Installation Guide.

To configure multiple realms using JBoss

JBoss supports multiple authentication realms by allowing multiple authentication sections in its configuration file `login-config.xml`.

The easiest way to configure multiple realms this is to let CE Setup create the initial authentication section in the JBoss file `login-config.xml` in the server's `\conf` directory (for example: `...\server\myserver\conf\login-config.xml`). After installation you can directly edit the xml to change the initial values or add additional authentication sections that point to additional naming contexts on your directory server.

This procedure assumes you have successfully installed CE with JBoss and that you selected to install the CE's Application Server Authentication Provider component (or that you configured JBoss authentication yourself before installing CE and did not install the Application Server Authentication Provider component).

1. Open `login-config.xml` in an editor. Find the `<authentication>` section. It will look similar to the following:

```
<application-policy name = "FileNet">
  <authentication>
    <login-module code="org.jboss.security.auth.spi.LdapExtLoginModule" flag="required">
      <module-option name=" java.naming.provider.url">ldap://yourserver:389</module-option>
      <module-option name=" java.naming.security.authentication">simple</module-option>
    ...
    <module-option name="directory server authentication attributes">required FileNet P8
values</module-option>
    ...
  </login-module>
</authentication>
</application-policy>
```

2. Make a copy of the `<authentication>` section and paste it right after the first. Change the required FileNet P8 values in the new section so that it points to the new realm. See the Directory service providers section for information about each application server's attributes and values.
3. Log on to EM as GCD administrator. Run the Create Directory Configuration wizard. Enter the same directory service configuration property values that you just added to the authentication provider.
4. Repeat steps 1 and 2 for each additional directory server naming context that you want to configure as FileNet P8 realms.
5. Test the new configuration by logging in to a client application with an account residing in the newly configured realm.
6. Grant the new users and groups access to objects by logging on to EM as object store administrator and adding the new accounts to document classes.

Deny an object store administrator access to a document

The following procedure provides specific steps needed to create a marking that denies an object store administrator access to a document. It follows the general steps described in the procedure [Create a marking set](#).

Creating a marking set and applying it to a class of objects is a multi-step procedure. Refer to [Markings](#) if you need more information about the options mentioned below (and especially about how to set the marking's Constraint Mask and Security). The following procedure is a sample, designed to accomplish one simple task and could be modified to accomplish additional tasks required by your security design.

CAUTION Any time you deny basic administrative privileges, like to an object store administrator, you run the risk of unintended errors

NOTE This procedure will deny access only to new documents based on the document class to which you add the property template created below. It will not automatically or instantly deny access to any existing documents.

To create a marking set that will deny an object store administrator access to a document

Create the marking

1. Right click the Enterprise Manager's P8 domain root property sheet and select the Marking Sets tab. All marking sets already defined for the P8 domain appear in this list.
2. Click Create to launch the Create a Markings Set Wizard. The Welcome page of the wizard opens.
3. Click Next. The Select Markings Set type page opens.
4. Select List (non-ordered) and click Next. The Add Markings page appears.
5. Type a name for the marking set. For example, you could enter "Deny Admin Access". Click New Marking. The Create New Marking property sheet appears.
6. In the General tab, type a name for the Marking Value that will deny object store administrators access to the affected documents. For example, you could enter "Modify Owner" (since it is the Modify Owner right that will be denied in the constraint mask).
7. Click the Constraint Mask tab and click Deselect All, then select only Modify owner. This is the access right that will be denied by the marking.
8. Click the Security tab and then click Add. This opens the Select Users and Groups dialog box.
 - a. Use the dialog box to find and then add the names of all users and groups that should be allowed access to the document. For these accounts make sure the Use Marked Objects right is selected. The marking's constraint mask will have no effect on these users and groups.
 - b. Use the dialog box to find and then add the names of all users and groups that should be denied the Modify Owner right. For these accounts make sure the Use Marked Objects right is deselected. The marking's constraint mask will apply to these users and groups.

(The other two rights, Add Marking and Remove Marking, are administrative permissions. For explanations, see [Markings](#).)
9. Click OK to close the Select Users and Groups dialog box and return to the Create New Marking property sheet's Security tab.
10. Click Next to proceed to the Completing the Create a Marking Set Wizard page.
11. Click Finish to complete and close the wizard. Click OK if a dialog box pops up to confirm that you have successfully created a Marking Set. The marking set you just created shows up in the P8 domain property sheet's Markings Set tab.

12. Click OK to close the property sheet return to the default view of the Enterprise Manager.

Create the Property Template

13. Create a new property template. Right click the Enterprise Manager's Property Template node and select New Property Template. This launches the Create a Property Template wizard.
14. In the Wizard's Welcome page, click Next.
15. In the Name and Describe the Property Template page, enter a name and optional description. For example, you could enter "Marking – Deny Admin Access". Click Next.
16. In the Select the Data Type page, select String type. Click Next.
17. In the Select a Choice List page, select Assign marking set and pick the marking set you created from the drop down list. Click Next.
18. In the Single or Multi-Value? page, select Single for the purposes of this sample. Optionally, you could click More and use the More tab to specify the desired marking value as the default value of the string property, rather than having to set it on each new document
19. Click Finish to complete and close the wizard. Click OK if a dialog box pops up to confirm that you have successfully created a Property Template. The property template you just created shows up in the Enterprise Manager's list of property templates.

Assign the property template to a class

20. Select and expand the Enterprise Manager's Document Class node. Select the document class that should be associated with the new marking set and select Add properties to class. This launches the Add Properties to a Class Wizard which will let you add a property based on the new property template you just created. The new custom property shows up on the right side of the Enterprise Manager when you select the class.

Create a document based on the class

21. Run the Create a New Document wizard to create a document based on that document class. Unless you set a default value for the property as described above, the wizard you use to create the object will not set the marking value for the marking-enabled property. You have to do that in the next step.
22. Set the markings on the document (but see the Note below):
 - Right-click the document and select Properties. Select the Properties tab.
 - In the property grid's Property Name column, find the property you added to the class in the step above.
 - Click either the Property Value cell or the cell's drop down arrow. Any markings that have already been set as values for the property will appear in the drop down list. You will also see Edit List... .
 - Click Edit List... to open the Add/Remove List Items dialog box which you can use to add or remove markings. Note that you will only be able to see those markings that you have permissions to apply.
 - When you are done setting values, click OK to close the object's property sheet. The object store administrators whose names were added to the first marking above will be able to run the Enterprise Manager but will not be able to see the documents.

NOTE If you created a single marking and if you set the default value as described above, you should not have to explicitly set the value as described in this step.

Modify an object's security

1. Open the object store and navigate to the object.
2. Display the object's properties, and then click the Security tab.
3. Use the security editor to add or remove users or groups and to set their security..

Restrict access to the root folder

When a user opens an object store using Workplace or Workplace XT, the user sees the contents of the object store's root folder. By default, the root folder's security grants Modify Properties access rights to the nonadministrative groups selected during installation. This means that unless you change the root folder's security, members of those groups can add folders at the top level of the folder tree. To specify who can add documents and folders at the top level:

1. In Enterprise Manager, right-click the Root Folder, and then click Properties.
2. Click the Security tab.
3. Add the groups who are allowed to add top-level folders, and set the Modify Properties access right to Allow for those groups.
4. Deny Modify Properties for those groups who are not allowed to add top-level folders.

Set security on workflow queues and rosters

You set access rights on workflow queues and rosters using Process Configuration Console, from the Admin page in Workplace or the Tools menu in Workplace XT. To use Process Configuration Console, you must be a member of the group defined by the Access Role PWConfiguration. To make configuration changes to queues and rosters, you must be a member of the Process Engine Configuration Group.

See Set security levels in Process Configuration Console online help for detailed instructions.

Take or change ownership

See Object ownership for reference information.

To take ownership of a document using the Enterprise Manager

1. Right-click the document, select Properties, and click the Security tab.
2. Click Advanced, and then click the Owner tab.
3. Select Take ownership of this object and click OK.

To change owner of a document using the Enterprise Manager

4. Right-click the document, select Properties, and click the Security tab.
5. Click Advanced, and then click the Owner tab.
6. Select Change owner to and click Find to open the Select Users and Groups dialog box which you will use to find and select the object's new owner.
7. Click OK.

Update object store with new users and groups

You can easily add new users to an object store that is already in production, as long as you only want them to have permissions on objects created subsequent to their addition. See *Add users and groups to a class* for this procedure.

However, adding new users so that they have default permissions to all existing objects requires a different procedure, and uses the script `WizUpdateOSSecurity.vbs` installed by the Content Engine setup program. This script updates an existing object store's security with users and groups as if they had been added when the object store was originally created. These users and groups can be given default permissions as end users or as object store administrators.

The `WizUpdateOSSecurity.vbs` script can be run through the security script wizard, or it can be run by itself. This script:

- Does not directly modify documents and custom objects. However, it does set permissions on Enterprise Manager's root folder. Thereafter, you can configure security parentage so that the root folder becomes the security parent of any folders, documents and custom objects that should inherit the new permissions. This applies the same effective security as if all these documents, custom objects, and contained folders had been directly modified. Keep in mind, however, the different behavior between directly applied security and inherited security.
- Does modify the security on all other securable objects.
- Does not remove or modify existing security permissions.

To update an object store with new users or groups

1. Right-click the object store node and run the Security Script Wizard.
2. When it asks you to select an XML security script information file, browse to and select `OSecurityUpdate.xml`.
3. When it asks you to define security roles, you should see two roles under Security Role: **Object Store Administrators** and **Object Store Users**. If you do not see these roles, you have not loaded `OSecurityUpdate.xml` properly.
4. Use the Add button to add security participants for the selected role. The Select Users and Groups dialog box will open. Click OK when you have added the participants for that particular role.
5. Click Finish when you are done. The wizard will proceed to apply the security permissions to the objects in the object store. This may take some time, depending on the number of objects that need to be updated. The wizard will inform you when the process of applying security is complete.
6. Examine the new permissions on Enterprise Manager's root folder. Depending on how you have configured the inheritance from the root folder and all generations of child folders, these new permissions may not yet have been inherited. You should configure the folder security parentage as appropriate.

Security example

This example describes a business process, analyzes the work performed by various users, and describes how to create groups, document classes, and folders to achieve appropriate security.

Imagine a credit card approval process where:

- A **clerk** adds new credit card applications to the object store. For each application, the clerk selects the document class CCAppls and files the new document in the /NewAppls folder. (The administrator has configured a subscription to launch a workflow for each new document in the CCAppls document class.) The workflow routes the application, as an attachment, to an applications processor.
- An **applications processor** reviews the application and approves it, denies it, or refers it to the analyst for further review. The decision is noted in the document properties. Depending on the decision, the workflow (using the processor's access rights) refiles the application in the appropriate /Approved, /Denied, or /Pending folder.
- The **analyst** investigates all applications filed in the /Pending folder. The analyst updates the application information as needed, approves or denies the application, and returns it to the approval representative.
- The **manager** writes and updates procedures from time to time and publishes them to the /Procedures folder. The manager spot-checks processed applications and keeps an eye on the work in the /NewAppls and /Pending folders.

Everyone must be familiar with department procedures.

The **system administrator** maintains workflows, publish templates, and document classes.

Analysis

The analysis requires these steps:

- Describe the work.
- Determine the access rights required for the work.
- List the data objects.

Describe the work

List the operations performed by the various users.

User/Function	Operation Performed
Clerks	Add applications to the /NewAppls folder
Processors	Participate in a workflow
	Display credit card applications
	Set status (a document property) to approved, denied, or pending
	Move documents from /NewAppls to /Approved, /Denied, or /Pending folder
Analyst	Participate in a workflow
	Display credit card applications
	Set status (a document property) to approved or denied
Manager	Add documents to /ProceduresSource folder
	Publish documents to /Procedures folder
	Display credit card applications in all folders
Administrator	Define and update workflows, publish templates, and document classes

Determine the access rights required for the work

List the access rights required for the folder and document operations.

Folder Operations	Users	Access Rights Required
Add document to the /NewAppls folder	Clerks	View Properties Add to Folder
View contents of /NewAppls folder	Processors	View Properties
Remove document from /NewAppls folder	Processors	View Properties Add to Folder

Add document to /Approved, /Denied, /Pending folders	Processors	View Properties Add to Folder
Add document to /ProceduresSource folder	Manager	View Properties Add to Folder
Publish document to /Procedures folder	Manager	View Properties Add to Folder
Display contents of /NewAppls and /Pending folders	Manager	View Properties

Document Operations	Users	Access Rights Required
Display applications	Processors Manager Analyst	View Properties View Content
Update application status (document property)	Processors	Modify Properties
Update application content	Analyst	Modify Properties Modify Content
Publish procedures (source) documents	Manager	View Properties View Content Modify Properties Publish
Use publish template to publish procedures	Manager	View Content
Define workflows, publish templates, and document classes	Administrator	None, the admin has Owner Control. Must belong to PWDesigner, if the group exists, to define workflows.
Display and print procedures	All users	View Properties View Content

List the data objects.

List the data objects that must be secured.

Document classes: CCAppls, CCProcedures

Folders: /NewAppls, /Approved, /Denied, /Pending, /ProceduresSource, /Procedures

Workflow: CCProcess

Publish Template: CCProceduresPublishing

Setup

To set up security for the credit card approval process, the administrator will:

- Create users and groups.
- Define document classes.
- Define a workflow.
- Define publish template.
- Create folders.

Create users and groups in the configured authentication provider's directory service.

To simplify maintenance, create a group for each function, even if only one person performs the function. You then update the group membership when job assignments change. For this example, we'll create the following groups:

CC_ApplsEntry
CC_Processing
CC_Analyst
CC_Manager.

Define document classes in FileNet Enterprise Manager.

Create the CCAppls document class. Display the Default Instance tab, and add the groups, with the access rights indicated in the table below. Optionally and alternatively, you may wish to associate the CCAppls document class with a security policy that will apply appropriate security to applications depending on the version status.

Group	Access Rights for Document Class: CCAppls
CC_ApplsEntry	View Properties
CC_Processors	Modify Properties
CC_Analyst	Modify Content
CC_Manager	View Properties

Create the CCProcedures document class. Display the Default Instance tab, and add the groups, with the access rights indicated in the table below. Optionally and alternatively, you can associate the CCProcedures document class with a security policy to apply security appropriate to the procedure's version status.

Group	Access Rights for Document Class: CCProcedures
CC_ApplsEntry	View Properties

CC_Processors	View Properties
CC_Analyst	View Properties
CC_Manager	Modify Properties Publish

Define the workflow using Process Designer.

Define the workflow used by applications processors and analysts.

Group	Access Rights for Workflow Definition: CCAppls
CC_Processors	View Properties
CC_Analyst	View Properties

Define the publish template using Publishing Designer.

Define the publish template used by the manager.

Group	Access Rights for Publish Template: CCProceduresPublishing
CC_Manager	View Content

Create folders using FileNet Enterprise Manager.

Create folders with the access rights indicated in the tables below.

Group	Access Rights for Folder: /NewAppls
CC_ApplsEntry	Add to Folder
CC_Processors	Add to Folder
CC_Analyst	View Properties
CC_Manager	View Properties

Group	Access Rights for Folders: / Approved and /Denied
CC_Processors	Add to Folder
CC_Manager	View Properties

Group	Access Rights for Folder: /Pending
CC_Processors	Add to Folder
CC_Analyst	View Properties
CC_Manager	View Properties

Group	Access Rights for Folder: /ProceduresSource
CC_Manager	Add to Folder

Group	Access Rights for Folder: /Procedures
CC_Manager	Add to Folder
All other groups	View Properties

Optionally, a folder can be the security parent for the contained objects (subfolder, documents, and custom objects). This requires configuration of the parent folder and each contained object.

Frequently asked questions about FileNet P8 Platform security

Questions

- Q: When you add non-administrative users while running the object store wizard, what default permissions do they get?
- Q: How is a document's security set?
- Q: Are all objects securable?
- Q: Can I have many instances of Enterprise Manager simultaneously running on different boxes under the same account, against the same object store?
- Q: Where does Content Engine store the security descriptors for its objects?

Answers

Q: When you add non-administrative users while running the object store wizard, what default permissions do they get?

A: On documents: View Content (on the class Default Instance Security ACL) plus Create Instance (on the class Security ACL).
On folders: Modify Properties.
On the object store: Use object store.

Q: How is a document's security set?

A: When first created, documents get permissions from the Default Instance Security ACL and default security policy of its class. They can also inherit permissions from a security parent, if configured. When documents go through versioning changes (by being checked out, checked in, promoted or demoted) their security can change if there is a properly configured security policy associated with the document.

Q: Are all objects securable?

A: Most objects that users and administrators work with are directly securable. If an object's property sheet contains a Security tab, then it is directly securable. However, some objects have the same security as some other (owner or container) object that they are dependent on. For example, once a choice list is added to a document, it has the same security as the document.

Q: Can I have many instances of Enterprise Manager simultaneously running on different boxes under the same account, against the same object store?

A: Yes. Enterprise Manager is fully MMC-compliant.

Q: Where does Content Engine store the security descriptors for its objects?

A: Content Engine's configured database has tables that contain all information about objects, properties, classes, etc, including the fully encrypted security descriptors for these objects.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

IBM	DB2	WebSphere
AIX	Tivoli	

FileNet is a registered trademark of FileNet Corporation, in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.



Program Number: 5724-R76, 5724-R81, 5724-R85, 5724-S19

Printed in USA

GC31-5524-00

