

NEXTLABS®



Control Center 8.7.2

Administrator's Guide

Revision 1

March 2019

Confidentiality notice

THIS DOCUMENT IS CONFIDENTIAL AND PROPRIETARY TO NEXTLABS, INC. AND MAY NOT BE REPRODUCED, PUBLISHED OR DISCLOSED TO OTHERS WITHOUT COMPANY AUTHORIZATION.

© 2009-2019 NextLabs, Inc. All rights reserved. The information in this document is subject to change without notice.

To provide feedback on this document, email the documentation team at techpubs@nextlabs.com.

Trademarks

NextLabs, the NextLabs Logo, Compliant Enterprise, the Compliant Enterprise Logo, Deep Event Inspection, 360 Degree Enforcement, and ACPL are trademarks or registered trademarks of NextLabs, Inc. in the United States. All other brands or product names used herein are trademarks or registered trademarks of their respective owners.

License agreement

This documentation and the software described in this document are furnished under a license agreement or nondisclosure agreement. The documentation and software may be used or copied only in accordance with the terms of those agreements. No part of this document may be reproduced, stored in a retrieval system or transmitted in any form or by any means, either electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's use, without the prior written permission of NextLabs, Inc.

The content of this document is provided for informational and instructional use only. It is subject to change without notice, and should not be construed as a commitment by NextLabs, Inc.

NextLabs, Inc. assumes no responsibility or liability for any inaccuracies or technical errors that may appear in the content of this document.

Published in San Mateo, CA, by NextLabs, Inc.

www.nextlabs.com

info@nextlabs.com

support@nextlabs.com

650.577.9101

Contents

Confidentiality notice.....	iii
Chapter 1: Introduction.....	10
Configure Control Center.....	11
About the console.....	11
About the Status section.....	13
Additional resources.....	14
Contacting Support.....	14
Chapter 2: About the Dashboard.....	16
Understand the Dashboard data.....	17
Chapter 3: Manage access to the console.....	20
About delegation policies.....	21
Adding and editing delegation policies.....	21
Deleting delegation policies.....	26
About managing user accounts.....	26
Adding or editing user accounts in the Control Center console.....	26
Deleting user accounts.....	27
Defining the number of unsuccessful login attempts before a user account gets locked....	27
Unlocking a user account.....	28
Resetting user password.....	28
Preventing users from using the same password from a number of past passwords.....	28
Configuring mail server settings after installing Control Center.....	29
Recovering your username or password.....	30
About importing user information from AD sources.....	31
Configuring AD sources.....	31
Adding users from an AD source.....	32
About importing user information from Azure AD.....	33
Configuring the connection between the Control Center and Azure AD.....	33
Adding users from Azure AD.....	34
Changing the superuser password.....	34
Chapter 4: Enrolling users, hosts, and groups.....	36
About enrollment.....	37
About the Information Network Directory.....	37
Entities that can be enrolled.....	37
Required entities.....	37
Optional entities.....	38
Enrolling entities for Windows and Linux systems.....	38
About Domain Group enrollment.....	39
Enrollment failures.....	39
Overview of enrollment types and procedures.....	40
About the Enrollment Manager utility.....	40
How enrollment connections are established.....	40
Before you enroll.....	43
Using the Enrollment Manager.....	44

About enrollment input files.....	47
About connection files.....	48
Definition files.....	54
Filter files.....	56
Configuration files for Domain Groups.....	58
Enrolling users and hosts.....	61
Configure a secure LDAP connection.....	61
How to enroll from a directory (single domain).....	63
Enrolling from an LDIF file for a single domain.....	64
Enrolling Domain Groups.....	65
How to maintain enrollments.....	67
Updating vs. synchronizing.....	67
Updating directory enrollments for a single domain.....	69
Updating LDIF file enrollments for a single domain.....	70
Updating Domain Group enrollments.....	71
Removing deleted LDAP users or groups from an enrollment.....	72
Synchronizing enrollments.....	73
Managing enrollment tasks.....	77
Checking enrollments.....	78
Deleting single domain enrollments.....	79
Delete Domain Group enrollments.....	79
Accessing enrollment help (usage).....	80
Chapter 5: Enrolling other entities.....	82
Enrolling applications.....	83
The definition file.....	84
Multiple application versions.....	85
Enrolling file shares.....	86
About Resource Path Discovery.....	86
Enrolling location sites.....	89
Enrolling SharePoint groups.....	91
Enrollment utility password requirements.....	97
Chapter 6: Defining custom properties.....	98
User, hosts, and application properties.....	99
About Property Manager.....	100
Adding properties.....	101
Deleting properties.....	103
Listing properties.....	103
Portal content properties.....	103
Defining custom portal properties.....	104
Chapter 7: Managing policy enforcers.....	106
About policy enforcers.....	107
Adding policy enforcers.....	107
About ICENet servers and policy enforcer profiles.....	107
Initial registration.....	107
Relationships of profiles ICENet servers, and hosts.....	108
Load-balanced or clustered ICENet servers.....	109
Moving profiles between ICENet servers.....	109
Load balancing.....	109
Setting up load balancing after Control Center installation.....	110
Stopping policy enforcers.....	111

Stopping desktop enforcers.....	111
Stopping file server enforcers and SharePoint enforcers.....	111
Confirming policy enforcer status.....	112
Restarting policy enforcers.....	112
Service user account permissions.....	113
Uninstalling policy enforcers.....	113
Reconfiguring policy enforcers.....	114
Viewing policy enforcer status.....	114
Managing policy enforcer profiles.....	116
Default profiles and passwords.....	117
Defining policy enforcer profiles.....	117
About push deployment.....	119
Removing a host from a policy enforcer profile.....	120
Modifying policy enforcer profiles.....	120
Deleting a policy enforcer profile.....	120
Database management.....	121
Configuring data access.....	121
About the four databases.....	122
Database management operations.....	125
Best practices.....	130
Other database tools.....	132
Configuration tools.....	134
About configuration files.....	134
Editing configuration files.....	136
Configuration settings.....	136

Chapter 8: Policy and Policy Model overview..... 158

About policies.....	159
Overview of policy implementation.....	159
Managing Policy Model resource types.....	160
Adding and editing subject types.....	161
Adding and editing resource types.....	163
Managing policy components.....	167
About component types.....	168
Adding and editing policy components.....	169

Chapter 9: Constructing and testing policies..... 174

Constructing policies.....	175
Adding policies.....	175
Using Advanced Conditions.....	181
Using wildcards in Advanced Conditions.....	182
Example condition for equal and multi-value equal.....	183
Example condition for multi-value equals_unordered operators.....	183
Example condition for the includes operator.....	184
Example condition for greater than/less than.....	184
Managing subpolicies.....	184
Examples of subpolicies.....	184
Adding subpolicies.....	185
Viewing and editing policy and component information.....	185
Viewing and editing policy and component hierarchy.....	185
Viewing policy and component version history.....	185
Viewing and editing policy and component properties.....	185
Cloning policies and components.....	186
Cloning policies.....	186

Cloning components.....	186
Using audit policies.....	187
Creating audit policies.....	187
Allow policy obligations.....	187
About enforcement logging.....	187
Logging vs. email.....	187
Constructing Delegated Administration policies.....	188
Testing policies.....	188
Chapter 10: Deploying and managing objects.....	194
About deployment.....	195
Deploying components.....	195
Deploying policies.....	195
Managing policies and components.....	195
Searching and filtering policy and component lists.....	196
Viewing policy version information.....	197
Modifying policies and components.....	197
Changing user access to components.....	197
De-activating policies and components.....	197
Deleting objects.....	198
Chapter 11: Exporting and Importing Policies.....	200
About exporting and importing policies and other items.....	201
Exporting and importing objects using the Control Center interface.....	201
Exporting and importing policies securely.....	202
Securely exporting and importing objects using the Control Center interface.....	203
Manually configuring properties for exporting or importing encrypted policy bundles....	204
Importing the digital signature certificate for importing encrypted policy bundles.....	205
Using versions.....	205
About object states.....	206
Chapter 12: Managing Reports.....	208
Introducing Reporter.....	209
About Reporter.....	209
Accessing the Reporter console.....	209
About the Reporter console.....	209
About the Reporter console.....	210
Dashboard tab.....	211
Reports tab.....	211
Monitoring tab.....	211
Banner.....	211
Using the Reporter Dashboard.....	212
Introducing the Reporter Dashboard.....	212
About Reporter Dashboard data.....	213
Working with reports.....	215
Reporting functionality and permissions.....	215
Creating a report.....	217
Running a report.....	225
Viewing report output.....	227
Saving reports.....	228
Sample reports.....	230
Report anomalies.....	235
Working with monitors and alerts.....	236

Monitoring functionality and permissions.....	236
Creating a monitor.....	237
Deactivating and deleting monitors.....	240
Viewing alerts.....	241
Analyzing alerts.....	243
Using Audit Logs.....	246
Sample reports.....	249
Chapter 13: Overview of PEP development.....	256
About PEPs and interactions with PDPs.....	257
Types of PDPs.....	257
Using embedded PDPs.....	258
Applications to be enforced and available SDKs.....	258
PEP development process.....	259
Policy Controller components.....	260
Chapter 14: PEP client SDK (OpenAz API).....	262
About the OpenAz PEP client SDK.....	263
Configuring a development environment.....	263
Configuring one-way and two-way SSL authentication between PDP and PEP.....	263
Configuring one-way SSL for Java Policy Controller.....	264
Configuring two-way SSL for Java Policy Controller.....	265
Using the Java client library.....	266
Setting up the Java SDK.....	266
Invoking the PDP.....	266
Making authorization requests using Java.....	272
Exception handling.....	273
SDK quick reference.....	273
Using the JavaScript client library.....	280
Setting up the JavaScript SDK.....	280
Required properties.....	281
Making authorization requests using JavaScript.....	281
Configuring user authentication for the REST API.....	283
REST API for external PDPs.....	286
About the REST API.....	286
How to set up the REST API.....	286
How to access the REST API.....	287
Types of requests.....	288
Request format.....	289
Examples of request data.....	289
NextLabs-specific environment attributes.....	295
Overview of plug-ins.....	297
About Control Center plug-ins.....	297
About Policy Controller plug-ins.....	298
Using attribute provider plug-ins.....	299
Using heartbeat plug-ins.....	302
Report log views.....	314
Using the sample enforcer.....	319
Installing the Policy Adapter SDK.....	321
C and C++ API reference.....	328
Chapter 15: Appendix.....	348
FAQs and troubleshooting.....	349

General FAQs.....	349
Troubleshooting FAQs.....	349
Changing memory settings.....	349
Restarting the Control Center Policy Server.....	350
Disabling the resource type checking in policy evaluations.....	350
Securing the JDBC connection between Control Center and Oracle Database.....	351
Extending datatypes in Oracle 12c.....	351
Password and users.....	352
Enforcer Profile Security Passwords.....	352
Utility Security Password.....	353
Database Password.....	353
SSL certificates.....	354
Overview of communication process.....	355
Using Java keytool to manage certificates.....	356
Replacing Control Center certificates.....	357
Replacing Policy Controller Certificates.....	376
Configuring communication between Control Center and Policy Controller when using middleware plug-ins.....	376
Glossary.....	378
Index.....	384

Chapter

1

Introduction

Topics:

- [Configure Control Center](#)
- [About the console](#)
- [About the Status section](#)
- [Additional resources](#)
- [Contacting Support](#)

This guide explains how to configure and use NextLabs® Control Center after initial setup is complete.

It also explains how to create, deploy, and manage the policies that control access to information and resources, and how to use reports, monitors, and alerts to review policy enforcement.

For information about installing or upgrading Control Center, see NextLabs Control Center Installation and Upgrade Guide which is available on the NextLabs Customer Portal at <https://customer.nextlabs.com/>.

Configure Control Center

To use NextLabs Control Center to enforce policies, the system must be configured for your environment, and policies must be designed and deployed.

Tasks include:

- Install or upgrade Control Center: For instructions, see [NextLabs Control Center Installation and Upgrade Guide](#).
- Identify policy requirements: Identify the kind of resources and information you need to protect, the subjects or users who access those resources, and the policies you want to enforce in your environment.
- Enroll users, hosts, groups, and other entities: Enrollment is the process of importing LDAP directory entities, Microsoft® SharePoint® user groups, hosts, and other entities into the Control Center database. This makes the information available for use by NextLabs components.
- Configure permissions for Control Center system users: Create access rules or permission sets based on user attributes and conditions.
- Set up a Policy Model: A Policy Model is one or more Resource Types that serve as templates for policy components. Resource Types establish the attributes, actions, and obligations available to policy components, such as resource, action, and subject components..
- Configure policy objects, and attributes: Policy objects, such as subject, action and resource components, must be configured before users can create policies.
- Write, test, and deploy policies: After a Policy Model and policy objects are defined, the policies themselves can be written, tested, and deployed.



Note: Policies that have been deployed are enforced only after policy enforcers have been installed and configured.

- Optional: Install and configure PEPs for additional applications: Developers can create PEPs (policy enforcement points) to enforce policies on applications that do not have predefined NextLabs enforcers.

Related concepts

[Policy and Policy Model overview](#) on page 158

[Manage access to the console](#) on page 20

[PEP development process](#) on page 259

Building PEPs involves these steps.

About the console

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

Interface components are described in the following table.

Table 1: Control Center interface overview

Interface section	Description
Dashboard	Summary information related to Control Center components. The data displayed in all panes of the dashboard is refreshed from the Activity Journal each time you refresh the Dashboard page. This means that data is updated on demand. If a pane shows a statistic for the past week, that reflects not the last seven whole calendar days, but the last seven 24-hour periods starting from the top of the current hour.
Policies	<p>Constructing policies refers to the process of building information control policies in Control Center. Policies use components as building blocks to represent rules that control information access and use in your organization.</p> <p>Deploying policies means the distribution of new or modified policies and policy components to the appropriate enforcement points on desktop PCs, laptops, and file servers throughout the organization. This means you can create, review and refine policies as long as you like, but they are not enforced until they are deployed and the appropriate enforcers are configured.</p>
Components	Components can be thought of as the parts of speech used to construct policy statements. For example, “noun” policy components might include All employees in the human resources department or Any file with an .xls extension, and “verb” components might include Copy, Print, and Rename File.
Policy Model	The Policy Model is an abstract model that represents the various parts of the enterprise environment, such as users, applications, documents, and desktop PCs. The actual work of modeling consists of defining policy components. A component is a named definition that represents a category or class of entities, such as users, data resources, or applications; or of actions, such as Open or Copy.
Policies	The interface where you create and manage policies.
Reports	The interface for creating reports and performing forensics, and it is the front end of the Report Server.

Interface section	Description
Administration	<p>The interface that enables administrators to configure and monitor the Control Center system.</p> <p>The Administrator interface includes the following sections:</p> <ul style="list-style-type: none"> • System Settings: The interface enables administrators to view the server and policy enforcer status and manage policy enforcer profiles. • Delegation Policies: The interface that enables administrators to manage user access to the Control Center interfaces as well as policies, and policy objects using rules based on user attributes and conditions. Using attributes and conditions mirrors the way policies are used to control access to resources and eliminates the need to create and apply roles to users or groups of users. • Users: The interface that enables Administrators to create and edit users. • User Source: The interface to configure the connection between the Control Center and the AD server or Azure Active Directory (Azure AD)
Tools	<p>The Tools interface contains the Policy Tester that enables you to test policies after they are saved. Testing policies in the console simulates sending authorization requests and enables you to view responses without building PEPs (policy enforcement points) or setting up application interactions.</p>

Related concepts

[About the Dashboard](#) on page 16

[About the Status section](#) on page 13

The information available on the Status section is designed to help you monitor current system activity.

[Manage access to the console](#) on page 20

[Default profiles and passwords](#) on page 117

If enforcer profiles are not explicitly defined, all enforcers in the system are automatically assigned the settings of the default profile that is automatically created in the Administrator console.

Related information

[Managing policy enforcers](#) on page 106

About the Status section

The information available on the Status section is designed to help you monitor current system activity.

It is organized into the following screens, which you access by clicking the sub-tabs:

- Status Overview: Displays a diagnostic overview that informs you of potential problems at a glance, statistical totals for various types of activity throughout Control Center, and the current status of each Control Center component. This screen displays in front by default.
- Policy Enforcer Status: Displays the current status of each host where policy enforcer software is installed, and shows which policy enforcer profile is assigned to each host.

Related reference

[About the console](#) on page 11

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

Additional resources

For additional documentation and self-help resources go to the NextLabs Customer Portal at <https://customer.nextlabs.com>.

Contacting Support

For help with NextLabs products, email Technical Support at <https://support.nextlabs.com>.

Chapter

2

About the Dashboard

Topics:

- [Understand the Dashboard data](#)

You access the Dashboard by clicking **Dashboard** on the left navigation bar.

The Dashboard displays a predefined statistical view of data that provide a snapshot of policy enforcement trends. These data include:

- Most commonly denied requests for access to data resources, by resource, user, and policy
- Most commonly allowed request for access to resources, by resource
- 30-day trend for the enforcement of all Deny policies, for all users and resources
- Detailed information about the most recent ten instances of any policy either allowing or denying a user access to any resource
- Statistics on alerts in the current week
- Detailed information about alerts raised in the current day

Data is refreshed from the Activity Journal each time you open the Dashboard. This means that data is updated on demand; for example, if a pane shows some statistic for the past week, that reflects not the last seven whole calendar days, but the last seven 24-hour periods starting from the top of the current hour.

Related reference

[About the console](#) on page 11

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

Understand the Dashboard data

This table summarizes the data presented in each of the predefined queries represented on the Dashboard, and discusses the possible implications of and uses of the data displayed in each.



Note: The system displays No data available if policies have not been deployed.

Table 2: Console Dashboard data contents

Graph	Description	Action required
Total Policies	The number of policies that have been added or edited in the past week.	Review the total number of policies to identify anomalies.
Deployed	The number of policies that have been deployed to policy enforcers in the past week.	Review the total number of policies to identify anomalies.
De-activated	The number of policies that have been deactivated in the past week.	Review the total number of policies to identify anomalies.
Draft Policies	Policies that are in the Draft state. Draft policies are policies that have been added but have never been deployed.	Complete policies that are required and delete drafts that are no longer needed.
Recent activities	A list of actions that have been performed on the console in the past week.	Review activities to identify anomalies. Identify improperly designed resource component or policies, which allow inappropriate users access to sensitive resource.
System configuration status	The current status of PDPs (Policy Decision Points) and IceNets. The information presented here is updated periodically: <ul style="list-style-type: none">• Active PDPs: PDPs that have been active in the past week• Heartbeat failed: The number of heartbeat failures in the past week• PDPs up to date: The number of PDPs that are up to date• PDPs not up to date: The number of PDPs that are out of date• Active IceNets: IceNets that have been active in the past week.• Heartbeat failed: The number of heartbeat failures in the past week	Review the status and take action to resolve issues with PDPs and IceNets.

Graph	Description	Action required
System details	The system displays the following details: <ul style="list-style-type: none"> • The number of days remaining in license expiry • The Control Center server host operating system version • The Control Center version installed • The number of audit records in the past week 	If your license is expiring soon or has expired, contact NextLabs Support at https://support.nextlabs.com .
PDP throughput	The number of requests sent to PDPs in the past day.	Investigate high numbers of requests that might indicate performance issues.
Top 10 activities by user	The number of users that have triggered Allow or Deny activities in the past week.	Identify incorrectly defined entitlements: users or group should have access, but policies are not updated or are incorrectly designed.
Top 10 activities by resource	Bar chart representing the five resources that any users have most frequently attempted to access and been allowed or denied by an active policy, in the past week.	Identify resources of interest to users who should not be using them. Identify incorrectly designed resource or user component, blocking users who should have access.
Frequently evaluated policies		Educate users about access policies and individual/group entitlements, at a broad level. Identify improperly designed policies that are blocking too many users who expect and are entitled to access or use.
Activity Alerts	The number of alerts that have been triggered by policies in the past week. Alerts are configured in the Reports section.	Identify policies that are being enforced at a high rate. Further review or action may be required.
Top 10 policies grouped by tags	A graphical representation of the tags related to the top ten policies that have been enforced in the past week.	Review policies being watched by monitors that are tagged, and are being enforced at a high rate.
Policies not matched in any request	A list of policies that have not been evaluated in response to any user actions.	Review policies to ensure they are relevant and constructed properly.
Enrollement	A list of enrollments, with the following information: <ul style="list-style-type: none"> • Domain name • Status • Last sync • Type 	

Chapter

3

Manage access to the console

Topics:

- [About delegation policies](#)
- [Adding and editing delegation policies](#)
- [Deleting delegation policies](#)
- [About managing user accounts](#)
- [About importing user information from AD sources](#)
- [About importing user information from Azure AD](#)
- [Changing the superuser password](#)

This section explains how to manage user access to Control Center interface elements through delegation policies, which control access using attributes.

In addition, this section explains how to manage Control Center user accounts manually, and how to change the superuser password.

Related concepts

[Constructing Delegated Administration policies](#) on page 188

Delegated Administration policies enables administrators to manage user access to the Control Center interfaces as well as policies, and policy objects using rules based on user attributes and conditions.

[Changing user access to components](#) on page 197

User access to components, policies, and other objects is determined by delegation policies.

[About Reporter](#) on page 209

Reporter is a web-based interface that is installed as a component of the console.

Related reference

[About the console](#) on page 11

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

About delegation policies

Delegation policies enable administrators to control user access to the Control Center user interface including policies, policy objects, and system settings, by creating policies based on user attributes and conditions. Using attributes mirrors the way policies are used to control access to resources and eliminates the need to create and apply roles to users or groups of users.

For example, Control Center users might have the following citizenship attributes:

- US
- Canada
- South Africa, California resident

An administrator could create the following delegation policies based on these attributes:

- US Citizens can manage policies tagged HIPPA or IPPA
- US Citizens can view policies tagged CMIA
- California residents can manage policies tagged CMIA
- Canadian Citizens can manage policies tagged NAPRA
- Canadian citizens cannot manage policies tagged HIPPA

When authorization access requests are received, the system evaluates the requests against the policies and responds accordingly.

Related concepts

[Reporting functionality and permissions](#) on page 215

Adding and editing delegation policies

Creating and editing delegation policies is similar to creating and editing the policies that govern access to resources.

Administrators specify the required attributes, conditions, tags, and obligations for each delegation policy. Once created, delegation policies are used to determine console access for users who have Control Center user accounts.

- Decide what kind of policy you want to create, and whether to base the policy on user attributes such as locations or usernames.
- Know the attributes or usernames you want to use in the policy.
- Know the relevant tags applied to policy components.

1. Log in to the Control Center web console with an account that has permission to manage delegation policies, such as the superuser Administrator account.
2. On the left navigation bar, select **Administration > Delegation Policies**.
3. Do one of the following:
 - Click **Add Delegation Policy** in the upper right.
 - Click the name of an existing policy.
4. Add or edit the policy properties described in the following table.

Table 3: Delegation policy properties

Option	Description
BASIC INFORMATION	The name and description of the policy. The name appears on the Delegation Policies list.

Option	Description
Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Description	A description of the item. This field is useful for explaining how the item is used and for documenting changes to the item. Descriptions can include multibyte characters, such as those used by Asian languages.
EFFECT & CONDITIONS	The policy actions and attributes that identify users.
Allow or Deny	<p>The action taken when conditions in the policy are met.</p> <ul style="list-style-type: none"> Allow: Permit the listed Subjects to perform the task specified in the policy. Allowing a set of Subjects to perform a task does not mean that others are blocked from performing that action. Allow policies can never result in Deny responses. Allow policies can only result in Allow or Not Applicable responses. Deny: Do not permit the listed Subjects to perform the task specified in the policy, but allow all others to do so. Deny policies can result in Deny, Allow, or Not Applicable responses.
Conditions	User attributes used in the policy definition. User attributes include location, AD (Active Directory) Group, and Department. Select the appropriate operator (is, to include users that have the specified attribute, or is not to exclude users with the specified attribute). Then specify the attribute definition, such as USA for the Location attribute. Click Add condition to add an attribute.
MODULES	<p>The Control Center interfaces where users perform tasks. Use tags and conditional settings to further refine access to tasks. If no tags or conditions are specified, users are able to access all items.</p> <p>When you select an action check box, all required and optional action check boxes are automatically selected across modules. The following table lists the actions and the corresponding required and optional actions. You can review each module and choose to clear any action check box that is optional.</p>

Option	Description
Policy	<p>Controls access to policy-management tasks such as:</p> <ul style="list-style-type: none"> • Create Policy: Set up new policies. • Deploy Policy: Make policies available to policy enforcers in the system. • Delete Policy: Deactivate policies. • View Policy: View the properties of existing policies. • Edit Policy: Change the properties of existing policies.
Component	<p>Controls access to policy-component-related tasks, such as:</p> <ul style="list-style-type: none"> • Create Component: Set up new policy components. • Deploy Component: Make policy components available for use in policies. • View Component: View the properties of existing policy components. • Delete Component: Remove policy components from the system. • Edit Component: Change the properties of existing policy components. <p> Note: If an administrator applies a tag-based restriction to a user, that user cannot create new components or sub-components in the Policies or Component details page.</p>
Policy Model	<p>Controls access to policy-model-related tasks, such as:</p> <ul style="list-style-type: none"> • Create Policy Model: Set up new Policy Models. • Delete Policy Model: Remove Policy Models from the system. • View Policy Model: View the properties of existing Policy Models. • Edit Policy Model: Change the properties of existing Policy Models.

Option	Description
Others	<ul style="list-style-type: none"> • Delegated Administration: Manage delegated policies and users. Tasks related to managing policies that govern user access to Control Center interfaces. Users access these tasks by clicking Administration > Delegation Policies in the Control Center left navigation bar. • Reports: Report-related tasks performed in the Reporter section of the Control Center interface. Users with permission to create, delete, view, or edit reporter actions access these tasks by clicking Reports in the Control Center left navigation bar. • System Settings: Access to manage system settings. • Tag Management: Access to create policy, policy model, and component tags. • Tools: Access to test policies within the Control Center console using Policy Tester. <p>Controls access to administrative tasks, such as:</p>

Table 4: Required and optional actions

Action	Required actions	Optional actions
Create Policy	<ul style="list-style-type: none"> • View Policy 	<ul style="list-style-type: none"> • Edit Policy • Create Policy Tags
Delete Policy	<ul style="list-style-type: none"> • Deploy Policy • View Policy 	Not applicable
Deploy Policy	<ul style="list-style-type: none"> • View Policy 	<ul style="list-style-type: none"> • Deploy Component
Edit Policy	<ul style="list-style-type: none"> • View Policy 	<ul style="list-style-type: none"> • Create Policy Tags
View Policy	Not applicable	Not applicable
Create Component	<ul style="list-style-type: none"> • View Component 	<ul style="list-style-type: none"> • Edit Component • Create Component Tags
Delete Component	<ul style="list-style-type: none"> • Deploy Policy • View Component 	<ul style="list-style-type: none"> • Edit Policy
Deploy Component	<ul style="list-style-type: none"> • View Component 	Not applicable
Edit Component	<ul style="list-style-type: none"> • View Component 	<ul style="list-style-type: none"> • Create Component Tags
View Component	Not applicable	Not applicable
Create Policy Model	<ul style="list-style-type: none"> • View Policy Model 	<ul style="list-style-type: none"> • Edit Policy Model • Create Policy Model Tags
Delete Policy Model	<ul style="list-style-type: none"> • View Policy Model 	Not applicable

Action	Required actions	Optional actions
Edit Policy Model	<ul style="list-style-type: none"> View Policy Model 	<ul style="list-style-type: none"> Create Policy Model Tags
View Policy Model	Not applicable	Not applicable
Manage Delegation Policies	Not applicable	Not applicable
Manage Users	Not applicable	Not applicable
Manage Reports	<ul style="list-style-type: none"> View Reports 	Not applicable
View Reports	Not applicable	Not applicable
Manage System Settings	Not applicable	Not applicable
Create Component Tags	<ul style="list-style-type: none"> View Component At least one of the following actions: <ul style="list-style-type: none"> Create Component Edit Component Manage Delegation Policies 	Not applicable
Create Policy Model Tags	<ul style="list-style-type: none"> View Policy Model At least one of the following actions: <ul style="list-style-type: none"> Create Policy Model Edit Policy Model Manage Delegation Policies 	Not applicable
Create Policy Tags	<ul style="list-style-type: none"> View Policy At least one of the following actions: <ul style="list-style-type: none"> Create Policy Edit Policy Manage Delegation Policies 	Not applicable
Policy Tester	Not applicable	<ul style="list-style-type: none"> View Policy View Component

5. Click **Save**. The policy is used to determine console access for users who have Control Center user accounts.

Add accounts for the Control Center users to be governed by the policy.

Related concepts

[About managing user accounts](#) on page 26

Control Center user accounts enable users to log in to the Control Center web console using a specified username and password.

[About Reporter](#) on page 209

Reporter is a web-based interface that is installed as a component of the console.

Deleting delegation policies

Deleting delegation policies removes them from the Control Center system.

1. Log in to Control Center web console with an account that has permission to manage delegation policies, such as the superuser Administrator account.
2. On the left navigation bar, select **Administration > Delegation Policies**.
3. Select the policies you want to delete, then click the trash can button.

About managing user accounts

Control Center user accounts enable users to log in to the Control Center web console using a specified username and password.

You can add users from Active Directory (AD) sources or add, edit or unlock user accounts manually. You can also unlock user account.

-  **Important:** Delegation policies determine the tasks users can perform when they are logged in to Control Center. If delegation policies prohibit users from performing any tasks in the Control Center interface, users can log in using Control Center user accounts, but their activities are limited to viewing the static dashboard and the Getting Started page.

You can set additional security measures for user passwords: locking a user account after a number of unsuccessful login attempts, and preventing users from using the same password within a certain number of past passwords.

Related tasks

[Adding and editing delegation policies](#) on page 21

Creating and editing delegation policies is similar to creating and editing the policies that govern access to resources.

[Setting up user authentication for the REST API](#) on page 154

To use the REST API with Java Policy Controllers, you need to set up user authentication.

Adding or editing user accounts in the Control Center console

Administrators with permission to manage administrator actions, including the Create Administrator and Edit user tasks, can add or edit Control Center user accounts as needed.

1. Log in to the Control Center web console with an account that has permission to Create and Edit Administrators in the Administrator section of Control Center.
2. On the left navigation bar, select **Administration > Users**.
3. Perform one of the following steps:
 - Click **Add User**.
 - Click the name of an existing user.
4. Add or edit the user account properties:

Option	Description
USER INFORMATION	The name and description of the policy. The name appears on the Delegation Policies list.
First Name	The first name of the user.
Last Name	The surname of the user.

Option	Description
Username	The name the user enters to log in to the Control Center web console. This name cannot be modified after it has been created.
Email	The email address of the user.
Password and Confirm Password	The password the user enters to log in to the Control Center web console. Passwords must be between 7 and 12 characters, and they must contain at least one number, one character, and one non-alphanumeric character other than '_.
USER ATTRIBUTES	User characteristics that can be referenced in delegation policies.

5. Click **Save**. The account is added or updated, and users can log in with the specified credentials immediately.
6. Notify the user that their account has been added or updated, and supply the initial login credentials.

Deleting user accounts

Administrators with permission to manage Administrator Actions, including the Delete Administrator task, can delete Control Center user accounts as needed. Also, users with Manage Users permission can delete Control Center user accounts.

1. Log in to the Control Center web console with an account that has permission to delete Administrators in the Administrator section of the Control Center interface.
2. On the left navigation bar, select **Administration > Users**.
3. Select the user account to be deleted, then click the trash can button. The user account is removed from the Control Center system.

Defining the number of unsuccessful login attempts before a user account gets locked

To prevent the try-and-error password guessing, you can lock a user out of their account after a number of unsuccessful login attempts.

When the user is locked out, the user will not be able to log in even with correct credentials, until the system administrator manually unlocks the user account. You can define the number of times the user can try to log in before the account gets locked.

1. In the Control Center host server, stop the Control Center Policy Server service.
2. For both Windows and Linux, navigate to: <installation folder>/PolicyServer/server/configuration/
3. Using any text editor, open the `cas.properties` file.
4. Search for the parameter `failed.login.attempts`.
5. Change the parameter value to the number of times you want to allow the user to log in before the account gets locked.



Note: The default value is 5. If you set the value to 0, this function is disabled.

```
# Number of failed login attempt which will cause the user account to be locked
failed.login.attempts=5
```

Figure 1: failed.login.attempts parameter

6. Save and close the file.
7. Restart the Control Center Policy Server service.

Unlocking a user account

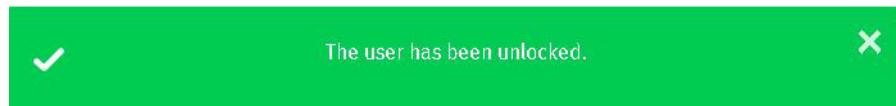
When a user gets locked out of their account, only the system administrator can unlock the account.

For example, if the number of unsuccessful attempts is set to 5, even if the user gets the password correct on the sixth try, the account will still be locked. In the event of a user lockout, the user needs to request that the system administrator unlock his account.

1. Log in to the Control Center with an account that has permission to manage users, such as the superuser Administrator account.
2. In the Control Center Console, navigate to **Administration > Users > <name of user>**.
- 3.



In the row corresponding to the user's name, click the padlock button to unlock the user account. An unlock confirmation message appears.



4. Inform the user that the account has been unlocked.

Resetting user password

Superuser administrator or a user who has permission to manage users can create, edit, or reset a user's password.

1. Log in to the Control Center web console.
2. On the navigation bar, select **Administration > Users**.
3. Click the name of the user whose password you want to reset.
4. In the Password box, type the new password.
5. In the Confirm password box, retype the new password.
6. Click **SAVE**.

A confirmation box appears.

Are you sure you want to reset the user's password?

7. Click one of the following buttons:

- **YES:** The user's password is reset.
- **NO:** The user's password remains unchanged.

Preventing users from using the same password from a number of past passwords

You can prevent users from using the same passwords when they are asked to change their passwords.

You can set the password history to the number of past passwords you want to stop the user from using again.

1. Stop the Control Center Policy Server service.
2. In the Control Center host server, for both Windows and Linux, navigate to: <installation folder>/PolicyServer/server/configuration
3. Using any text editor, open the `cc-console-app.properties` file.
4. Search for the `enforce.password.history` parameter:

```
# Application user account password history enforcement
# Default: 5, 0: Disabled
enforce.password.history=5
```

5. Change the parameter value to the number of past passwords you want to stop the user from using again.
6. Using any text editor, open the `server.xml` file.
7. Search for the `EnforcePasswordHistory` parameter:

```
<!--[ADMIN_COMPONENT_BEGIN]-->
<Context path="/administrator" reloadable="false" docBase="${catalina.home}/..apps/mgmtConsole.war" workDir="${catalina.home}/work/administrator">
  <Parameter name="ComponentName" value="pf-win4.gmf1.galab01.nextlabs.com_mgmt"/>
  <Parameter name="Location" value="https://pf-win4.gmf1.galab01.nextlabs.com:443/mgmt"/>
  <Parameter name="DNSLocation" value="https://pf-win4.gmf1.galab01.nextlabs.com:8443/dns"/>
  <Parameter name="InstallHome" value="install home:/>
  <Parameter name="EnforcePasswordHistory" value="5"/>
</Context>
<!--[ADMIN_COMPONENT_END]-->

<!--[REPORTER_COMPONENT_BEGIN]-->
<Context path="/reporter" reloadable="false" docBase="${catalina.home}/..apps/inquiryCenter.var" workDir="${catalina.home}/work/reporter">
  <Parameter name="DACLocation" value="https://pf-win4.gmf1.galab01.nextlabs.com:8443/dac"/>
  <Parameter name="DNSLocation" value="https://pf-win4.gmf1.galab01.nextlabs.com:8443/dns"/>
  <Parameter name="ComponentName" value="pf-win4.gmf1.galab01.nextlabs.com_reporter"/>
  <Parameter name="Location" value="https://pf-win4.gmf1.galab01.nextlabs.com:443/reporter"/>
  <Parameter name="InstallHome" value="install home:/>
  <Parameter name="EnforcePasswordHistory" value="5"/>
</Context>
<!--[REPORTER_COMPONENT_END]-->
```

8. Change the parameter value to the number of past passwords you want to stop the user from using again.

 **Note:** The default value is 5. If you set the value to 0, this function is disabled. Make sure that the value for the `EnforcePasswordHistory` parameter in step 4 and `EnforcePasswordHistory` parameter (in the admin and the reporter component sections) is the same.

9. Save and close the file.

10. Restart the Control Center Policy Server service.

Related tasks

[Restarting the Control Center Policy Server](#) on page 350

Whether running on a single host, on distributed hosts, or in a failover cluster, the Control Center Policy Server service must be manually started after the initial installation.

Configuring mail server settings after installing Control Center

You can edit the `cas.properties` file if you have not configured your mail server settings during Control Center installation, or if you want to change the existing mail server settings.

1. In the Control Center host server, stop the Control Center Policy Server service.
2. Navigate to the following location.
`<installation folder>/PolicyServer/server/configuration/`
3. Using any text editor, open the `cas.properties` file.
4. Search for the following section.

SMTP settings for reset password email

Example SMTP settings section.

```
# SMTP settings for reset password email
smtp.host=smtp.gmail.com
smtp.port=465
smtp.username=dummy.acc.test123@gmail.com
smtp.password=sa549f6ba05c840e5f43ef63e06a8ae1a
smtp.auth=true
smtp.starttls.enable=true
smtp.sender=dummy.acc.test123@gmail.com
```

5. Specify the following information.

Table 5: SMTP configuration

Field	Description
smtp.host	The name of your POP2 mail server host.
smtp.port	The SMTP port number. Use the default, port 25, unless you know it is different in your system.
smtp.username	The user name of the user authorized to connect to the mail server.
smtp.password	The password of the authorized user.
smtp.auth	Indicates whether to enable SMTP authentication. Set the value of this field to <code>true</code> .
smtp.starttls.enable	Indicates whether to turn an existing insecure connection into a secure one. Set the value of this field to: <ul style="list-style-type: none">• <code>true</code>: if you are using a secure smtp.port.• <code>false</code>: if you are using the non-secure default smtp.port.
smtp.sender	The email address to appear in the From field of email notifications sent out by policy enforcement. This must be a valid administrator's address so that recipients can respond if necessary.

6. Save and close the `cas.properties` file.
7. Restart the Control Center Policy Server service.

Related tasks

[Recovering your username or password](#) on page 30

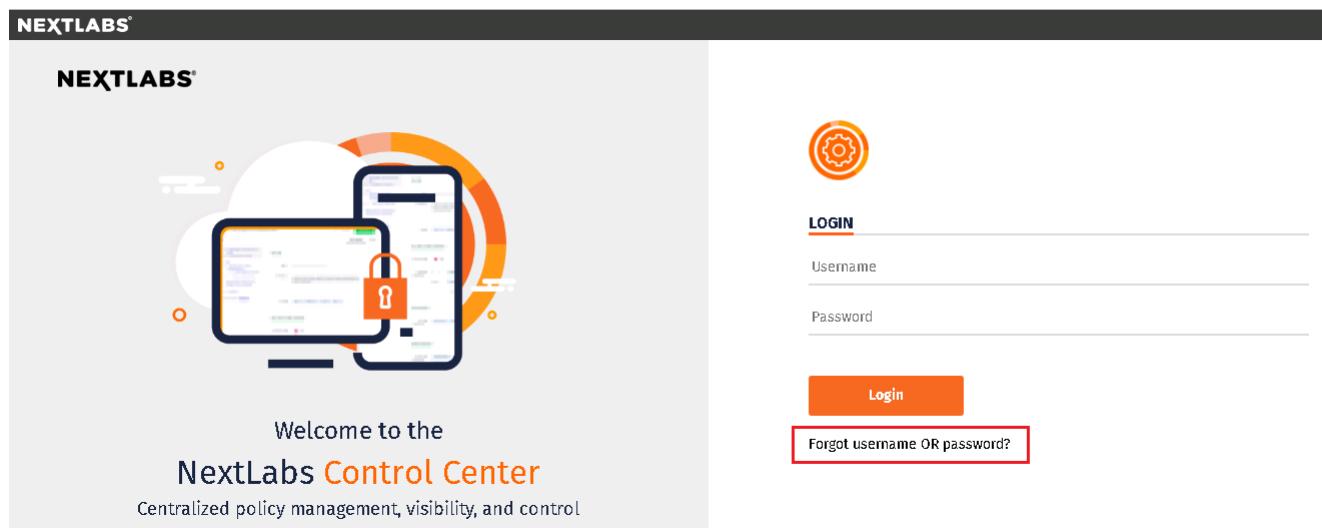
If you have forgotten your username or password, follow these steps to have it emailed to you or reset.

Recovering your username or password

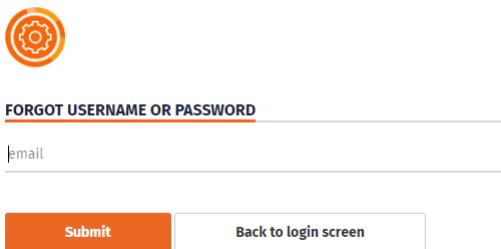
If you have forgotten your username or password, follow these steps to have it emailed to you or reset.

 **Important:** Ensure that the mail settings are configured in the `cas.properties` file.

1. On the Control Center web console login page, click **Forgot username OR password?**



The FORGOT USERNAME OR PASSWORD pop-up window appears.



FORGOT USERNAME OR PASSWORD

email

Submit Back to login screen

2. In the email box, type your registered email address.

3. Click **Submit**.

An email containing your username and a link to reset your password is sent to your registered email address.

4. To reset your password, click the reset password link.

5. In the Password box, type the new password.

6. In the Confirm password box, retype the new password.

7. Click **OK**.

Your password is reset.

Related tasks

[Restarting the Control Center Policy Server](#) on page 350

Whether running on a single host, on distributed hosts, or in a failover cluster, the Control Center Policy Server service must be manually started after the initial installation.

[Configuring mail server settings after installing Control Center](#) on page 29

You can edit the `cas.properties` file if you have not configured your mail server settings during Control Center installation, or if you want to change the existing mail server settings.

About importing user information from AD sources

You can connect the Control Center console to an Active Directory (AD) server to import user information.

After user information is imported, authorized users can log in to the Control Center web console using their system login credentials.

Configuring AD sources

To import user information, you must configure the connection between the Control Center and the AD server.

1. Log in to the Control Center web console with an account that has permission to manage user sources.
2. On the left navigation bar, select **Administration > User Source**.
3. Click **ADD USER SOURCE**.
4. From the Type drop-down list, select **Active Directory/LDAP**.
5. Provide the following information:

Option	Description
Name	The name you want to give to the AD server. This name appears on the list of AD sources.

Option	Description
LDAP URL	The URL and port number of the LDAP server. For example: ldap://domain:389
LDAPS URL	The URL and port number of the LDAP server. For example: ldaps://domain:636
LDAP domain	For more information, see Configuring the certificates for LDAP servers on page 62.
LDAP Root DN	The domain of the LDAP server. For example: domain.example.com
Username	The name of the root DN (distinguished name). This entry identifies the directory in which user searches occur. For example: ou=Presidents,dc=tdomain,dc=lab01,dc=nextlabs,dc=nextlab,dc=com
Password	The name of an account that has read access to the directory.
LDAP User Filter	The password of the specified account.
User Search Base	The filter used to search the AD for users. For example: userPrincipalName={dn}
	The location in the AD server where searches start. For example: (&(objectclass=user)(objectclass=person)(objectclass=organizationalPerson) (! (objectclass=computer)))

6. To verify the login information, click **Connect**. The Control Center attempts to connect to the AD server using the specified configuration. A message appears stating whether or not the connection was successful. If the connection is not successful, verify the configuration information and try again.
7. In the User Attributes section, map the Control Center attributes `username` and `firstName` to the appropriate attributes in the AD source. For example, `sAMAccountName=username`.
8. Optional: Add any user attributes from the AD source that you want to import to Control Center user accounts when users are imported. For example, if your AD information includes job titles for users, you could add that attribute here, and it would be imported as a user attribute when you add users from the AD source. User attributes can be used in policies to control user access to resources.

 **Note:** When users are added to Control Center from an AD source, only the user attributes listed in this section are imported as part of the user information.

9. Click **Save**.

When the connection between the Control Center and the AD connection is successful, you can add users from the AD source.

Related tasks

[Configuring the certificates for LDAP servers](#) on page 62

Obtain and configure certificates for LDAP servers.

Adding users from an AD source

After you have configured an AD source, you can create Control Center users by importing user information from that source.

1. Log in to the Control Center web console with an account that has permission to manage users.
2. On the left navigation bar, select **Administration > Users**.
3. Select **ADD EXTERNAL USER**.

4. In the Import user from drop-down list, select the AD source from which you want to import users.
5. Do any of the following:
 - In the Search for user first name or username drop-down list, enter information related to the users you want to add.
 - In the Number of Records drop-down list, change the number of records to display on the page.
 - Select the check box to the left of the username to select specific users.
 - Select the check box above the list to select all users on the page.
6. Perform one of the following actions.
 - Click **Apply** to apply the filters and display users list as per the filter criteria.
 - Click **Import** to import selected users.

Imported users can immediately log in to the Control Center using their domain credentials and perform actions according to specified delegation policies. When user information is updated in the Active Directory, it is automatically updated in the Control Center provided that the Control Center remains connected to the AD server.

About importing user information from Azure AD

You can connect the Control Center console to Azure AD to import user and group information.

After user and group information is imported, authorized users can log in to the Control Center web console using their system login credentials.

Configuring the connection between the Control Center and Azure AD

To import user information, you must configure the connection between the Control Center and Azure AD.

1. Log in to the Control Center web console with an account that has permission to manage user sources.
2. On the left navigation bar, select **Administration > User Source**.
3. Click **ADD USER SOURCE**.
4. From the Type drop-down list, select **Microsoft Azure AD**.
5. Provide the following information:

Option	Description
Name	The name you want to give to the Azure AD source. This name appears on the list of directories. For example: Microsoft AAD Authentication.
Enabled	If enabled, only this user can log in to Control Center using Azure AD credentials.
Username	The name of an account that has read access to the directory.
Password	The password of the specified account.
Authority URI	The URL to access Azure AD. For example: https://login.microsoftonline.com/
Extended Attribute URI	The URL to the Microsoft API that provides access to extensive user and group attributes that can be used by Control Center. For example:

Option	Description
	<code>https://graph.microsoft.com/beta/users/\${id}</code>
	 Note: The beta URL is subject to change. Specify the updated URL if the beta URL changes.
Tenant ID	The ID of the representative using the directory.
	 Note: Register your application as multi-tenanted if you want to set as multi-tenanted.
Application ID	The unique identifier Azure AD issues to an application registration that identifies a specific application and the associated configurations.
Application Key	The key value that you provide with the application ID to sign in as the application.

6. Click Save.

Adding users from Azure AD

After you have configured the connection between the Control Center and Azure AD, you can create Control Center users by importing user information from Azure AD.

1. Log in to the Control Center web console with an account that has permission to manage users.
2. On the left navigation bar, select **Administration > Users**.
3. Select **ADD EXTERNAL USER**.
4. In the Import user from drop-down list, select the AD source from which you want to import users.
5. Do any of the following:
 - In the Search for user first name or username drop-down list, enter information related to the users you want to add.
 - In the Number of Records drop-down list, change the number of records to display on the page.
 - Select the check box to the left of the username to select specific users.
 - Select the check box above the list to select all users on the page.
6. Perform one of the following actions.
 - Click **Apply** to apply the filters and display users list as per the filter criteria.
 - Click **Import** to import selected users.

Imported users can immediately log in to the Control Center using their domain credentials and perform actions according to specified delegation policies. When user information is updated in the Active Directory, it is automatically updated in the Control Center provided that the Control Center remains connected to the AD server.

Changing the superuser password

The superuser Administrator account cannot be deleted or renamed. However, anyone logged in to the Control Center interface with the superuser account can change the superuser password as described in this section.

1. Log in to the Control Center web console as the superuser Administrator.
2. In the drop-down list in the upper right of the screen, select **Change Password**.

3. Provide the old and new passwords, then click **Submit**. The password for the Control Center interface and all utilities is changed.

Chapter

4

Enrolling users, hosts, and groups

Topics:

- [About enrollment](#)
- [Entities that can be enrolled](#)
- [About Domain Group enrollment](#)
- [Overview of enrollment types and procedures](#)
- [About the Enrollment Manager utility](#)
- [About enrollment input files](#)
- [Enrolling users and hosts](#)
- [How to maintain enrollments](#)
- [Managing enrollment tasks](#)

This section explains enrollment and provides instructions for enrolling users, groups, and host groups from LDAP, and users and user groups from Azure Active Directory (Azure AD).

About enrollment

Enrollment is the process of importing LDAP directory entities, SharePoint user groups, hosts, and other entities into the Control Center database. For Azure AD, only users and groups can be imported into the Control Center database.

It is one of the most important activities for Control Center administrators. During enrollment, information about the entities is copied and stored in the Information Network Directory, one of Control Center's internal databases. This makes the information available for use by NextLabs components. For example, after user information is enrolled, policies can be written to restrict document access based on user information.

Enrollment occurs as two basic steps: enrolling and syncing. Enrolling refers to mapping data types to the Control Center in preparation for data population. Syncing refers to actually retrieving data values and populating the Control Center Information Network Directory.

Enrollment is performed after installation and is a crucial step in setting up the Control Center system. However, it is also an important part of ongoing administrative responsibilities because entities such as location sites, applications, and SharePoint groups that are added or changed need to be enrolled to be available to NextLabs components.

About the Information Network Directory

The Information Network Directory (IND) is the database schema where Control Center stores information about organization entities, such as users, computers, groups, applications, and so on.

The database uses a structured information resource model, which provides consistent definition and organization to make it possible to define policy components. The Information Network Directory is installed on the same database host as the Activity Journal, in PostgreSQL, Oracle, or SQL Server.

When information is enrolled, it is connected with an enrollment domain. In this context, the term "domain" refers to a source where enrollment information resides, not a network domain. Further, multiple enrollment domains can be imported from different sources within a single Active Directory in a single network domain. Entities can be enrolled from any number of enrollment domains.

The initial enrollment should be performed soon after the Control Center is installed. The Enrollment Manager can be configured to automatically synchronize the information resource model at regular intervals. The information can also be updated manually as needed to match the changing needs of the organization.

Entities that can be enrolled

The Control Center enrollment utilities import information about the underlying information network, including the entities described in this section.

Most entities are mandatory; only location sites and SharePoint groups are optional. For a summary of entities, see the Overview of enrollment types and procedures section.

Required entities

Enrolling the following entities is required for all installations.

- **Users and User Groups:** Users and User Groups are required for creating user components. The Enrollment Manager enables administrators to enroll entities directly from an LDAP server, such as Microsoft Active Directory, enroll groups from SharePoint, or enroll information from any correctly formatted LDIF (Lightweight Directory Interchange Format) file. Users and user groups can be enrolled from more than one enrollment domain into the same Control Center. For users, hosts, and groups, you can also create Domain Groups to enroll information that spans multiple enrollment domains.

- Hosts and Host Groups: Hosts and host groups are required for creating computer components. The Enrollment Manager enables administrators to enroll hosts and host groups either from an LDAP server directory or from an LDIF file, and from one or more Active Directory domains. Hosts and Hosts Groups can also be enrolled through a Domain Group.
- Applications: Applications are required for creating Application Components for any applications you want to monitor
- File Shares: File share information is required to facilitate mapping to specific resource paths. File shares are enrolled using a utility called Resource Path Discovery, which queries servers in the organization.

Related concepts

[Enrolling users and hosts](#) on page 61

This section includes procedures related to configuration options and for performing enrollments:

[Enrolling file shares](#) on page 86

Enrolling information about file shares ensures that the use of differing file share paths to access the same document does not result in inconsistent policy enforcement.

Related tasks

[Enrolling applications](#) on page 83

Applications, unlike users, cannot be automatically enrolled from LDAP directories.

Optional entities

Enrolling the following entities is optional.

- SharePoint Groups: SharePoint groups need to be enrolled only if the organization uses NextLabs SharePoint Enforcers and plans to design policies to monitor and control the servers. NextLabs SharePoint Enforcers are available only with the NextLabs Entitlement Management product. SharePoint groups are enrolled using the Enrollment Manager. SharePoint Groups can also be included in a Domain Group enrollment.
- Location Sites: Location sites are physical or logical locations, such as Headquarters, Boston Office, VPN, expressed as one or more ranges of IP addresses. These entities can be enrolled and used to define location-based computer components. Site information is gathered from a text file that specifies the computers included in each site. Location sites are enrolled using a utility called Import Locations.

Related concepts

[Enrolling SharePoint groups](#) on page 91

The SharePoint Enforcer is available exclusively with the Entitlement Management product. If you are not using that product, you can ignore this section.

Related tasks

[Enrolling location sites](#) on page 89

A location site is a Control Center term referring to a group of hosts that you want to consider as a single location because they share certain characteristics.

Enrolling entities for Windows and Linux systems

Entities are enrolled into the Control Center system the same way regardless of whether enforcers are being installed on Windows file servers, Linux file servers, or a mixture of both.

In other words, there is no such thing as Windows enrollment or Linux enrollment. Administrators simply need to ensure that the Samba server configuration maps the LDAP users in the Active Directory on the Windows side, to the users defined on the Linux side. Ordinarily, this has to be done in any case, so the two systems can work together. As long as this is configured properly, Control Center manages all enrolled entities internally, and policies are enforced properly on both Linux and Windows servers.

The one exception to this is that discovering and enrolling file shares requires running separate utilities on the Windows and Linux environments.

 **Note:** File server enforcers are available with the NextLabs Entitlement Management product only.

Related concepts

[About Resource Path Discovery](#) on page 86

The enrollment process involves discovering file shares defined on network hosts, and then enrolling them into the Information Network Directory.

About Domain Group enrollment

A Domain Group contains multiple User, Host, and Group sources, referred to as subdomains, that can be managed together in a single enrollment process.

The Domain Group option is required for enrollments where entities are located across multiple sources and there are interdependencies between the entities.

Consider the following example. Users are stored in two different sources: one set of users in Active Directory (AD) and one set managed in an LDIF file as shown in the following figure. Both sets of users are referenced in Groups managed within SharePoint. To design policies around the SharePoint Groups, an administrator would need to enroll all three sources into Control Center and retain information about Group membership across the two user domains (AD and LDIF) using the Domain Group enrollment option. All pertinent enrollment sources would be listed as “subdomains” in the Domain Source file, and they would be enrolled and managed as part of one process.

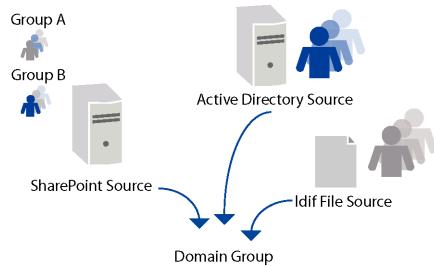


Figure 2: Users and Groups Managed Across Multiple Sources

Related tasks

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

Enrollment failures

It is possible that an enrollment may fail for various reasons—a connectivity problem, improper syntax in a connection file or definition file, a typo in the command line parameters, and so on.

Even if an enrollment attempt fails, it is still recorded as an enrollment, and is present in the management data repository. Administrators must manually delete failed enrollments using the `-delete` command before re-attempting the enrollment.

For Domain Group enrollment, if a subdomain fails, all subsequent subdomains listed in the group are still synced or enrolled.

Control Center considers each instance of enrolling a specified set of LDAP data as a distinct metadata entity called an enrollment. Information about enrollments is saved internally, independent of the actual data that was imported, so that administrators can refer to it later if necessary. All events connected to enrollments are stored in a special log file.

Overview of enrollment types and procedures

Different types of enrollments require different arguments and use different auxiliary files.

The following table provides an overview of the procedures for different enrollment methods, with links to more information.

Table 6: Summary, enrolling Control Center entities

Entity type	Enrollment method	Connection file	Definition file or Configuration file	Reference
Users & User Groups; Hosts & Host Groups	Enrollment Manager utility Type = DIR (from AD or other LDAP directory), or LDIF (from LDIF file)	DIR: ad.sample.default.conn LDIF: None AAD (only users and groups): azure.sample.default.conn	Definition file: ad.sample.default.def Definition file: ldif.sample.default.def Definition file for Azure AD: azure.sample.default.def	Enrolling users and hosts on page 61
Domain Group	Enrollment Manager utility Type = DOMAINGROUP (can include enrollment types: DIR, PORTAL, or LDIF)	DIR: ad.sample.default.conn LDIF: None Portal: sp.sample.default.conn	Definition file: ad.sample.default.def Configuration file: domainGroup.sample.default.cfg Definition file: ldif.sample.default.def Definition file: sp.sample.default.def	Enrolling Domain Groups on page 65
Applications	Application Discovery utility to generate an LDIF file; then Enrollment Manager utility, Type = LDIF	None	Definition file: app.sample.default.def	Enrolling applications on page 83 & Enrolling users and hosts on page 61
File Shares	Windows: Resource Path Discovery utility Linux: Samba Directory mapping utility	None	None	About Resource Path Discovery on page 86 & Enrolling Linux file servers on page 88
Location Sites	Manually create a plain-text Locations file, then Import Locations utility	None	None	Enrolling file shares on page 86
SharePoint Groups	Enrollment Manager utility, Type = PORTAL	sp.sample.default.conn	Definition file: sp.sample.default.def	

About the Enrollment Manager utility

The Enrollment Manager utility enrolls entities into the Network Information Directory, directly from an LDAP directory, an LDIF file, or Azure AD.

These entities include users and user groups, hosts and host groups, applications, and SharePoint groups. For location sites and file shares special utilities for Windows and Linux are used.



Note: Enrollment Manager can be used only by authorized users who are assigned a System Administrator role. Authorized users are defined in the Administrator console, on the Users and Roles tabs.

How enrollment connections are established

The Enrollment Manager utility can be installed on any host in the network, as long as the utility can connect to the server where Control Center is running.

Similarly, the AD, Azure AD, or other LDAP server can be anywhere in the network, remote from the Control Center. This means there are two connections required: the Enrollment Manager's connection

to the Control Center Server, and the Control Center server's connection to the LDAP server or other enrollment domain.

The section describes the connection processes for single domain enrollment and domain group enrollment. This section also introduces the various input files, connection, definition, filter, and configuration, that are used to configure these connections.

Related concepts

[About enrollment input files](#) on page 47

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

Connection during an enrollment

The connection process is as follows.

- The connection between the Enrollment Manager and the Control Center Server is defined at the command line, when the `-enroll` or `-sync` command runs. The `-s` (server) and `-p` (port) options point the Enrollment Manager to the Control Center Server.
- For single domain enrollments, the command line provides the location of the connection file (the `-a` option), definition file (`-d`), and filter file (`-f`). The connection file establishes the connection between the Control Center and the LDAP server. The definition and filter files specify what information should be retrieved from the enrollment domain. For domain group enrollments, use the `-g` command to provide the location of the configuration file, which references all connection, definition, and filter files.
- The Control Center then uses the connection information to establish its connection to the LDAP server. The connection is defined in the first four lines of the connection file—server, port, login, and password.
- In the case of a Domain Group enrollment, when the enrollment (`-enroll`) of the first subdomain is complete, the Control Center locates the next subdomain listed in the configuration file, and continues enrolling until it completes the last subdomain.

The connection sequence for a single domain enrollment is represented in the following figure.

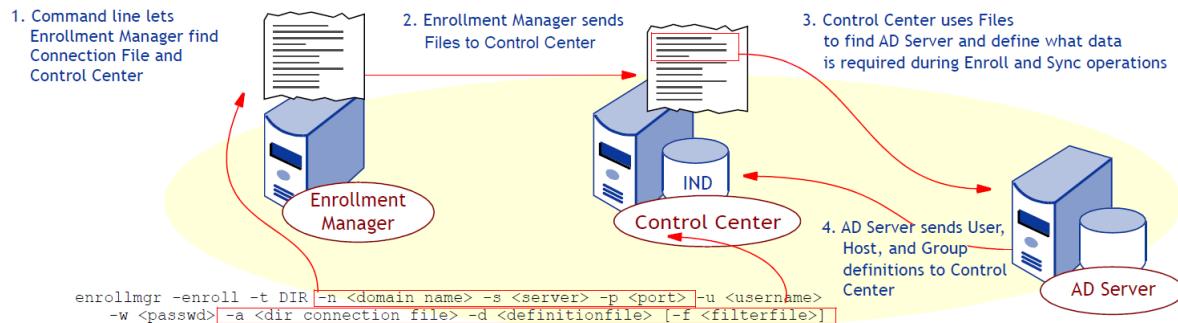


Figure 3: Connections to Retrieve AD Users, Groups, and Hosts (Single Domain)

Related concepts

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Definition files](#) on page 54

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

About secure LDAP connections

LDAP connections can be configured to use standard SSL protocols to establish secure communication between the Control Center and the LDAP Server.

Configuration steps include obtaining the LDAP server's certificate and locating it on the Control Center host, inserting the certificate into the Control Center's truststore, and configuring settings in the connection files. For Domain Group enrollment, these steps would need to be performed for each subdomain.



Note: In the current release, LDAP and LDAPS are supported.

Related concepts

[Configure a secure LDAP connection](#) on page 61

After an enrollment is defined, a manual or automatic (scheduled) sync establishes a connection between the Control Center and the enrollment domain.

About Azure AD connections

This section includes procedures related to creating and configuring an Azure AD application.

Creating an Azure AD application

This section describes the steps that are required to create an Azure AD application.

1. In your browser, type:

<https://portal.azure.com/>

2. Log in to your Azure account using an account that has the privilege to create an Azure AD application.

3. On the left navigation pane, click **Azure Active Directory**.

4. Click **App registrations**.

5. Click **New application registration**.

6. In the Name box, specify a meaningful name for the application.

For example: NextLabs – Control Center.

7. From the Application type drop-down list, select **Web app / API**.

8. In the Sign-on URL box, type the following URL.

https://server_domain/console

9. Click **Create**.

Configuring the logout URL

This section describes the steps that are required to configure the logout URL for your application.

1. Navigate to the App Registrations blade, and then select your app.

2. Click **Settings**.

3. Click **Properties**.

4. In the Logout URL box, type the following URL.

https://server_domain/console/logout

5. Toggle between multi-tenanted. This affects the tenant ID value when setting up the user source in the Control Center console.

6. Click **Save**.

Configuring the reply URL

This section describes the steps that are required to configure the reply URL for your application.

1. Navigate to the App Registrations blade, and then select your app.

2. Click **Settings**.

3. Click **Reply URLs**.

4. In the blank box, type the following URL.

https://server_domain/cas/authority/callback

5. Click Save.

Granting permissions to your application

This section describes the steps that are required to grant permissions to your application.

1. Navigate to the App Registrations blade, and then select your app.
2. Click **Settings**.
3. Click **Required permissions**.
4. Click **Windows Azure Active Directory**.
5. In the DELEGATED PERMISSIONS section, select the check boxes next to the following labels:
 - Read hidden memberships
 - Sign in and read user profile
 - Read all users' basic profiles
 - Read all users' full profiles
 - Read all group
 - Read directory data
 - Access the directory as the signed-in user
6. Click **Save**.
7. Request your administrator to grant application permissions.

Getting an application ID and an authentication key

This section describes the steps that are required to get your application ID and authentication key.

1. From the App registrations page in Azure Active Directory, click your application.
2. Copy the application ID and store it in your application code.
3. To generate an authentication key, follow these steps.
 - a. Click **Settings**.
 - b. To generate an authentication key, click **Keys**.
 - c. Provide a description of the key, and a duration for the key.
 - d. Click **Save**.

After saving the key, the value of the key is displayed.



Important: Copy the key value because you cannot retrieve the key later. You provide the key value with the application ID to log in as the application. Store the key value where your application can retrieve it.

About connections and AD load balancing

If Active Directory is running on more than one host with a load balancing device sharing the AD authentication traffic, the enrollment must be run directly to one or the other hosts, not to the load sharing device.

Specifically, this means the hostname and port specified in the connection file must refer to one of the AD hosts directly, not the load balancing appliance.

Before you enroll

Before enrolling LDAP data, you must run the Control Center installation wizard, and then start Control Center.

Specifically, the Management Server component must be installed and running before you begin the enrollment.

Related tasks

[How to enroll from a directory \(single domain\)](#) on page 63

This section explains how to enroll from a single domain.

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

LDAP directory requirements

Before you start, be sure any data to be enrolled from the LDAP directory meets the following requirements.

- All entries must have an `objectClass` attribute.
- All entries must belong to the corresponding object class, as shown in the following table:

Table 7: Entries and classes

Entry	objectClass
User	person
Host	computer
Structural group	organization or organizationalUnit
Enumerated group	group

- All entries must have an `objectGUID` attribute, which holds the static identifier of that entry in the source directory.
- All user and host entries (also called “terminal” entries) must have an `objectSid` attribute. This attribute holds the system reference information for that entry—that is, the token or identifier that identifies the entity within the Windows system. This attribute is assigned to each user and host by the LDAP directory. If this attribute is not present, the entry is ignored and a warning is logged.
- User entries must have a `userPrincipalName` attribute that serves as a unique, user-friendly name for this entry within the directory.
- Host entries must have a `dnsHostName` attribute that serves as a unique, user-friendly name for this entry within the directory.
- If Linux file servers are combined with Windows AD systems, configure the Samba server so that all users and groups map properly between the Windows and Linux systems.

Requirements for Domain Group enrollment

An additional requirement applies for Domain Group enrollment.

Trust relationships between subdomains of a Domain Group must be manually configured to ensure that policy bundles are created with all subdomain information.

Related concepts

[Configuring trusted domains](#) on page 139

During normal operation, Control Center tracks users and hosts according to the domain where they are enrolled, for the purposes of preparing policy bundles. Control Center needs this information so that it can include, in policy bundles, only the information that is relevant to a particular policy enforcement point.

Using the Enrollment Manager

The Enrollment Manager is automatically installed on the Control Center host.

The Enrollment Manager is installed in the following directory.

`\Program Files\NextLabs\PolicyServer\tools\enrollment`

If Control Center components are divided among several hosts in a distributed installation, the Control Center host refers to the server where the Management Server is installed. However, the Enrollment Manager may be installed anywhere in the network, as long as it can connect to the Control Center host.

To use the Enrollment Manager, open a command line window, navigate to this location, and use the command `enrollMgr` plus the specific command and arguments for the desired function.

The `-enroll` and `-update` commands require a reference to a definition file and/or a connection file. Definition files provide details on how information elements in the source directory or file map to the data structures inside Control Center's data tables. Connection files provide information necessary to connect to the LDAP directory data source. For the Domain Group option, Configuration files list all subdomains in a Domain Group as well as the location of relevant connection, definition, and filter files. When using the `-enroll` and `-update` commands, reference the configuration file only.

Related concepts

[Definition files](#) on page 54

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Configuration files for Domain Groups](#) on page 58

Domain Group enrollment requires an additional input file: the Configuration file.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Command line arguments for Enrollment Manager

Command	Purpose	Full Command String
-enroll	Enroll entities from an LDAP directory server or a source file	<code>enrollmgr -enroll -t <type> -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>} -a <ad_connection_file> -d <definitionfile> [-f <filterfile>]</code>
	Enroll entities from Azure AD	<code>enrollmgr -enroll -t AAD -n <domain_name> -s localhost -p <port> -u <username> {-w <passwd> -ew <encryptpwd>} -a <Azure_AD_connection_file> -d <Azure_AD_definitionfile></code>
		Not all arguments are required for all enrollment types.
-update	Update an existing enrollment from an LDAP directory or a source file	<code>enrollmgr -update -t <type> -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>} -a <ad_connection_file> -d <definitionfile> [-f <filterfile>]</code>
		Not all arguments are required for all enrollment types.
-sync	Synchronize an enrollment	<code>enrollmgr -sync -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>}</code>
	Synchronize an enrollment for Azure AD	<code>enrollmgr -sync -n <domain> -u <username> {-w <passwd> -ew <encryptpwd>}</code>
-delete	Delete an enrollment	<code>enrollmgr -delete -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>}</code>
-list	List enrollments	<code>enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>}</code>
	List enrollments for Azure AD	<code>enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> -ew <encryptpwd>}</code>
-h	Display a help screen showing definitions of all valid command line arguments	<code>enrollmgr -h</code>

The following table provides an overview of different arguments that can be supplied to establish connections and to reference the location of input files, including connection, definition, filter, and configuration files.

Argument	Description
<code>-t</code>	Used with the <code>-enroll</code> and <code>-update</code> commands to specify the type of enrollment. Valid values are: <ul style="list-style-type: none"> • <code>DIR</code>: enrolling from an LDAP server • <code>PORTAL</code>: enrolling SharePoint group definitions • <code>LDIF</code>: enrolling from an LDIF file • <code>DOMAINGROUP</code>: enrolling multiple subdomains in a single enrollment (which can include <code>DIR</code>, <code>PORTAL</code>, and <code>LDIF</code> type enrollments) • <code>AAD</code>: enrolling from Azure AD These values are case sensitive, and must be capitalized as shown.
<code>-n <domain_name> or <enrollment_name></code>	The name of the enrollment. Each enrollment needs a unique name, so it can be identified in various management functions, such as confirming, synchronizing, and updating. For User and Host enrollments, this value must be the name of the domain from which entities are being enrolling. Accordingly, domains can only be enrolled once. After a domain is enrolled, it can be updated using the same name to incorporate any changes.
<code>-g <domaingroup_file></code>	SharePoint or application enrollments do not need to use the domain name; they can use any name. This makes it possible to perform multiple enrollments from the same SharePoint server, as long as the names are unique.
<code>-s <server></code>	A Domain Group also receives its own domain name, as does each subdomain in the Domain Group. Subdomain enrollment names are defined in the Configuration file.
<code>-p <port></code>	The location of the Domain Group configuration file.
<code>-u <username></code>	The name or URL of the host where Control Center is running. For distributed Control Center installations, where Enrollment Manager is installed on a different host, this server should be the host where the Enrollment Manager is installed. If this argument is not present, the default value <code>localhost</code> is used.
<code>-w <passwd></code>	The port of the Control Center host, which is 8443 unless it was explicitly changed. This argument is optional; if it is not present, the default value 8443 is used.
<code>-ew <encryptpwd></code>	The username of a Control Center user with administrator privileges. NextLabs recommends using a placeholder username, dedicated only to enrollment functions, rather than using a live account. This prevents problems that can arise when a live user changes a username or password.
<code>-a <connection_file></code>	Password for the Control Center administrative user specified by <code>-u</code> . NextLabs recommends using a placeholder password, dedicated only to enrollment functions, rather than using a live account.
<code>-d <definitionfile></code>	Encrypted password for the Control Center administrative user specified by <code>-u</code> . You can generate an encrypted password using the <code>mkpassword.bat</code> utility.
<code>-f <filterfile></code>	The name of the directory connection file, which is required for enrolling or updating with an LDAP server and for enrolling portal sites. To create a connection file, copy one of several provided template or default files that matches the kind of entity being enrolled.
<code>-v</code>	The name of the definition file, which is required for all enrollments and updates. To create a definition file, copy one of several provided template or default files that matches the kind of entity being enrolled.
<code>-delta</code>	(Optional) The name of the LDAP filter file.
	Verbose option; may be used with any command. When this is used and an error occurs, a detailed description of the issue appears in the command window. If this option is not used, a shorter summary is displayed.
	(Optional) Make changes to an existing enrollment after performing manual synchronization.

Related concepts

[Connection during an enrollment](#) on page 41

The connection process is as follows.

[Using the Enrollment Manager](#) on page 44

The Enrollment Manager is automatically installed on the Control Center host.

[Configuration files for Domain Groups](#) on page 58

Domain Group enrollment requires an additional input file: the Configuration file.

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Definition files](#) on page 54

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

[Filter files](#) on page 56

Filter files enable you to selectively import leaf element—users, contacts, hosts, applications, and groups—from a directory server.

Related tasks

[Incremental synchronization using LDIF delta file](#) on page 74

If you want make changes to an existing enrollment after performing manual synchronization, you can use the Enrollment Manager utility with the `-delta` option to perform incremental synchronization.

[How to enroll from a directory \(single domain\)](#) on page 63

This section explains how to enroll from a single domain.

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

[Updating directory enrollments for a single domain](#) on page 69

You should use the `-update` command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

[Updating LDIF file enrollments for a single domain](#) on page 70

You can use the `-update` command to update user and host information in the Control Center Information Network Directory, based on the content of an LDIF file.

[Updating Domain Group enrollments](#) on page 71

If you make changes to any of the input files for the subdomains for an existing Domain Group enrollment, or add or delete subdomains for an enrolled Domain Group, you need to update the enrollment.

[Manual synchronization](#) on page 73

The `-sync` command reuses the parameters provided the last time the `enroll` command was called for the specified enrollments—including the references to the appropriate definition and connection files—based on the enrollment name parameter (`-n`).

[Enrolling SharePoint users and groups](#) on page 92

Enrolling SharePoint groups is similar to the procedure for enrolling users and groups.

About enrollment input files

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

- Connection files
- Definition files
- Filter files

- Configuration files for Domain Groups

Related concepts

[How enrollment connections are established](#) on page 40

The Enrollment Manager utility can be installed on any host in the network, as long as the utility can connect to the server where Control Center is running.

Related tasks

[How to enroll from a directory \(single domain\)](#) on page 63

This section explains how to enroll from a single domain.

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

[Updating directory enrollments for a single domain](#) on page 69

You should use the `-update` command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

[Updating LDIF file enrollments for a single domain](#) on page 70

You can use the `-update` command to update user and host information in the Control Center Information Network Directory, based on the content of an LDIF file.

About connection files

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

Directory connection files must be specified for Enroll or Update commands that use the type = DIR or PORTAL. Directory connection files are not required for commands that use type = LDIF or TEXT. SharePoint connections are relevant only to Entitlement Management products. For Domain Groups, connection files are used for DIR and PORTAL subdomains, but not for LDIF subdomains. Rather than referencing the location of connection files when running the `enroll` or the `update` command, you reference the configuration file.

This file also controls the automatic directory synchronization feature, which is available only for directory enrollments, not LDIF file enrollments.

Several specialized connection file templates are automatically copied to the Control Center host during installation in the following directory:

`tools\enrollments`

Templates include:

- `ad.sample.default.conn`: for LDAP directory.
- `sp.sample.default.connL`
- `Azure AD: azure.sample.default.conn`

Related concepts

[Connection during an enrollment](#) on page 41

The connection process is as follows.

[Using the Enrollment Manager](#) on page 44

The Enrollment Manager is automatically installed on the Control Center host.

[Scheduled synchronization](#) on page 77

You can configure an enrollment to be automatically synchronized on a regular schedule. This is recommended for all production implementations.

[Enrolling other entities](#) on page 82

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

[Elements of Active Directory connection files](#) on page 50

[Elements of configuration files](#) on page 59

This section describes the elements of the configuration file for domain group enrollment.

Connection file details

The connection file is a template that contains a number of commented lines explaining how to use the elements.

The following figures show the elements in the ad.sample.default.conn and the azure.sample.default.conn file. Make a copy of the template, rename it as needed, and, for all the elements indicated by rectangles in the figure, replace the placeholder values with actual values. Change the optional settings, which are indicated by the ovals in the example, as needed.

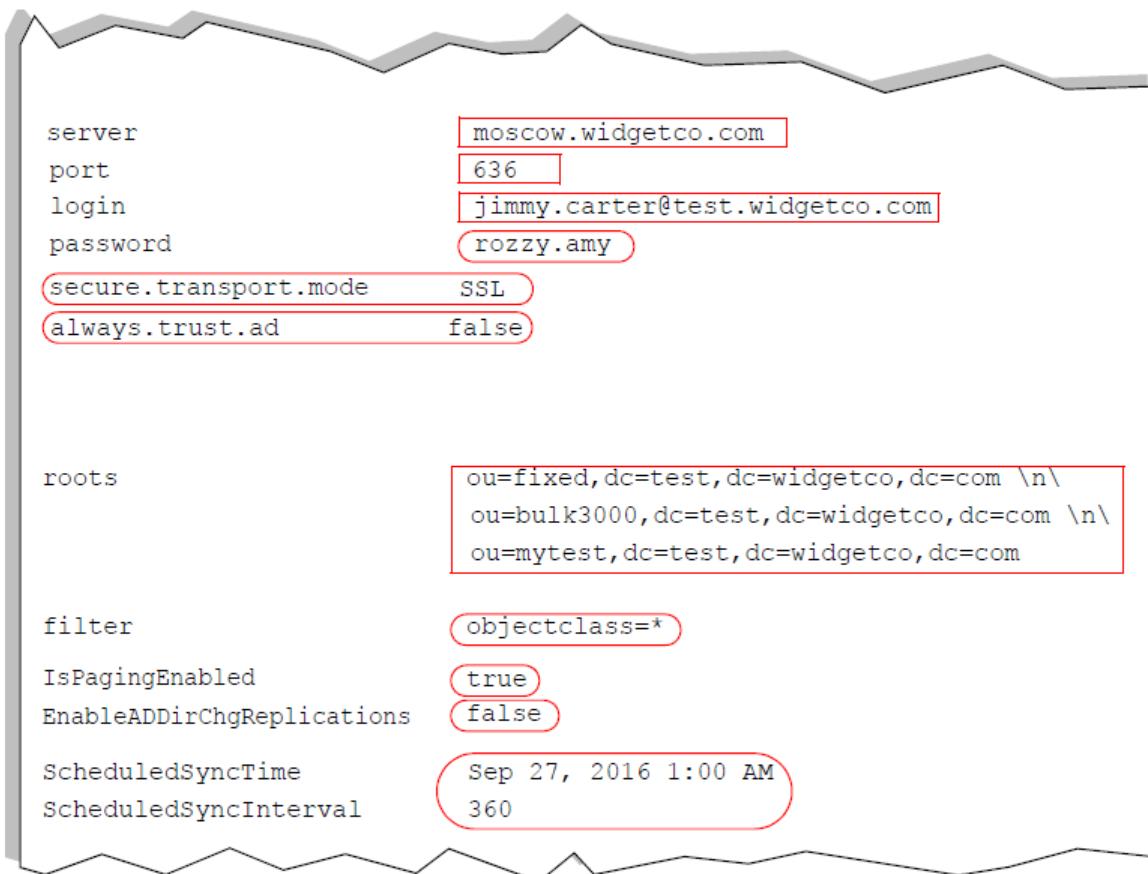


Figure 4: Sample directory connection file (ad.sample.default.conn)

```

# This is the connection file for a default Azure AD enrollment

## Usually this value will not need to be changed
azure-oauth-authority      https://login.microsoftonline.com/

tenant          mytenant.onmicrosoft.com
application-id <application-id>

# Comment this line out to be prompted for the value when performing
# enrollment
application-key <application-key>

# Paging should be used for large enrollments
IsPagingEnabled true

#-----
# The start time and pull interval for automatic sync
# The format of ScheduledSyncTime must be format of "Oct 4, 2006 8:14 PM"
# The ScheduledSyncInterv is a positive number in minutes.
# Zero value of ScheduledSyncInterv means auto-sync is disabled
#-----
ScheduledSyncTime Sep 27, 2006 8:14 AM
ScheduledSyncInterv 0
ScheduledSyncTimeFormat dd-MMM-yyyy HH:mm

```

Figure 5: Sample Azure AD connection file (azure.sample.default.conn)



Note: When editing these files, be sure to change the filename. Template files are overwritten when Control Center is upgraded, and any changes made to these files would be lost during the next upgrade. For example, you could delete the string .sample from the filename to make it easy to identify the template on which the file is based.

Related concepts

[Scheduled synchronization](#) on page 77

You can configure an enrollment to be automatically synchronized on a regular schedule. This is recommended for all production implementations.

Configure connection files for Domain Group enrollment

Connection file configuration is similar for both single and Domain Group enrollments, with two important differences.

- Sync times and intervals are not specified in connection files used for a Domain Group enrollments. Instead, the sync time and interval values are defined in the configuration file for the Domain Group.
- When running the Enrollment Manager for a Domain Group, do not point to the location of the connection file in the command line. Instead, point to the location of the configuration file, which points to all input files (connection, definition, and input files) for the listed subdomains.

Related tasks

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

Elements of Active Directory connection files

The following table describes the elements of Active Directory connection files.

Table 8: Elements of Active Directory connection files

Element	Description
server	The hostname of the LDAP server.
port	The LDAP server port number. The default port for LDAP is 389. For LDAPS the default port is 636, but these ports can be changed as needed.
login	The username of an account with access privileges sufficient to connect to the LDAP server and access the enrollment information. The login uses this format <name>@<domain>.com.
password	(Optional) The password associated with the login. The connection file is an unencrypted string, so the password parameter is optional. Including passwords in the connection file is convenient during testing or in other restricted-use contexts when security is not a concern. If the password is not included in the file, however, the user is prompted to enter the password during enrollment.
secure.transport.mode SSL	Use SSL for secure communication during manual or automatic (scheduled) sync between the Control Center and the LDAP Server. Currently, the only mode that is supported is LDAPS, so the only valid entry here is SSL. If this option is commented out, standard LDAP communication protocols are used. There are additional steps required to configure secure LDAP.
always.trust.ad	If this flag is set to true, Control Center trusts AD or LDAP servers without authenticating their security certificates. If this flag is not used, or is set to false, the AD or LDAP server certificates must be present in the Control Center truststore and authenticated.
roots	Each line describes one branch in the LDAP directory tree being enrolled. Technically, any number of root elements can be listed in a single connection file, as long as they are separated by the escape string \n\ at the end of each line. There are three root elements listed in the example above. However, it is generally easier to keep track of, update, and synchronize enrollments when a single branch is specified.
filter	The LDAP filter strings used in this file. For example, objectclass=user and department=HR. The default value, objectclass=* uses no filters.
IsPagingEnabled	A flag that indicates whether LDAP paging control is supported by the LDAP server. Change this flag to false if the LDAP server does not support this function.
EnableADDirChg Replications	A flag that controls Control Center's synchronization autotracking feature. When left at the default, false, Control Center reenrolls all entities from the Active Directory source during every synchronization. (This is specific to Active Directory.) When changed to true, the feature is enabled, and during synchronization Control Center enrolls only new entities and those whose properties have changed since the last sync operation. This feature makes sync procedures more efficient and faster, but it requires special permissions. See the Microsoft documentation: http://support.microsoft.com/?kbid=891995 .

Element	Description
ScheduledSync Time	<p>Automatically synchronize the enrollment with its source at the specified time. The enrollment is synchronized at the interval specified by the ScheduledSyncInterval parameter.</p> <p>The date and time value is locale-specific. Use Java's MEDIUM style for the date and the SHORT style for the time. For example:</p> <ul style="list-style-type: none"> For English (US): Sep 18, 2016 6:00PM For English (UK):18-Sep-2016 18:00 For the German (Germany):18.09.2016 18:00 <p>At enroll-time, the date must be a future date.</p> <p>This value is only active if the Scheduled Sync Interval is set to some positive value. If ScheduledSyncInterval is set to the default, 0, the synchronization feature is disabled.</p> <p>If the connection file is to be used in a configuration file as a part of Domain Group enrollment, no scheduled Sync Time should be specified. The sync time is defined in the configuration file.</p>
ScheduledSync Interval	<p>When left at the default, 0, disables the automatic synchronization feature. When changed to any positive value, specifies a regular re-synchronization schedule, expressed in minutes following the time specified by the ScheduledSyncTime.</p> <p>This setting has no influence until after the ScheduledSyncTime passes. That is, in the example above Control Center would wait until 1 a.m. on September 27 to synchronize the enrollment to its source, and then would re-sync it every 6 hours (360 minutes) thereafter.</p> <p>To perform manual synchronizations only, set this value to 0.</p> <p>If the connection file is to be used in a configuration file as a part of Domain Group enrollment, no scheduled Sync Time should be specified. The sync time is defined in the configuration file.</p>

Related concepts

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Configure a secure LDAP connection](#) on page 61

After an enrollment is defined, a manual or automatic (scheduled) sync establishes a connection between the Control Center and the enrollment domain.

Elements of Azure AD connection files

The following table describes the elements of Azure AD connection files.

Table 9: Elements of Azure AD connection files

Element	Description
azure-oauth-authority	The hostname of the LDAP server.
tenant	The identifier of your Azure AD tenant.

Element	Description
application-id	The username of an account with access privileges sufficient to connect to the LDAP server and access the enrollment information. The login uses this format <name>@<domain>.com.
application-key	(Optional) The password associated with the login. The connection file is an unencrypted string, so the password parameter is optional. Including passwords in the connection file is convenient during testing or in other restricted-use contexts when security is not a concern. If the password is not included in the file, however, the user is prompted to enter the password during enrollment.
IsPagingEnabled	A flag that indicates whether LDAP paging control is supported by the LDAP server. Change this flag to false if the LDAP server does not support this function.
ScheduledSync Time	Automatically synchronize the enrollment with its source at the specified time. The enrollment is synchronized at the interval specified by the ScheduledSyncInterval parameter. The date and time value is locale-specific. Use Java's MEDIUM style for the date and the SHORT style for the time. For example:
	<ul style="list-style-type: none"> • For English (US): Sep 18, 2016 6:00PM • For English (UK):18-Sep-2016 18:00 • For the German (Germany):18.09.2016 18:00
	At enroll-time, the date must be a future date.
	This value is only active if the Scheduled Sync Interval is set to some positive value. If ScheduledSyncInterval is set to the default, 0, the synchronization feature is disabled.
	If the connection file is to be used in a configuration file as a part of Domain Group enrollment, no scheduled Sync Time should be specified. The sync time is defined in the configuration file.
ScheduledSync Interval	When left at the default, 0, disables the automatic synchronization feature. When changed to any positive value, specifies a regular re-synchronization schedule, expressed in minutes following the time specified by the ScheduledSyncTime. This setting has no influence until after the ScheduledSyncTime passes. That is, in the example above Control Center would wait until 1 a.m. on September 27 to synchronize the enrollment to its source, and then would re-sync it every 6 hours (360 minutes) thereafter.
	To perform manual synchronizations only, set this value to 0.
	If the connection file is to be used in a configuration file as a part of Domain Group enrollment, no ScheduledSyncTime should be specified. The sync time is defined in the configuration file.
ScheduledSyncTimeFormat	

Definition files

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

The format and content of these files differs, depending on the enrollment source, and you can create and modify definition files as needed.

In addition, several specialized definition file templates are automatically copied to the Control Center host during installation in the following directory:

```
\tools\enrollments
```

Templates include:

- `ad.sample.default.def`, for use with Active Directory or other LDAP directories (`-enroll -t DIR`)
- `sp.sample.default.def`, for use in enrolling SharePoint groups (`-enroll -t PORTAL`)
- `ldif.sample.default.def`, for use with LDIF files for users, hosts, and groups (`-enroll -t LDIF`)
- `app.sample.default.def`, for use with LDIF files for application generated with the AppDiscovery utility (`-enroll -t LDIF`)
- `site.sample.default.def`, for use in enrolling host sites (`-enroll -t FILE`)

To use these templates, copy them to a known location, rename them, make the required changes, then point to them by filename and path in the `-d <definitionfile>` parameter in the `-enroll` or `-update` commands.

When editing these files, be sure to change the filename. Template files are overwritten when Control Center is upgraded, and any changes made to these files would be lost during the next upgrade. For example, you could delete the string `.sample` from the filename to make it easy to identify the template on which the file is based.

Related concepts

[Connection during an enrollment](#) on page 41

The connection process is as follows.

[Using the Enrollment Manager](#) on page 44

The Enrollment Manager is automatically installed on the Control Center host.

[The definition file](#) on page 84

The `-enroll -t LDIF` command requires a definition file.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

[Elements of configuration files](#) on page 59

This section describes the elements of the configuration file for domain group enrollment.

The Active Directory definition file

The Active Directory definition file elements need to be changed only if the mappings need to be changed.

For example, mappings might need to be changed if the Active Directory diverges from the default schema, or if the system uses a non Active Directory LDAP with a different schema.

The LDIF definition file

The default LDIF definition file can be used as the template for the definition file when enrolling or updating users, hosts and groups from an LDIF file.

The following figure shows the default LDIF definition file `ldif.sample.default.def`. There is a different template for applications.

```

# This is the definition file for an LDIF file enrollment.
# It assumes a default Active Directory schema.

enroll.users                      true
enroll.computers                   true

#
# Required attributes
#
computer.requirements             (objectClass=Computer)
user.requirements                 (&(objectClass=User) (! (objectClass=Computer)))
group.requirements                (objectClass=Group)
group.attributefor.enumeration    member

#
# Attribute mappings for user type
#
user.string.principalName         userPrincipalName
user.string.displayName            name
user.string.firstName              givenName
user.string.lastName               sn
user.string.windowsSid            objectSid

#
# Attribute mappings for host type
#
computer.string.dnsName          dnsHostName
computer.cs-string.windowsSid    objectSid

ldif.filename

```

Supply name
and location
of LDIF file here.
Note forward
slashes.

e:/p4_ajones_1.0/personal/ldifoutput.ldif

Figure 6: LDIF definition file template (ldif.sample.default.def)

LDIF definition file template (ldif.sample.default.def)

Normally you edit only the last element in this file, `ldif.filename`, which tells the enrollment utility which file to enroll. However, you must provide the full path to the LDIF file, and you must use front-slash characters rather than back slashes for the path separators.

This file is based on the default Active Directory schema. If the directory from which the LDIF file was exported differs from this default, edit the other elements in this file accordingly.

Related concepts

[The definition file](#) on page 84

The `-enroll -t LDIF` command requires a definition file.

dnsHostName requirement

All hosts in the network must have a `dnsHostName` attribute defined for them, regardless of whether they are being enrolled directly from an Active Directory or from an LDIF file.

Though in most cases all hosts have this attribute, it is not technically required by Active Directory; however, it is required for the Control Center enrollment. If any hosts do not have a `dnsHostName` defined, they are enrolled, but policies might not deploy properly or work as expected. If any hosts on the network do not have this attribute, edit the definition file, which is either `ad.default.def` or `ldif.default.def`, to avoid problems after enrollment. In the definition file, locate the following line: `computer.requirements (objectClass=Computer)`

and change it to the line:

```
computer.requirements (&(objectClass=computer) (dnsHostName=*)) )
```

After this change is made, the enrollment ignores any computers that do not have a dnsHostName, and otherwise proceeds normally.

Definition file for Domain Group enrollment

There is no difference in how you configure a Definition file for a Domain Group enrollment.

There are differences in how you reference Definition files, however:

- Do not point to the Definition file when running Enrollment Manager at the command line. Instead, reference Definition files for each Domain Group listed in the Configuration file.
- Rather than listing separate Definition files for each domain, you can also use a global Definition file for all subdomains within a Domain Group. In this case, you would enroll the same information from all listed subdomains.
- If you do specify a global Definition file, you can also specify exceptions to the global file, by listing a Definition file for individual subdomains.

Related concepts

[Configuration files for Domain Groups on page 58](#)

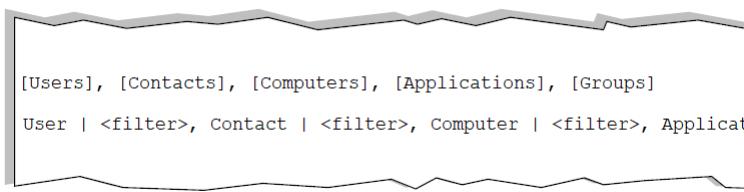
Domain Group enrollment requires an additional input file: the Configuration file.

Filter files

Filter files enable you to selectively import leaf element—users, contacts, hosts, applications, and groups—from a directory server.

If the directory being enrolled contains a large amount of data, you can create a filter file to selectively enroll only those users, hosts, applications, contacts, and groups that are relevant to policy enforcement requirements. To do this, create a filter file in comma-separated value (CSV) format. It is easiest to set up the filters using a spreadsheet program like Microsoft Excel and then save it as a CSV file, but you can also create the CSV file directly in any text editor. The following figure shows the LDAP filter file format.

 **Note:** A new filter file can be defined only during the initial enrollment of a given domain. After you enroll a domain using a filter file, you can use the same filter when you sync, but you cannot apply a different filter, or define a new one, when syncing an already enrolled domain.



```
[Users], [Contacts], [Computers], [Applications], [Groups]
User | <filter>, Contact | <filter>, Computer | <filter>, Application | <filter>
```

Figure 7: LDAP filter file format

The CSV file must use the format shown in the previous figure, where `<filter>` is a search specification in LDAP format. When placed in the `Users` column, the filter is applied within the set of all users in the directory; if under `Computers`, within all computers; and so on. The filter can be based on any attributes of objects, such as job titles of users, for example, (`title=Manager`). Here are some examples of filters that could be included in the `Users` column:

- `(lastName=M*)` Everyone whose last name begins with the letter M
- `(firstName=Jack)(firstName=Terry)`: Everyone whose first name is Jack OR Terry
- `!(firstName=Amy)` Everyone but Amy

As shown in the example above, the first row must contain the header cells, including square brackets:

```
[Users], [Contacts], [Computers], [Applications], [Groups]
```

The remaining rows list the following information:

- Under the [Users] header, each cell is either the login name of a user to be enrolled, or a filter in LDAP format, specifying which users to enroll from the set of all users in the directory.
- Under the [Contacts] header, each cell is either the contact of a user to be enrolled, or a filter in LDAP format, specifying which users to enroll from the set of all users in the directory.
- Under the [Computers] header, each cell is either the name of a computer, or a filter in LDAP format, specifying which computers to enroll from the set of all computers in the directory.
- Under the [Applications] header, each cell is either the name of an application, or a filter in LDAP format, specifying which application to enroll from the set of all applications in the directory.
- Under the [Groups] header, each cell is either the name of a group to be enrolled (for example, Hardware Upgrade Project), or a filter in LDAP format, specifying which groups to enroll from the set of all groups in the directory.

The same search filter syntax can be used in every column. The headers `Users`, `Contacts`, `Computers`, `Applications`, and `Groups` are provided so that you can omit filter criteria based on the `objectClass` or `objectCategory` attributes, since the type of object is already known from the column header. For example, to enroll all managers in an organization, you could use the filter (`title=Manager`) in a cell in the [Users] column. When the spreadsheet is complete, save it as a CSV file (with commas as the delimiters between cells).

For more information about the LDAP filter syntax, see the LDAP documentation and the proposed standard from the Internet Engineering Task Force.

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

[Elements of configuration files](#) on page 59

This section describes the elements of the configuration file for domain group enrollment.

Editing filter properties

This file contains three settings that control filtering behavior for hosts, users, and groups.

When using a filter, you might need to edit the content of a special file called `selectivefilter.sample.properties`, which is stored under the `\tools\enrollment` directory.

By default, the filter sample file specifies the entities in the header: users, computers, and groups. If all three filters are used, the properties file can be used without modification. However, to filter by only one or two of the entities, edit the properties file as described in this section.



Note:

Currently, there can only be one `selectivefilter.property` file to determine the format of all filter files used in a deployment. It must be named `selectivefilter.property` file.

1. Open the file with any text editor.
2. Comment out all three lines in the last section of the file, as shown in the following figure.
3. For the line or lines representing the entities you want to filter by, copy the first part of the line and paste it as a duplicate below the original, as shown in the following figure. The duplicate line should contain only the first element, but not the second (the actual LDAP filter, in parentheses). In the example, the file has been edited to filter by users and groups, but not hosts.
4. Save the file as `selective.filter.properties` in the `\tools\enrollment` directory.

```

#####
# Selection Filter Configuration Parameters
#####
.
.
.

# Get everything when no selections are made for a particular type:
[# all.valid.hosts.filter (&(objectCategory=Computer) (dNSHostName=*))
[# all.valid.users.filter (&(objectCategory=Person) (sAMAccountName=*))
    all.valid.users.filter
[# all.valid.groups.filter (&(objectCategory=Group) (sAMAccountName=*))
    all.valid.groups.filter

```

Comment out all three

Add duplicates of those you want to filter by

Figure 8: Selective Filter Properties File

After you have edited the filter properties in this way, you can proceed with the enrollment.

Configuration files for Domain Groups

Domain Group enrollment requires an additional input file: the Configuration file.

This file lists each subdomain included in the Domain Group, along with all necessary Connection, Definition, and Filter files for the subdomains. A template configuration file defaults the enrollment directory upon installation, named `domainGroup.sample.default.cfg`.

To use this template, copy it to a known location, make the changes necessary to configure the file for your environment, and then point to the file when you run enrollment manager (by filename and path) in the `-g <domaingroup>` parameter. You need to point to this file in both `-enroll` or `-update` commands.



Note: You should rename all these files to remove the `.sample` string from the names, just as with other input files.

Related concepts

[Using the Enrollment Manager](#) on page 44

The Enrollment Manager is automatically installed on the Control Center host.

[Definition file for Domain Group enrollment](#) on page 56

There is no difference in how you configure a Definition file for a Domain Group enrollment.

Related tasks

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Order of subdomains in configuration files

Enrollment Manager is designed to permit the subdomains of a Domain Group enrollment to be enrolled in any order.

The numbering of subdomains in the Configuration file (.0, .1, .2) controls the sync order. Subdomains must be numbered starting from .0, and numbers must be sequential (integers from .0 to .n). The following figure shows an example configuration file.

```

# Global files
definition.filename      C:/tmp/ad.default2.def

-----
# The start time and pull interval for automatic sync
# The format of ScheduledSyncTime must be format of "Oct 4, 2016 18:14 PM"
# The ScheduledSyncInterv is a positive number in minutes.
# Zero value of ScheduledSyncInterv means auto-sync is disabled
-----
ScheduledSyncTime          Oct 30, 2012 8:14 AM
ScheduledSyncInterv        1440

# Optional FilterFile
# FilterFile                {path.filter}

# Setup multiple enrollment in multi domain.

subdomain.0.type          DIR
subdomain.0.name            sdomain3.qalab01.nextlabs.com
subdomain.0.connection.filename C:/tmp/ad.sdomain0.conn
subdomain.0.definition.filename C:/tmp/ad.default0.def
subdomain.0.filter.filename c:/tmp/ADifilter.filter.txt

subdomain.1.type          LDIF
subdomain.1.name            ldif.qalab01.nextlabs.com
subdomain.1.definition.filename C:/tmp/ldif.sdomain1.conn

subdomain.2.type          PORTAL
subdomain.2.name            portal.qalab01.nextlabs.com
subdomain.2.connection.filename C:/tmp/portal.sdomain2.conn
subdomain.2.definition.filename C:/tmp/portal.sdomain2.conn

```

Figure 9: Example configuration file

Elements of configuration files

This section describes the elements of the configuration file for domain group enrollment.

Table 10: Elements of configuration files

Element	Description
definition.filename	Specifies the location of the global definition file used to establish what information should be drawn from the enrollment domain's schema. You can point to other definition files to be used in place of this one for individual subdomains (definition files listed for individual domains override the global definition file). Comment out this value to only specify definition files for individual subdomains.

Element	Description
ScheduledSync Time	<p>The scheduled sync time to begin the Domain Group enrollment. The sync time for a Domain Group should be defined here and not in the individual connection files for each subdomain. After the date and time specified here, the enrollment automatically synchronizes at the interval specified by the ScheduledSyncInterval parameter. The date and time value is locale-specific, and you must use Java's MEDIUM style for the date, and the SHORT style for the time. For example:</p> <ul style="list-style-type: none"> • For English (US): Sep 18, 2013 6:00PM • For English (UK):18-Sep-2013 18:00 • For the German (Germany):18.09.2013 18:00 <p>At enrollment time, the sync date must be in the future.</p>
ScheduledSync Interval	<p>For all subdomains listed in the configuration file, processing occurs one subdomain at a time, in the sequence of numbered subdomains <n>. The processing of each subsequent domain begins upon the completion of the previous domain.</p> <p>If a Domain Group includes an LDIF file, the file is synced automatically only if the LDIF file changes.</p>
Filter File	<p>The scheduled sync interval in between Domain Group enrollments. The sync interval for a Domain Group should be defined here and not in the individual connection files for each subdomain.</p> <p>The sync interval should be large enough for the processing of all subdomains to occur. To ascertain how long this interval should be, you can run a trial sync process, note the duration after the sync is complete, and then update the interval as appropriate.</p> <p>When left at the default, 0, disables the automatic synchronization feature. When changed to any positive value, specifies a regular re-synchronization schedule, expressed in minutes following the time specified by the ScheduledSyncTime.</p> <p>This setting has no influence until after the ScheduledSyncTime passes.</p>
subdomain.<n>.type	<p>If the Domain Group includes an LDIF file, it is automatically synced only if there are changes to the LDIF file.</p> <p>A filter file that should be used across all subdomains in the Domain Group. As with a global definition file, you can overwrite the default by specifying a different filter file for subdomains.</p>
	<p>The type of enrollment. Valid enrollment types are DIR, PORTAL, and LDIF. <n> should be replaced with an integer for the order of the enrollment, beginning with 0.</p>

Element	Description
subdomain.<n>.name	The name of the subdomain. <n> should be replaced with an integer for the order of the enrollment, beginning with 0.
subdomain.<n>.connection.filename	The location of the connection file for the subdomain. <n> should be replaced with an integer for the order of the enrollment, beginning with 0.
subdomain.<n>.definition.filename	The location of the definition file for the subdomain. <n> should be replaced with an integer for the order of the enrollment, beginning with 0. You can specify a global definition file (definition.filename). If you also specify a different definition file for a subdomain, it overrides the global definition file.
subdomain.<n>.filter.filename	Filter files are optional. You can supply a global filter file to apply to all subdomains in the Domain Group (Filter File above), or point to individual filter files for each subdomain.

Related concepts

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Definition files](#) on page 54

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

[Filter files](#) on page 56

Filter files enable you to selectively import leaf element—users, contacts, hosts, applications, and groups—from a directory server.

Enrolling users and hosts

This section includes procedures related to configuration options and for performing enrollments:

Related concepts

[Required entities](#) on page 37

Enrolling the following entities is required for all installations.

Configure a secure LDAP connection

After an enrollment is defined, a manual or automatic (scheduled) sync establishes a connection between the Control Center and the enrollment domain.

You can configure this connection to use standard SSL protocols for secure communication. The procedure consists of the following basic steps.

Related concepts

[About secure LDAP connections](#) on page 42

LDAP connections can be configured to use standard SSL protocols to establish secure communication between the Control Center and the LDAP Server.

Related tasks

[Configuring the certificates for LDAP servers](#) on page 62

Obtain and configure certificates for LDAP servers.

[Configuring connection files for SSL Secure Mode](#) on page 62

To use SSL for communications between the AD server and Control Center, add the SSL flag to connection files.

Related reference

[Elements of Active Directory connection files](#) on page 50

Configuring the certificates for LDAP servers

Obtain and configure certificates for LDAP servers.

These instructions assume you are using JAVA keytool.

1. You need to know the SSL certificate password that was defined during Control Center installation.
2. Obtain a certificate signed by the appropriate Certificate Authority (CA) for all LDAP servers for which you are configuring SSL Secure Mode.
3. Locate the certificate(s) on the Policy Server host in the following location: <install directory>/server/certificates.
4. Insert a certificate into the main truststore for the Control Center (dcc-truststore) using the following command, where <certificate_name.cer> is replaced with the name of the certificate being used for the LDAP server:

```
keytool -importcert -alias <alias name> -file <certificate_name.cer> -keystore dcc-truststore.jks
```

5. When prompted, enter the SSL certificate password.
6. Repeat the previous steps for all other certificates for LDAP servers.

Add the SSL flag to connection files.

Related concepts

[Configure a secure LDAP connection](#) on page 61

After an enrollment is defined, a manual or automatic (scheduled) sync establishes a connection between the Control Center and the enrollment domain.

Related tasks

[Configuring connection files for SSL Secure Mode](#) on page 62

To use SSL for communications between the AD server and Control Center, add the SSL flag to connection files.

[Configuring AD sources](#) on page 31

To import user information, you must configure the connection between the Control Center and the AD server.

Configuring connection files for SSL Secure Mode

To use SSL for communications between the AD server and Control Center, add the SSL flag to connection files.

Configure the certificates for LDAP servers.

1. Open the connection file for the LDAP server, change the port value from the default for LDAP (389) to 636.

 **Note:** This is the default port number for LDAPS. You can specify a different port number if required for your environment.

2. Ensure the line `secure.transport.mode SSL` is not commented out.

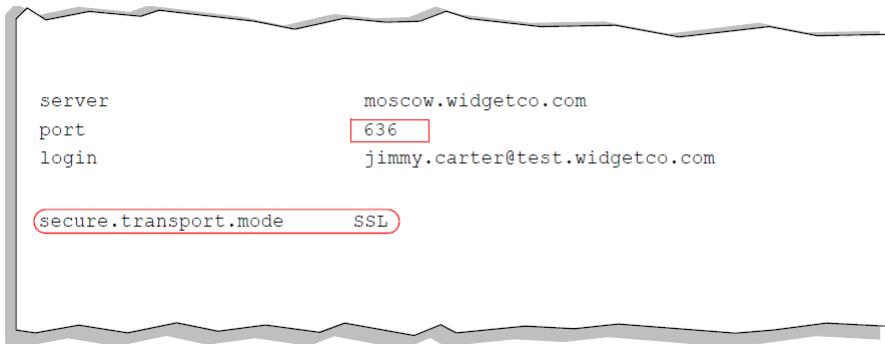


Figure 10: Configuring a connection file for LDAPS

3. Optional: You can add the a flag, always.trust.ad. When this flag is set to true, the Control Center server accepts any security certificate presented by the AD or LDAP server, including self-signed certificates. Otherwise, certificates must be signed by certificate authorities to be accepted.
4. Save the file.
5. When you are ready to perform the enrollment, do one of the following:
 - For a new single domain enrollment, reference the connection file in the command line.
 - For a new Domain Group enrollment, reference the connection file for the subdomain within the configuration file.
6. To update an existing enrollment (either single or Domain Group), run Enrollment Manager in update mode.

Related concepts

[Configure a secure LDAP connection](#) on page 61

After an enrollment is defined, a manual or automatic (scheduled) sync establishes a connection between the Control Center and the enrollment domain.

Related tasks

[Configuring the certificates for LDAP servers](#) on page 62

Obtain and configure certificates for LDAP servers.

How to enroll from a directory (single domain)

This section explains how to enroll from a single domain.

- Prepare all required input files.
- Ensure that your environment meets all requirements.

In addition, you need to have the following information to use the enrollment utilities:

- The domain name containing the objects to be enrolled. This is also used as a unique name assigned to this enrollment.
- The hostname of the Control Center, and its port number, which is 8443 by default. In many cases this is the same host where you are running command window, but you may be running the enrollment remotely.
- The username and password of an authorized Control Center administrator.
- The name of the connection and definition files you have prepared for this enrollment.
- (Optional) The name of a filter file.

1. On the Control Center host, open a console window.
2. Change to the directory where the Enrollment Manager is installed:
`\Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the Enrollment Manager using the following command, filling in the appropriate values for the arguments:

```
enrollmgr -enroll -t DIR -n <domain_name> -s <server> -p <port> -u  
<username> {-w <passwd> | -ew <encryptpwd>} -a <dir_connection_file> -d  
<definitionfile> [-f <filterfile>]
```

 **Note:** The connection parameters you specify in the command line here—domain, server, port, user and password—refer to the Control Center server, not the LDAP server. That information is provided by the connection file.

When this utility runs, it reads the connection file on the Control Center host, connects to the LDAP server and reads the entries in the LDAP directory, transforms them into the format Control Center requires, and performs the enrollment.

When this step is complete, a success message appears.

4. (Optional) To review your progress so far, you can see a list of enrollments you have created by running the command:

```
enrollMgr -list -s <server> -p <port> -u <username> {-w <passwd> | -ew  
<encryptpwd>}
```

5. To enroll data from multiple domains, repeat step 3 for each domain, using the domain name of each for the `-n` argument.
6. After each enrollment, the `-sync` process is still required to synchronize the data in the enrollment. This is because the enrollment itself only imports metadata and sets up the required data structures in the Information Network Directory; the structures are populated only during the synchronization procedure.

Related concepts

[About enrollment input files](#) on page 47

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

[Before you enroll](#) on page 43

Before enrolling LDAP data, you must run the Control Center installation wizard, and then start Control Center.

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Enrolling from an LDIF file for a single domain

Use a different template for the definition file, called `app.sample.default.def` to enroll.

- Prepare all required input files.
- Ensure that your environment meets all requirements.

In addition, you need to have the following information to use the enrollment utilities:

- The domain name, which is the unique name assigned to this enrollment.
- The hostname of the Control Center, and its port number, which is 8443, by default. In many cases this is the same host where you are running command window, but you may be running the enrollment remotely.
- The username of an authorized Control Center administrator.
- The utility security password.

- The name of the definition file you have prepared for this enrollment.

After each enrollment, the `-sync` process is still required to synchronize the data in the enrollment. This is because the enrollment itself only imports metadata and sets up the required data structures in the Information Network Directory; the structures are populated only during synchronization procedure.

1. On the Control Center host, open a console window.
2. Change to the directory where the Enrollment Manager is installed—by default:
`\Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the `-enroll` command, filling in the appropriate values for the arguments:
`enrollmgr -enroll -t LDIF -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> | -ew <cryptpwd>} -d <definitionfile>`
When this utility runs, it reads the LDIF file specified in the definition file, transforms its contents into the format Control Center requires, and performs the enrollment.
When this step is complete, a success message displays.
4. (Optional) To review your progress so far, you can see a list of enrollments you've created by running the command:
`enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> | -ew <cryptpwd>}`
5. After each enrollment, the `-sync` process is still required to synchronize the data in the enrollment. This is because the enrollment itself only imports metadata and sets up the required data structures in the Information Network Directory; the structures are populated only during synchronization procedure.

Related concepts

[About enrollment input files](#) on page 47

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

[Before you enroll](#) on page 43

Before enrolling LDAP data, you must run the Control Center installation wizard, and then start Control Center.

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

[Enrolling applications](#) on page 83

Applications, unlike users, cannot be automatically enrolled from LDAP directories.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Enrolling Domain Groups

There are a few concepts and requirements unique to Domain Group enrollment.

You should be familiar with the following information before performing this procedure:

- A Domain Group can include enrollment types DIR, PORTAL, and LDIF.
- Rather than pointing to connection, definition, and filters files during the command line procedure, you list input files for each subdomain in the configuration file.

- In the Configuration file, all subdomains are listed in numerical order, starting with 0; the order of the subdomains does not matter because data interdependencies across subdomains are resolved after all subdomains have been processed.
- For subdomains listed in the Configuration file, there should be no gaps in the number sequence, from subdomain.0 to subdomain.n.
- You can use a global definition file for all subdomains listed in the configuration file; this would mean that the same information is being retrieved from each subdomain. Conversely, you can omit a global definition file and use a distinct definition file for each subdomain. A third option is you can include a global definition file, then overwrite it with definition files specified for individual subdomains.
- For DIR and PORTAL type enrollments, you should specify the sync time and interval in the configuration file, not the connection file(s). LDIF files that are included in a Domain Group are re-synced only if the LDIF file has been updated since the last scheduled sync. Otherwise, the LDIF file is skipped as Enrollment Manager processes sequential subdomains.
- Bundle building for Domain Groups requires you to configure trusted domain relationships between subdomains.

After you have assembled all the input files and have ensured your system meets requirements, you can enroll Domain Group.

- Prepare all required input files.
- Ensure that your environment meets all requirements.

In addition, you need to have the following information to use the enrollment utilities:

- The domain name, which is the unique name assigned to this enrollment.
- The hostname of the Control Center, and its port number, which is 8443, by default. In many cases this is the same host where you are running command window, but you may be running the enrollment remotely.
- The username of an authorized Control Center administrator.
- The utility security password.
- The name of the configuration file you have prepared for this enrollment.

1. On the Control Center host, open a console window.

2. Change to the directory where the Enrollment Manager is installed—by default:

`\Program Files\NextLabs\PolicyServer\tools\enrollment`

3. Run the `-enroll` command, filling in the appropriate values for the arguments:

```
enrollmgr -enroll -s <server> -p <port> -n <domain_name> -t DOMAINGROUP  
-g <domain_group configuration file> -u <username> {-w <passwd> | -ew  
<encryptpwd>}
```

When the utility runs, it begins with the first subdomain. If that domain is a DIR or PORTAL, Enrollment Manager reads the connection file, connects to the LDAP or SharePoint server, reads the target data to be enrolled, transforms the data into the format Control Center requires, and performs the enrollment. If the enrollment type is LDIF, Enrollment Manager reads the LDIF file specified in the definition file, transforms its contents into the format Control Center requires, and performs the enrollment.

 **Note:** If a subdomain is already enrolled, it is updated during the enrollment of the Domain Group. If it did not exist prior to the creation of the Domain Group, it is enrolled. In this way, you can add a subdomain to an existing Domain Group.

4. After each enrollment, the `-sync` process is still required to synchronize the data in the enrollment. This is because the enrollment itself only imports metadata and sets up the required data structures in the Information Network Directory; the structures are populated only during synchronization procedure.

Related concepts

[About Domain Group enrollment](#) on page 39

A Domain Group contains multiple User, Host, and Group sources, referred to as subdomains, that can be managed together in a single enrollment process.

[Configure connection files for Domain Group enrollment on page 50](#)

Connection file configuration is similar for both single and Domain Group enrollments, with two important differences.

[Configuration files for Domain Groups on page 58](#)

Domain Group enrollment requires an additional input file: the Configuration file.

[Configuring trusted domains on page 139](#)

During normal operation, Control Center tracks users and hosts according to the domain where they are enrolled, for the purposes of preparing policy bundles. Control Center needs this information so that it can include, in policy bundles, only the information that is relevant to a particular policy enforcement point.

[About enrollment input files on page 47](#)

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

[Before you enroll on page 43](#)

Before enrolling LDAP data, you must run the Control Center installation wizard, and then start Control Center.

[Synchronizing enrollments on page 73](#)

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

[Scheduled synchronization on page 77](#)

You can configure an enrollment to be automatically synchronized on a regular schedule. This is recommended for all production implementations.

How to maintain enrollments

Once you have enrolled users, groups and hosts into your system, you need to keep the information about them up to date.

To help with this, Control Center provides utilities for updating and synchronizing enrollments.

Updating vs. synchronizing

Because the enrollment name is the same as the domain or Domain Group name, you can only perform one enrollment from each domain or Domain Group (again, the term “domain” refers to the location where enrollment information is stored, not a network domain).

If you later want to change the entities enrolled from an existing domain or Domain Group—for example, add new entity types or different roots, or delete a subdomain from a Domain Group—you must perform an `-update`. This command changes the data structure of the enrollment only; it does not import the data itself. For this reason, you should run a `-sync` command after all updates.

If you only need to refresh the enrolled data without redefining the structure of the enrollment—that is, leaving the entity types and other metadata the same—you use the `-sync` command. (This is why the `-update` command includes the definition file parameter, while the `-sync` command does not.)

The following table summarizes the various options available for updating and synchronizing your enrollments. Details on all procedures are provided in the following sections.

Table 11: User, host and group synchronization options

	Update	Manual sync	Scheduled sync
LDAP Directory	<p>YES: Run enrollMgr - update with Type=DIR, and specify the enrollment name. Both definition and connection files are required. Also, you must run the Sync procedure after updating.</p> <p>Effect: The data structure for this enrollment is replaced in the Information Network Directory, with the metadata from the directory specified in the connection file, according to the syntax specified in the definition file. Update does not import the data itself; this is why you must run a Sync after updating.</p>	<p>YES: Run enrollMgr -sync, and specify enrollment name. No auxiliary files are required.</p> <p>Effect: Based on the enrollment name, Control Center retrieves the connection and definition information from its internal tables, and synchronizes its Information Network Directory data to the data in the specified enrollment domain or subdomains (in the case of a Domain Group). No change can be made to the structure of the entities. Any changed information is updated, any entities that have been removed from the directory are removed from the IND, and any entities that have been added to the directory are added to the IND.</p>	<p>YES: Sync time and interval are based on the last two parameters in the definition file.</p>

	Update	Manual sync	Scheduled sync
LDIF File	<p>YES: Run enrollMgr -update with type=LDIF and specify enrollment name. The definition file is required, and you must run a Sync command after the update.</p> <p>Effect: The data structure for this enrollment is replaced in the Information Network Directory, by the metadata from the LDIF file specified in the definition file, according to the syntax specified in the definition file.</p>	<p>YES: Run enrollMgr -sync, and specify enrollment name. No auxiliary files are required.</p> <p>Effect: Based on the enrollment name, Control Center retrieves the LDIF filename and path from its internal tables, and synchronizes its Information Network Directory data to the contents of the LDIF file. Any changed information is updated, any entities that are no longer present in the LDIF file are removed from the IND, and any that have been added to the file are added to the IND.</p>	<p>NO if a single domain enrollment: For single domain enrollments, the scheduled sync depends on a connection file, which LDIF File enrollments do not use. You should manually update the enrollment procedure using enrollMgr update, with type=DIR.</p> <p>YES if part of a DOMAIN GROUP: For Domain Group Enrollments, an LDIF file may be listed in a configuration file alongside other kinds of enrollment types. When a scheduled sync occurs, Enrollment Manager detect whether the LDIF file has been updated since the last sync. If there is new information in the LDIF file, it is synced at the scheduled time.</p> <p>Effect: Is the LDIF file has been updated and not synced, Control Center retrieves the LDIF filename and path from its internal tables, and synchronizes its Information Network Directory data to the contents of the LDIF file. Any changed information is updated, any entities that are no longer present in the LDIF file are removed from the IND, and any that have been added to the file are added to the IND.</p>

Updating directory enrollments for a single domain

You should use the –update command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

This requires a definition file, where the different entities are specified. The update process imports new metadata from the source into CE; to import the data itself, you must perform a manual sync after the update.

To update user, group, and host data from an LDAP directory, perform the following steps.

You need to have the following information to use the enrollment utilities:

- The unique name of the enrollment you are updating, which is the same as the domain name.
- The hostname of the Control Center, and its port number (8443, by default). In many cases this is the same host where you are running command window, but you may be running the enrollment remotely.
- The username of an authorized Control Center administrator.
- The utility security password.
- The name of the connection and definition files you have prepared for this enrollment.

1. (Optional): The name of a filter file.

2. On the Control Center host, open a console window.

3. Change to the directory where the Enrollment Manager is installed—by default, `\Program Files\NextLabs\PolicyServer\tools\enrollment`

4. At the prompt, run the `-update` command as follows, filling in the appropriate values for the arguments:

```
enrollmgr -update -t DIR -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> | -ew <encryptpwd>} -a <ad_connection_file> -d <definitionfile> [-f <filterfile>]
```

When this utility runs, it reads the directory specified in the connection file, transforms its contents into the format Control Center requires, and performs the enrollment.

When this step is complete, a success message appears.

When the update is finished, the data structure in Control Center has been updated, but no new data content has been imported. To do that, you have to run the `-sync` command as a separate operation.

Related concepts

[About enrollment input files](#) on page 47

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

Related tasks

[Adding properties](#) on page 101

You can use the `-add` command to add properties to users, hosts, and applications.

[Removing deleted LDAP users or groups from an enrollment](#) on page 72

After deleting users or groups from an LDAP directory, the deleted users or groups are marked as inactive in the enrollment.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Updating LDIF file enrollments for a single domain

You can use the `-update` command to update user and host information in the Control Center Information Network Directory, based on the content of an LDIF file.

This command updates metadata only; the `-sync` command updates the data itself.

You need to have the following information to use the enrollment utilities:

- The unique name of the enrollment you are updating, which is the same as the domain name.

- The hostname of the Control Center, and its port number, which is 8443, by default. In many cases this is the same host where you are running the command window, but you may be running the enrollment remotely.
- The username of an authorized Control Center administrator.
- The utility security password.
- The name of the definition file you have prepared for this enrollment.

If you want to remove deleted users in LDAP from the enrollment rather than marking them as inactive, see the [Removing deleted LDAP users or groups from an enrollment](#) section.

1. On the Control Center host, open a console window.
2. Change to the directory where the Enrollment Manager is installed—by default, `\Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the `-update` command as follows, filling in the appropriate values for the arguments:

```
enrollmgr -update -t LDIF -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> | -ew <cryptpwd>} -d <definitionfile>
```

When this command runs, it reads the specified LDIF file, transforms its contents into the format Control Center requires, and performs the enrollment.

When this step is complete, a success message appears.

When the update is finished, the data structure in Control Center has been updated, but no new data content has been imported. To do that, run the `-sync` command as a separate operation.

Related concepts

[About enrollment input files](#) on page 47

The command lines for enrolling and updating users, hosts and groups have several filename parameters, which refer to the following kinds of auxiliary input files.

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

Related tasks

[Removing deleted LDAP users or groups from an enrollment](#) on page 72

After deleting users or groups from an LDAP directory, the deleted users or groups are marked as inactive in the enrollment.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Updating Domain Group enrollments

If you make changes to any of the input files for the subdomains for an existing Domain Group enrollment, or add or delete subdomains for an enrolled Domain Group, you need to update the enrollment.

If you want to remove deleted groups in LDAP from the enrollment rather than marking them as inactive, see the [Removing deleted LDAP users or groups from an enrollment](#) section.

1. Make relevant changes to all input files and the Domain Group configuration file, as required.
2. On the Control Center host, open a console window.
3. Change to the directory where the Enrollment Manager is installed—by default, `\Program Files\NextLabs\PolicyServer\tools\enrollment`
4. At the prompt, run the `-update` command as follows, filling in the appropriate values for the arguments:

```
enrollmgr -update -s <server> -p <port> -n <Domain Name> -t DOMAINGROUP -g <DomainGroup configuration file> -u <username> {-w <passwd> | -ew <cryptpwd>}
```

When this utility runs, it starts processing the first subdomain, reads the information from the connection file, transforms its contents into the format Control Center requires, and performs the enrollment. For descriptions of all command line arguments.

When this step is complete, a success message displays.

When the update is finished, the Control Center data structure has been updated, but no new data content has been imported. To do that, you have to run the `-sync` command as a separate operation.



Note: A subdomain type cannot be changed through an `-update` alone. For example, you cannot change a configuration file to for a subdomain from type DIR, to type Portal, and then run the update. Instead, you must delete the existing subdomain, change the type for the subdomain in the configuration file, and then run the update with the new subdomain type. The new subdomain with the adjusted type is created during the next sync.

Related concepts

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

[Delete Domain Group enrollments](#) on page 79

The process for deleting enrollments for a Domain Group is a little more complex than for single domain enrollments.

Related tasks

[Removing deleted LDAP users or groups from an enrollment](#) on page 72

After deleting users or groups from an LDAP directory, the deleted users or groups are marked as inactive in the enrollment.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Removing deleted LDAP users or groups from an enrollment

After deleting users or groups from an LDAP directory, the deleted users or groups are marked as inactive in the enrollment.

If you want to remove deleted users or groups from the enrollment rather than marking them as inactive, follow these steps.

1. Open the definition file using any text editor.
2. Add the following line:

```
delete.inactive.group.members=true
```

Note: By default, the `delete.inactive.group.members` parameter is set to false.

3. Save and close the definition file.
4. Run the `update` command.

Related tasks

[Updating LDIF file enrollments for a single domain](#) on page 70

You can use the `-update` command to update user and host information in the Control Center Information Network Directory, based on the content of an LDIF file.

[Updating Domain Group enrollments](#) on page 71

If you make changes to any of the input files for the subdomains for an existing Domain Group enrollment, or add or delete subdomains for an enrolled Domain Group, you need to update the enrollment.

[Updating directory enrollments for a single domain](#) on page 69

You should use the `-update` command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

Synchronizing enrollments

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

To do this, you can use the `-sync` command, which synchronizes one or more enrollments with their originating domains in the LDAP directory. By using this command, you can keep the Control Center Information Network Directory up to date with changes in your organization. Bear in mind you can only synchronize the same data you have already enrolled; if you need to change the structure of the entities enrolled from a domain, you need to use the `-update` command.

You can perform manual synchronizations for either directory- or file-based enrollments, but you can schedule automatic synchronizations only for directory-based ones. If you have enrolled AD data from more than one domain, you can set both to sync automatically, but they must be scheduled to run at different times—the sync processes must not overlap.

If the Control Center encounters any problem that prevents it from completing a sync procedure, it sends a notification email message to the default address defined in the config file's `<MessageHandlers>` section. This message contains details about the problem, for troubleshooting purposes.

Related tasks

[How to enroll from a directory \(single domain\)](#) on page 63

This section explains how to enroll from a single domain.

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

[Updating directory enrollments for a single domain](#) on page 69

You should use the `-update` command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

[Updating LDIF file enrollments for a single domain](#) on page 70

You can use the `-update` command to update user and host information in the Control Center Information Network Directory, based on the content of an LDIF file.

[Updating Domain Group enrollments](#) on page 71

If you make changes to any of the input files for the subdomains for an existing Domain Group enrollment, or add or delete subdomains for an enrolled Domain Group, you need to update the enrollment.

[Adding properties](#) on page 101

You can use the `-add` command to add properties to users, hosts, and applications.

Manual synchronization

The `-sync` command reuses the parameters provided the last time the `enroll` command was called for the specified enrollments—including the references to the appropriate definition and connection files—based on the enrollment name parameter (`-n`).

The `-sync` command is the same for single domain or Domain Group enrollments. When you manually sync subdomains, you supply the name of the subdomain as the Domain name (`-n` below). For LDAP enrollments, the connection file provides the information for the LDAP directory you want to sync to; for LDIF files, the definition file contains the required file and path information.

If you need to change the parameters for any of the enrollments (for example, to change the security password), use the `-enroll` command again with the desired enrollment name, using the new parameters.

You need to have the following information to use the enrollment utilities:

- The unique name of the enrollment you are updating, which is the same as the domain name.

- The hostname of the Control Center, and its port number, which is 8443, by default. In many cases this is the same host where you are running command window, but you may be running the enrollment remotely.
 - The username of an authorized Control Center administrator.
 - The utility security password.
- On the Control Center host, open a console window.
 - Change to the directory where the Enrollment Manager is installed—by default, <installation_directory>\Nextlabs\PolicyServer\tools\enrollment
 - At the prompt, run the `-sync` command, filling in the appropriate values for the arguments:

```
enrollmgr -sync -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> | -ew <encryptpwd>}
```

When this utility runs, it reads either the LDAP directory or the LDIF file on which the specified enrollment is based (the connection and file information is stored in Control Center's internal tables), and refreshes all relevant information in the Network Information Directory.

When this step is complete, a success message appears.
 - (Optional) To review your progress so far, you can see a list of enrollments you have created by running the command:

```
enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> | -ew <encryptpwd>}
```

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Incremental synchronization using LDIF delta file

If you want make changes to an existing enrollment after performing manual synchronization, you can use the Enrollment Manager utility with the `-delta` option to perform incremental synchronization.

To add, delete, or update users, perform the following steps:

- Create an LDIF delta file that contains changes to an existing enrollment (add, delete, or update users) and add a `changetype` attribute (see the following table) for each entry in the LDIF file to indicate that the file is an LDIF delta file.

Table 12: changetype attributes

Operation	Changetype Attribute	Example of an entry in an LDIF Delta File
Add	<code>changetype:add</code>	<p>Example of an LDIF delta file entry to add a new user TEST009.</p> <pre>dn: cn=TEST009,dc=sap,dc=nextlabs,dc=com changetype:add objectclass: user cn: TEST009 Country: US Department: Purchasing Full Name: TEST009 First Name: TEST009 Last Name: TEST009 user Principal Name: TEST009@sap.nextlabs.com SAPUserId: TEST009 objectGUID: 010</pre>

Operation	Changetype Attribute	Example of an entry in an LDIF Delta File
Delete	changetype:delete	<p>Example of an LDIF delta file entry to delete a user TEST008.</p> <pre>dn: cn=TEST008,dc=sap,dc=nextlabs,dc=com changetype:delete objectclass: user cn: TEST008 Country: US Department: Purchasing Full Name: TEST008 First Name: TEST008 Last Name: TEST008 user Principal Name: TEST008@sap.nextlabs.com SAPUserId: TEST008 objectGUID: 009</pre>
Update	changetype:add (change attributes of the existing enrolled users)	<p>Example of an LDIF delta file entry to update department from Purchasing to Finance of the user TEST009. dn:</p> <pre>dn: cn=TEST009,dc=sap,dc=nextlabs,dc=com changetype:add objectclass: user cn: TEST009 Country: US Department: Finance Full Name: TEST009 First Name: TEST009 Last Name: TEST009 user Principal Name: TEST009@sap.nextlabs.com SAPUserId: TEST009 objectGUID: 010</pre>

2. Do one of the following:

- Update the existing `delta_ldif_com.def` file:
 - a. Using any text editor, open the `delta_ldif_com.def` file.
 - b. Replace the existing LDIF file entry with the LDIF delta file entry. For example:
`ldif.filename C:/Program Files/Nextlabs/PolicyServer/tools/enrollment/<LDIF_delta_file>.ldif`
 - c. Save the `delta_ldif_com.def` file.
 - Update the existing LDIF file:
 - a. Back up the existing LDIF file by copying the file to a secure location.
 - b. Remove all the data in the LDIF file.
 - c. Copy the data from the LDIF delta file and paste it in the existing LDIF file.
 - d. Save the LDIF file.
3. On the Control Center host, open a console window.
4. Change to the directory where the Enrollment Manager is installed—by default, `<installation_directory>\Nextlabs\PolicyServer\tools\enrollment`
5. (Only if you have updated the existing `delta_ldif_com.def` file) Run the `-update` command to update the LDIF file. For example:

```
enrollmgr -update -t LDIF -u Administrator -n delta_ldif -d
delta_ldif_com.def {-w <passwd> | -ew <cryptpwd>}
```

 **Note:** If you have update an existing LDIF file, you do not need to run the -update command before enabling incremental synchronization.

- At the prompt, run the Enrollment Manager using the following command, filling in the appropriate values for the arguments:

```
enrollmgr -sync -delta -n <domain_name> -s <server> -p <port> -u <username>
{-w <passwd> | -ew <cryptpwd>}
```

The following table lists different scenarios and the corresponding results while using the -delta option.

Table 13: Scenarios while using the -delta option

Are you specifying the -delta option?	Are you providing an LDIF delta file?	Result
Yes	Yes	The system performs synchronization with incremental changes.
Yes	No	<p>The sync operation fails and the system displays the following error message:</p> <p>File is not in the delta format. Remove the -delta option from the command and sync again.</p> <p>You can either provide a delta file or modify the command and perform the sync operation again.</p>
No	Yes	<p>The system performs synchronization without incremental changes. This may result in loss of data (by making the existing data inactive) and only add new entries that will be active.</p> <p>A warning message is logged in the enrollment log file. If you want to perform incremental synchronization, use the -delta option and run the command again.</p>
No	No	The system performs synchronization without incremental changes.

When this utility runs, it reads either the LDAP directory or the LDIF file on which the specified enrollment is based (the connection and file information is stored in Control Center's internal tables), and refreshes all relevant information in the Network Information Directory.

When this step is complete, a success message appears.

- (Optional) To review your progress so far, you can see a list of enrollments you have created by running the command:

```
enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> | -ew
<cryptpwd>}
```

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's -list command at any time to check which domains have been enrolled.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Scheduled synchronization

You can configure an enrollment to be automatically synchronized on a regular schedule. This is recommended for all production implementations.

To set this up, simply provide a value for the last two parameters in the connection file (for single domain enrollment) or the configuration file (for Domain Group enrollment):

- `ScheduledSyncTime`: The date and time when the first autosynchronization is to be performed.
- `ScheduledSyncInterval`: The interval, in minutes, at which the synchronization is to occur, after the time specified in the `ScheduledSyncTime` parameter. A zero value here disables the autosync feature.

This option is available only for enrollments from LDAP directories, not from LDIF files. For those, you can perform a manual synchronization, using the `enrollMgr -sync -t LDIF` command.

Scheduled synchronizations should be done frequently, to ensure that your Control Center policies are consistent with the actual network entities. As a rule, you should configure this to run at least every 24 hours (`ScheduledSyncInterval = 1440`).

-  **Note:** You cannot synchronize more than one domain (that is, more than one Active Directory) at the same time. If you attempt to do this, one of the synchronizations fails. If you have enrollments from multiple domains, be sure to schedule the automatic sync operations so that they do not overlap.

Related concepts

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Connection file details](#) on page 49

The connection file is a template that contains a number of commented lines explaining how to use the elements.

Related tasks

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

Scheduled synchronizations for Domain Group enrollments

Enrollment Manager synchronizes domains one at a time. When you schedule synchronization intervals for a Domain Group, you should allow enough time for all subdomains to be processed.

This avoids overlapping processes, which are inefficient and can impact system performance.

NextLabs recommends that you run a trial synchronization when you first enroll a Domain Group and note the time it takes to complete. You can use this information to determine the proper interval to avoid overlap between scheduled syncs. To update the sync times, modify the configuration file, run the `-update` command to update the enrollment, and then run a `-list` to confirm the changes.

Managing enrollment tasks

This section describes how to use Enrollment Manager commands to perform managerial tasks.

Related tasks

[Checking enrollments](#) on page 78

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

[Deleting single domain enrollments](#) on page 79

To delete an enrollment from your Control Center system for any reason, you can use the Enrollment Manager's `-delete` command.

[Accessing enrollment help \(usage\)](#) on page 80

Use the `enrollmgr -h` command to display a list of all valid commands and their arguments. The command `-help` is also supported.

Checking enrollments

You can use the Enrollment Manager's `-list` command at any time to check which domains have been enrolled.

This command generates a list of all current enrollments that were created with the `-enroll` command; each of these represents one domain. The list does not include any domains that were enrolled but later dropped using the `-delete` command.



Note: This command returns a list of enrollments, not of the network objects (users, PCs, servers, and so on.) actually imported.

1. On the host where the Management Server component of the Control Center is installed, open a console window.
2. Change to the directory `<InstallDir>\tools\enrollment`. By default, this is: `\Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the `-list` command as follows, filling in the appropriate values for the arguments:

```
enrollmgr -list -s <server> -p <port> -u <username> {-w <passwd> | -ew <cryptpwd>}
```

This returns a list of all enrollments, with the following information:

- Name of the enrollment.
- Type of enrollment (Domain Groups and subdomains both display in the list).
- Source of the enrolled data.
- A (filtered) tag indicates that the data was enrolled using an inclusive filter.
- A (pending) tag indicates that the enrollment is set to occur the next time `synchronize.bat` runs; that is, `enroll.bat` was called with the argument `-z TRUE`.
- Date when the enrollment was last updated.

Related concepts

[Managing enrollment tasks](#) on page 77

This section describes how to use Enrollment Manager commands to perform managerial tasks.

[Filter files](#) on page 56

Filter files enable you to selectively import leaf element—users, contacts, hosts, applications, and groups—from a directory server.

Related tasks

[How to enroll from a directory \(single domain\)](#) on page 63

This section explains how to enroll from a single domain.

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

[Manual synchronization](#) on page 73

The `-sync` command reuses the parameters provided the last time the `enroll` command was called for the specified enrollments—including the references to the appropriate definition and connection files—based on the enrollment name parameter (`-n`).

[Incremental synchronization using LDIF delta file](#) on page 74

If you want make changes to an existing enrollment after performing manual synchronization, you can use the Enrollment Manager utility with the `-delta` option to perform incremental synchronization.

Deleting single domain enrollments

To delete an enrollment from your Control Center system for any reason, you can use the Enrollment Manager's `-delete` command.

This command removes the enrollment entry from the Information Network Directory tables, so that no more updates or syncs can be performed on it. Thus if the enrollment was set up to be automatically synchronized, `-delete` cancels that feature.

An important case where you would need to use the delete command, is any time a single enrollment fails and you need to perform it again. In such cases, the failed enrollment must be manually deleted before any subsequent attempt can succeed.

You should run the `-delete` command from wherever the Enrollment Manager is installed—by default, `<installDirectory>\ControlCenter\tools\enrollment`. You should run it from that location.

1. On the host where the Management Server component of the Control Center is installed, open a console window.
2. Change to the directory `<InstallDir>\tools\enrollment`. By default, this is: `\Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the `-delete` command as follows, filling in the appropriate values for the arguments:

```
enrollmgr -delete -n <domain_name> -s <server> -p <port> -u <username> {-w <passwd> | -ew <encryptpwd>}
```

When the enrollment has been deleted, a confirmation message appears.

4. To verify the deletion, use the `-list` command.

After you have deleted an enrollment, any policy components based on the objects in that enrollment no longer work. For this reason, if you need to change the contents of an enrollment, it is preferable to use Enrollment Manager to update and then re-sync it, rather than deleting altogether.

Related concepts

[Managing enrollment tasks](#) on page 77

This section describes how to use Enrollment Manager commands to perform managerial tasks.

Delete Domain Group enrollments

The process for deleting enrollments for a Domain Group is a little more complex than for single domain enrollments.

For instance, you can delete subdomains from a Domain Group enrollment, while leaving the Domain Group and the rest of the subdomains intact. You could also delete an entire Domain Group along with all its associated subdomains. This section describes these procedures.

Related tasks

[Updating Domain Group enrollments](#) on page 71

If you make changes to any of the input files for the subdomains for an existing Domain Group enrollment, or add or delete subdomains for an enrolled Domain Group, you need to update the enrollment.

Deleting a subdomain from a Domain Group

Before deleting a subdomain, ensure that information in that subdomain is not referenced by another subdomain. When a source subdomain is deleted, the related element in the related domain may not be enrolled properly.

Perform the following steps to delete a subdomain from a Domain Group.

1. Run the `-delete` command for the enrollment name (`-n`) of the subdomain.
2. Remove the subdomain from the configuration file.
3. Run an `-update` to update the new configuration file in the Control Center.

Upon the next `-sync`, enrollment manager references the new configuration file and does not attempt to update the deleted subdomain. If the `-update` step is skipped, and the configuration file still contains a deleted subdomain, Enrollment Manager attempts to create the new enrollment.

Deleting a Domain Group

To properly delete an entire Domain Group (meaning all the top-level group as well as all subdomains), follow these steps.

1. Run the `-delete` command for the enrollment name (`-n`) of all subdomains for the domain group.
Each must be deleted individually.
2. Run the `-delete` command for the enrollment name (`-n`) of the Domain Group.

Accessing enrollment help (usage)

Use the `enrollmgr -h` command to display a list of all valid commands and their arguments. The command `-help` is also supported.

Related concepts

[Managing enrollment tasks](#) on page 77

This section describes how to use Enrollment Manager commands to perform managerial tasks.

Chapter

5

Enrolling other entities

Topics:

- [Enrolling applications](#)
- [Enrolling file shares](#)

This section describes how to enroll applications, file shares, and sites.

Related concepts

[About connection files](#) on page 48

Connection files, as the name implies, contain the information Control Center needs to locate and connect to an information source: either to the LDAP directory tree from which information is to be enrolled, or to the SharePoint server whose groups you want to enroll.

[Enrolling file shares](#) on page 86

Enrolling information about file shares ensures that the use of differing file share paths to access the same document does not result in inconsistent policy enforcement.

[Enrolling SharePoint groups](#) on page 91

The SharePoint Enforcer is available exclusively with the Entitlement Management product. If you are not using that product, you can ignore this section.

[Enrollment utility password requirements](#) on page 97

Some enrollment utilities require users to enter the password of the super user Administrator account, which is initially configured during Control Center installation. This password can be changed as needed.

Related tasks

[Enrolling applications](#) on page 83

Applications, unlike users, cannot be automatically enrolled from LDAP directories.

[Enrolling location sites](#) on page 89

A location site is a Control Center term referring to a group of hosts that you want to consider as a single location because they share certain characteristics.

Enrolling applications

Applications, unlike users, cannot be automatically enrolled from LDAP directories.

To be able to define application-type components, applications must be enrolled into the Control Center Information Network Directory manually. After applications are enrolled, they become available in the lookup list of applications and they can be used to define application components.

Applications can be enrolled into the information network directory using a special utility called App Discovery. This utility generates an LDIF file containing all the required fields for the specified applications. The LDIF file is then enrolled into the Control Center Information Network Directory using Enrollment Manager's standard -enroll -t LDIF and -sync commands.

Normally this action is performed on a PC or laptop on which all the applications to be monitored are installed. One convenient option is to use a standard PC image that system administrators use for setting up desktops.

1. Locate the AppDiscovery folder from the Support folder in the Control Center installation directory.
2. Copy the AppDiscovery folder to any host location where you want to search for applications to enroll.
3. At the command prompt, run the following command:

<LocationCopiedTo>\appDiscovery>appdiscovery.cmd

The Application Discovery window appears, as shown in the following figure.

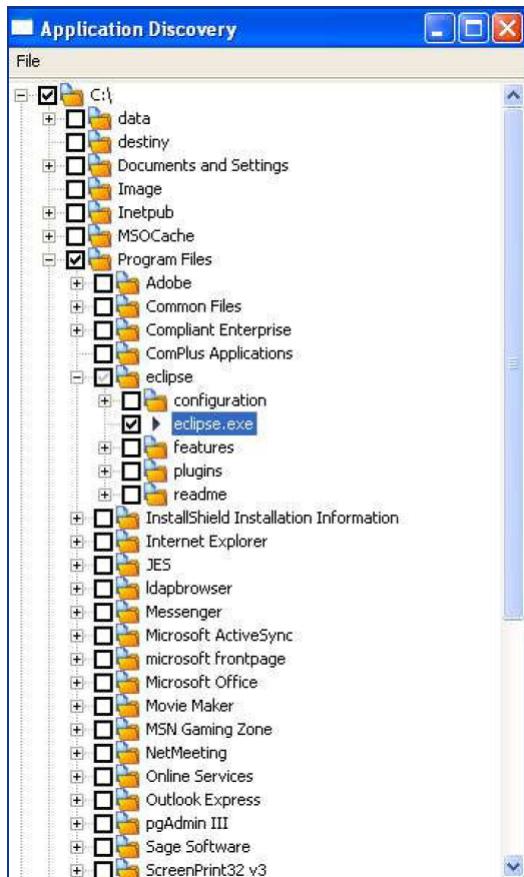


Figure 11: Enrolling Applications

4. In this window, navigate to the Program Files directory, or wherever else application EXE files are installed, and select the check box next to the applications you want to enroll.
5. In the File menu, use the GenerateLDIF command to generate an output LDIF file.

6. Assign a name and path to the file, then close the Application Discovery window.
7. Copy the generated LDIF file onto the host where Control Center is running, in the same directory as the Enrollment Manager.
8. Use Enrollment Manager's `-enroll -t LDIF` and `-sync` commands to enroll the LDIF file. For this process you need to prepare a definition file, specifying the name of the LDIF file you are enrolling. The Enrollment Manager is installed on the Control Center host machine, in the `Control Center \tools\enrollment\` directory. You can run it there, or copy it to another location to run it; just be sure to copy the LDIF file to the same directory before running the utility.

Related concepts

[Required entities](#) on page 37

Enrolling the following entities is required for all installations.

[Enrolling other entities](#) on page 82

[The definition file](#) on page 84

The `-enroll -t LDIF` command requires a definition file.

Related tasks

[Enrolling from an LDIF file for a single domain](#) on page 64

Use a different template for the definition file, called `app.sample.default.def` to enroll.

The definition file

The `-enroll -t LDIF` command requires a definition file.

The following figure shows the default LDIF definition file `app.sample.default.def`, which you should use as the template for the definition file when enrolling or updating applications from an LDIF file. It differs slightly from the one you use for users, hosts and groups. The actual file, when you open it, displays additional comments and instructions for editing.

In this file you need to edit the `ldif.sample.filename` element only; the syntax mapping is consistent with the AppDiscovery utility's LDIF output. When you save this file, you must rename it to remove the `.sample` string, as with all definition and connection files.

```

# This is the definition file for an LDIF file enrollment.
# It assumes a default Active Directory schema.

enroll.users           false
enroll.contacts         false
enroll.computers        false
enroll.applications    true

#
# Required attributes
#
entry.attributefor.staticid      uniqueGlobalIdentifier
app.requirements                 (objectClass=Application)
user.requirements                (objectClass=User)
contact.requirements             (objectClass=Contact)
computer.requirements            (objectClass=Computer)
group.requirements               (objectClass=Group)
group.attributefor.enumeration   member

#
# Attribute mappings
#
application.string.uniqueName   fullyQualifiedName
application.string.displayName   cn
application.cs-string.appFingerPrint applicationFingerPrint
application.cs-string.systemreference sn
isRecurring                      false
ldif.filename                     e:/p4_ajones_1.0/personal/applications.ldif

```

Supply name
and location
of LDIF file here.
Note front slashes!

Figure 12: LDIF definition file template for applications (app.sample.default.def)

Related concepts

[The LDIF definition file](#) on page 54

The default LDIF definition file can be used as the template for the definition file when enrolling or updating users, hosts and groups from an LDIF file.

[Definition files](#) on page 54

Definition files contain all the information required to map the data being enrolled from the information source to the format Control Center uses in its Information Network Directory tables.

Related tasks

[Enrolling applications](#) on page 83

Applications, unlike users, cannot be automatically enrolled from LDAP directories.

Multiple application versions

It is important to understand that Control Center considers each version of an application as a separate application.

If more than one version of an application is running in the environment, and policies need to govern all versions of that application, each version of the application must be enrolled separately.

When you enroll an application, Control Center automatically assigns a default name to it based on the executable filename. This default name does not identify applications by version. As a result, if you have more than one version of an application, the same default name is used and the second application overwrites the first unless you manually change the default application name.

1. Manually change the default name of the first version of the application in the LDIF file.
2. Enroll the first version of the application using `-enroll -t LDIF`.

3. Generate an LDIF file for the second version of the application and manually change the application's default name.
4. Enroll the second version of the application using `-enroll -t LDIF`.

Enrolling file shares

Enrolling information about file shares ensures that the use of differing file share paths to access the same document does not result in inconsistent policy enforcement.

In a policy definition, a policy might identify a file by specifying a particular drive and path; however, in a real world organization, users might use various shared drive configurations to access that file. Control Center needs to be aware of all the various file shares that are in use in the organization, so that it can consistently apply the correct policies to any given document, no matter which file share is used to access the document.

File shares typically are defined on file servers, but by definition they also include directories on any PC, that have been opened for sharing by other users. If you are running policy enforcers on both Windows and Linux file servers, file shares have to be discovered on both platforms.

Related concepts

[Required entities](#) on page 37

Enrolling the following entities is required for all installations.

[Enrolling other entities](#) on page 82

About Resource Path Discovery

The enrollment process involves discovering file shares defined on network hosts, and then enrolling them into the Information Network Directory.

Because this requires active discovery on the live network, it requires separate procedures for Windows and Linux.

Related concepts

[Enrolling entities for Windows and Linux systems](#) on page 38

Entities are enrolled into the Control Center system the same way regardless of whether enforcers are being installed on Windows file servers, Linux file servers, or a mixture of both.

Enrolling Windows file shares

To enroll Windows file shares, you use a utility called Resource Path Discovery (`ResourcePathDiscovery.exe`).

This utility runs a discovery procedure against the list of servers in the network, and generates a complete list of file shares on those servers, showing the actual network and directory path for each share, with the duplicate shares (different shares that refer to the same actual directory) grouped together. This list is referred to as a `MachineList` file.

This procedure requires special permissions, since it relies on visibility into multiple machines in the network.

As new file shares are added or existing file shares are deleted, it is important to periodically run Resource Path Discovery to update the file share enrollment. It's a good idea to set this up in a script that runs on a schedule, such as once a week.

By default, the Resource Path Discovery utility is installed in the `<installDirectory>\ControlCenter\tools` folder. You should run it from that location.

You can enroll file shares for all file servers.

1. Create a text file containing the list of any hosts in the organization where you want to search for file shares. This should include all monitored file servers and any PCs where file shares may have been defined. This file, called the MachineList file, can contain three sections:
 - [DOMAIN_MAP] is an optional section which is used in the case where DFS shares span multiple domains (for example, //sand.mycompany.com/dfsroot/dfslink --> c:/share on machinec.child.mycompany.com). In this example, CHILD is the domain name and child.mycompany.com is how the domain is referenced in DNS.
 - [WORKSTATION_LIST] is the list of computers you want to search for file shares, and is the only required section. Each machine name must be on a separate line and specified by the full DNS name.
 - [DFS_DOMAINS] is an optional section which lists the domains that contain domain-based DFS shares (for example, //mycompany.com/dfsroot . . .).

The following figure shows a file that would be used to enroll two computers named Alta and Kirkwood, in a single domain. When creating the MachineList file, be sure to use the upper case, square-bracket format for group headings, exactly as shown.

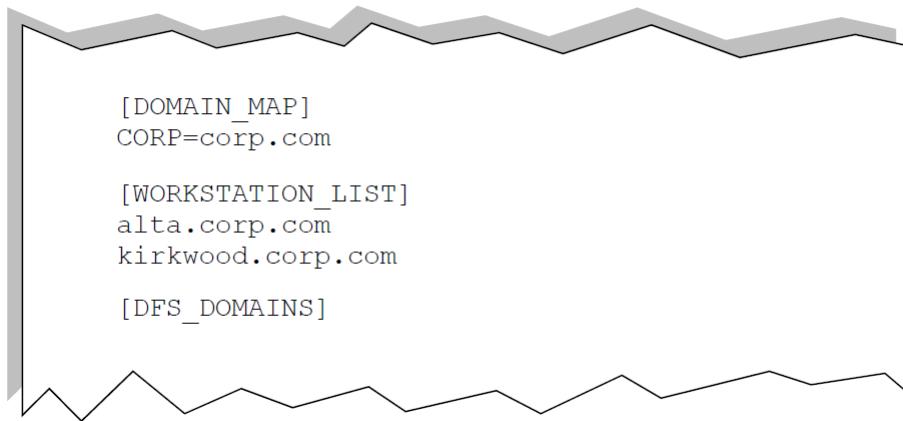


Figure 13: Sample Resource Path Discovery MachineList File

2. On any machine in the network that can connect to all the hosts listed in the file, log in as administrator of the domain that contains the listed servers. You must be logged in with one of the privileges required by Resource Path Discovery.
3. Change to the directory <InstallDir>\tools. For example: cd Program Files\NextLabs\Control Center\tools
4. Run the Resource Path Discovery utility, using the following syntax:

```
ResourcePathDiscovery aliases.txt <MachineListFile>
```

 where aliases.txt is the output file, and <MachineListFile> is the name of the text file you created in step 1.
5. When Resource Path Discovery finishes, open the output file and check its contents to be sure the utility completed the file correctly. The file should contain a section for each host listed in the MachineList file, with all of its file shares listed below it. The first line of each section should end with 0, indicating that no errors occurred during discovery. The following figure shows two hosts, Alta and Kirkwood.

```

alta.corp.com 0
'\\alta.corp.com\nestedshare' 'C:\Public\nestedshare'
'\\alta.corp.com\Public\nestedshare' 'C:\Public\nestedshare'
'\\kirkwood.corp.com\DFSRoot\DFSLink\nestedshare' 'C:\Public\nestedshare'
'\\kirkwood.corp.com\DFSRoot\DFSLink' 'C:\Public'
'\\alta.corp.com\Public' 'C:\Public'

kirkwood.corp.com 0
'\\kirkwood.corp.com\SYSVOL\corp.com\SCRIPTS' 'C:\WINDOWS\SYSVOL\sysvol\corp.com\SCRIPTS'
'\\kirkwood.corp.com\NETLOGON' 'C:\WINDOWS\SYSVOL\sysvol\corp.com\SCRIPTS'
'\\kirkwood.corp.com\DFSRoot' 'C:\DFSRootShare'
'\\kirkwood.corp.com\SYSVOL' 'C:\WINDOWS\SYSVOL\sysvol'

```

Figure 14: Sample Resource Path Discovery Output File

6. When you are satisfied that the output file (`aliases.txt`) has been correctly generated, put it into production by copying it to the `aliased_shared` directory on the Control Center host, like this:
`<InstallationDirectory>\server\aliased_shares\aliases.txt`
 7. Restart Control Center to force the Management Server to reread the file. This is required whenever the `aliases.txt` file is changed.
- All Windows file shares have been discovered and enrolled.

Related concepts

Permission requirements

[Microsoft Distributed File System](#) on page 89

Microsoft Distributed File System, or DFS, aggregates Windows file share names into a single machine or domain name with links to various physical shares.

Enrolling Linux file servers

For Linux servers, there are two remaining steps in the enrollment process. These steps must be performed after Windows file shares are enrolled.

1. On the Control Center host, change the Sharing properties of the `\server\aliased_shared` directory so it is open to file sharing from the Linux server.
2. From the Linux server host, generate a second list of file shares, for the Linux server, using a utility called Samba Directory Mapping. To do this, type the following command on the Linux file server:
`# /usr/local/ce/bin/smbDirMapping >~/aliases.txt`
This creates an output file on the Linux host, also called `aliases.txt`. This procedure, unlike the Windows discovery, does not require any `MachineList` file.
3. Append this new file to the Windows `aliases.txt` file, using the following commands from the Linux host:

```

# mkdir ~/mnt/aliased_shares
# mount -t cifs //<PolicyServerHost>/aliased /mnt/aliased_shares -o
user=<ControlCenterUser>
#cat ~/aliases.txt >> /mnt/aliased_shares/aliases.txt

```

The file shares discovered on the Linux server are added to the enrolled Windows file shares. If you run enforcers on more than one Linux server, you must repeat this procedure for each.

Microsoft Distributed File System

Microsoft Distributed File System, or DFS, aggregates Windows file share names into a single machine or domain name with links to various physical shares.

To enforce policy on the various physical shares when it is accessed through a DFS share, the Control Center must maintain a map of the share paths.

The list of share paths can be obtained by using the Resource Path Discovery utility. The following procedure is specific to using Resource Path Discovery with DFS shares.

Resource Path Discovery syntax:

```
ResourcePathDiscovery <outputfile name> <config filename>
```

The configuration file is a text file with the following sections:

```
[DOMAIN_MAP]
CORP=myco.corp.com
TEST=test.corp.com
[WORKSTATION_LIST]
lindsey.myco.corp.com
jasmine.myco.corp.com
natlie.test.corp.com
[DFS_DOMAINS]
corp.com
```

[DOMAIN_MAP] section lists the domains for a DFS share that spans multiple domains, for example, \\myco.corp.com\dfsroot\dfslink is a DFS share in the CORP domain but it points to a physical share in another domain such as \\tahiti.test.corp.com\myshare in the TEST domain. The list maps the NetBIOS domain name to the DNS domain name, for example, CORP=myco.corp.com.

[WORKSTATION_LIST] is the list of standalone DFS machine names. There are two types of DFS shares: domain-based and standalone with the following UNC formats.

- Domain-based: \\myco.corp.com\dfsroot\dfslink
- Standalone: \\jasmine.myco.corp.com\dfsroot\dfslink

This list contains only stand-alone DFS machines names, for example, jasmine.myco.corp.com. If domain-based DFS is used, the following section is used instead.

[DFS_DOMAINS] section lists the domains that use domain base DFS shares, for example: \\myco.corp.com\dfsroot\dfslink

The output file must be incorporated into the aliases.txt file as described in the Enrolling Windows file shares section.

Related tasks

[Enrolling Windows file shares on page 86](#)

To enroll Windows file shares, you use a utility called Resource Path Discovery (ResourcePathDiscovery.exe).

Enrolling location sites

A location site is a Control Center term referring to a group of hosts that you want to consider as a single location because they share certain characteristics.

These may be geographical, such as all computers at the Boston branch office, or virtual, such as all users connecting via a VPN. You define a site by creating a locations file that specifies one or more IP addresses or ranges of addresses and gives each a name, then using a utility called Import Locations to enroll the information from this file.

Once defined and enrolled, sites are not themselves components, but rather are available as a property of computer components. This enables policy designers to write policies that refer to a group of hosts based on their location. Location Sites—so called to distinguish them from SharePoint or other portal sites—are groups of computer resources that can be defined manually based on geographical proximity and other characteristics. Location Sites can be very useful for defining policies that cover branch offices or organizational units, or buildings on a campus. The sites might already have been defined during the initial enrollment of all other entities, but they can be added at any time. Defining sites is also known as enrolling sites.

The required sites might already have been defined during the initial enrollment of all other entities (technically we refer to defining sites as enrolling them); but they are not required, and can be added at any time.

You create sites by specifying one or more ranges of IP addresses and using the enrollment utilities to enroll network information about them as single network entities. These entities are then available as properties of computer components. These entities are then available as properties of computer components, as shown in the following figure.

The screenshot shows the 'Computer Component' properties dialog box. It includes fields for Display Name (set to 'Computer Component'), Description (set to 'Example computer component'), and Tags (set to 'Tags that classify the component'). Below these is a 'CONDITIONS' section with a table showing a single condition: 'Site != UnitedStates'. There is an 'ADD CONDITION' button and a '+ Add Condition' link.

Name	Operator	Value
Site	!=	UnitedStates

Figure 15: Computer component properties

To enroll sites, you use the Import Locations utility to extract site information from a locations file and add it to Control Center's Information Network Directory. Before you do this, you need to create a locations file manually. You can give this file any name you like.

By default, the Import Locations utility (`importLocation.bat`) is installed in the `<installDirectory>\Control Center\tools` folder. You should run it from that location.

1. Create the locations file. This is a text file containing any number of location definitions, one per line. Each line has the format:

`<LocationName> "<AddressMask>"`

where `<LocationName>` is the name by which you want to refer to the site when referring to it in Control Center, and `<AddressMask>` is a CIDR-like mask for the 32-bit IP address of a machine that is in the given location. Note the following requirements:

- Each address mask must be enclosed in double quotes
- The location name may not contain spaces

- The two elements must be separated by a space

For example, to define a location called VPN that represents all PCs connecting through a virtual private network, you might create the following entry:

```
VPN "192.168.254.0/24"
```

Create additional similar lines for all the other hosts that are part of the VPN group. Here is another example:

```
intranet "10.0.0.0/8"
```

You can also put comments in the file by beginning each comment line with #. For example:

```
# The following lines define the machines in the
# Boston office
```

2. Change to the directory <InstallDir>\tools. By default, the directory is:

```
\Program Files\NextLabs\PolicyServer\tools\
```

3. Run the Import Locations utility with appropriate values for all parameters, as shown in the following example. This line must provide the name and path of the locations file and connection information for the system database. The last parameter, -i, is required only if the database type is Oracle or SQL Server.

```
importLocation.bat -l <LocationsFile> -u <DB_user> -w <DB_password> -s
<DB_server> -p <DB_port> -d <oracle|postgres|SQLServer> [-i <instance>]
```

When the utility finishes running, all the locations defined in the input file are present in the Information Network Directory. You can use the enrolled sites as values for the Site property when defining Computer components.

For information on using the utility, use the command importlocation.bat -h.

Related concepts

[Optional entities](#) on page 38

Enrolling the following entities is optional.

[Enrolling other entities](#) on page 82

Enrolling SharePoint groups

The SharePoint Enforcer is available exclusively with the Entitlement Management product. If you are not using that product, you can ignore this section.

If you plan to deploy SharePoint-based policies, you may want to enroll SharePoint groups. SharePoint groups are required only if you are enforcing policies based on groups defined in SharePoint. If you are relying on Active Directory groups instead, you do not need to enroll SharePoint groups. However, since it is hard to know in advance what is required for policy implementation, it is usually best to enroll SharePoint groups so that they are available for use in policies.

Related concepts

[Optional entities](#) on page 38

Enrolling the following entities is optional.

[Enrolling other entities](#) on page 82

Setting access permissions

To enable the Control Center to connect to the SharePoint server, you need to create a service account and set it up with proper access level and security settings.

1. On the SharePoint server, create a new service account. In Windows 2003 server, you can use lusrmgr.msc to do this.

2. As a security measure, configure this service account so that it cannot log in locally or through terminal services. In Windows 2003 server, for example, you use `gpedit.msc` to do this: **Computer Configuration > Windows Settings > Security Settings > User Rights Assignment**, then select Deny log on Locally and Deny log on through Terminal Services.
3. Allow this service account full read access to the SharePoint application:
 - a. Select **SharePoint Central Administration: Application Management > Policy for Web Application**.
 - b. Click **Change Application**, and select the web application to be enrolled.
 - c. Click **Add Users** and select **all zones**.

 **Note:** Allowing full read access for the SharePoint Sites or Site Collections is not sufficient. Access must be allowed at the Application level, which is a larger unit than Sites or Site Collections.
4. Configure SharePoint to use NTLM authentication:
 - a. Select **Central Administration > Application Management > Authentication Providers**.
 - b. Select the **Integrated Windows Authentication**, **NTLM** check box, and clear the **Basic Authentication** check box.
 - c. Repeat the previous step for all zones, such as Default and Intranet.
5. When enrolling a SharePoint site, add the service account's username and password to the connection file. In the example shown in the Sample SharePoint Connection File (`sp.sample.default.conn`), the service account username is `SharePointCxn`.

```
# This is a template connection file for enrolling SharePoint groups.

login          SharePointCxn
domain        <domain name>
#password     mYpAssWOrD345
portals       http://sharepoint2016/sites/Mysite \n\
              http://sharepoint2016/MyOtherSite

ScheduledSyncTime   May 2, 2016 1:00 AM
ScheduledSyncInterv 360
```

Figure 16: Sample SharePoint Connection File (`sp.sample.default.conn`)

Related tasks

[Enrolling SharePoint users and groups](#) on page 92

Enrolling SharePoint groups is similar to the procedure for enrolling users and groups.

Enrolling SharePoint users and groups

Enrolling SharePoint groups is similar to the procedure for enrolling users and groups.

You use the Enrollment Manager to enroll the groups, based on a connection file and a definition file. For SharePoint sites, however, you need not make any changes to customize the definition file—only the connection file.

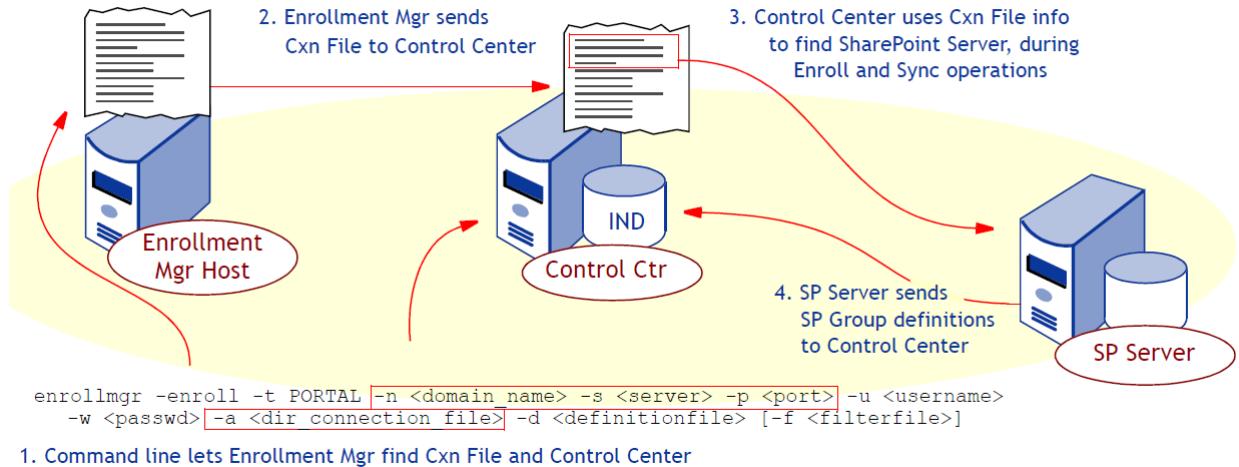


Figure 17: Enrollment Connections, SharePoint Groups

The site connection needs to be set up to retrieve data on groups, which is the only SharePoint data that is actually enrolled into the Control Center's data tables. Control Center connects to the SharePoint server whenever a user browses the site while defining components; but site content is not actually enrolled the way Active Directory content is.

If you need to enroll groups from more than one SharePoint site, you must designate each site in the SharePoint connection file.

1. Open the sample SharePoint connection file `sp.sample.default.conn`, and edit the elements as shown in the following figure. Required elements are indicated by rectangles; optional elements are indicated by ovals.
2. Save the changes and rename the file to prevent it from being overwritten the next time the Control Center is upgraded. Be sure to specify the name properly with the `-a` flag in the `Enroll` command line (Step 4).
3. You ordinarily do not need to make any changes to the content of the definition file, `sp.sample.default.def`, but you do need to copy it somewhere where you can point to it in the `Enroll` command line (Step 4), and you should rename the filename to prevent the file from being replaced the next time you upgrade Control Center.
4. Change to the directory `<InstallDir>\tools`. By default, this is:

```
\Program Files\NextLabs\PolicyServer\tools\enrollment
```

```
# This is a template connection file for enrolling SharePoint groups.

login           SharePointCxn
domain          <domain name>
#password       mYpAssWOrD345
portals         http://sharepoint2016/sites/MySite \n\
                http://sharepoint2016/MyOtherSite

ScheduledSyncTime May 2, 2016 1:00 AM
ScheduledSyncInterv 360
```

Figure 18: Sample SharePoint Connection File (`sp.sample.default.conn`)

Table 14: Elements of SharePoint connection files

Element	Description
login	Specifies a username with access privileges to connect to the SharePoint server—specifically, of the Windows service account that was specially created for this connection.
domain	Specifies the fully qualified domain name (FQDN) on which the SharePoint server is located. If the SharePoint server uses SSL encryption, the FQDN value must match the Common Name (CN) value of the SharePoint certificate.
password	<p>Specifies a valid password for this user, for connecting to the SharePoint server.</p> <p>This parameter is optional. Because it is an unencrypted string, you may prefer not to expose it in the connection file this way. If the parameter is not present in the file (that is, if you leave it commented out), the user is prompted for the connection at the time when he performs the enrollment. If the parameter is present, it is simply be read, with no such prompt. This latter may be preferable for the sake of convenience, during testing or in other restricted-use context when security is not a concern.</p> <p>To define an unencrypted password, remove the # character at the beginning of the line.</p>
portals	Each line here represents one site on the SharePoint site collection, which you want to enroll. It must contain the full URL delimited by backslash characters, as shown in the placeholder values in the template file. The hostname in the URL must match the Common Name (CN) of the SharePoint certificate. If you are adding multiple portal sites, you can separate them with \n\
ScheduledSyncTime	<p>Optional parameter. Enables you to set a specific time for the enrollment to be automatically synchronized with its source. After this date and time, the enrollment autosynchronizes at the interval specified by the ScheduledSyncInterval parameter.</p> <p>The date and time value is locale-specific, and you must use Java's MEDIUM style for the date, and the SHORT style for the time. For example:</p> <p>For English (US): Sep 18, 2013 6:00PM</p> <p>For English (UK):18-Sep-2013 18:00</p> <p>For the German (Germany):18.09.2013 18:00</p> <p>At enroll-time, the date must be a future date.</p> <p>This value is only active if the Scheduled Sync Interval is set to some positive value. If you leave ScheduledSyncInterval at the default, 0, the synchronization feature is disabled.</p>

Element	Description
ScheduledSyncInterval	<p>Optional parameter. When left at the default, 0, disables the automatic synchronization feature. When changed to any positive value, specifies a regular re-synchronization schedule, expressed in minutes following the time specified by the ScheduledSyncTime.</p> <p>This setting has no influence until after the ScheduledSyncTime passes. That is, in the example above Control Center would wait until 1 a.m. on May 2 to synchronize the enrollment to its source, and then would re-sync it every 6 hours (360 minutes) thereafter.</p> <p>Manual synchronization is possible only if this value is set to zero. If you have changed it to enable automatic synchronization, you must change it back to zero to run a manual sync.</p>

5. Run the Enrollment Manager's Enroll command with the Type = PORTAL along with the other standard arguments, as in the following example:

```
enrollmgr.bat -enroll -t PORTAL -n MyCompany2 -s <Moskau> -p 8443 -u Administrator -w MY876password -a sp.default.conn -d sp.default.def
```

This line must specify a unique enrollment name, the name and port of the Control Center host, a Control Center user and password, and the names and paths of the connection and definition files.

6. Run the Enrollment Manager's Sync command, as shown in the following example. Neither the connection nor the definition file is required for this command; the enrollment name is sufficient.

```
enrollmgr.bat -sync -n MyCompany2 -s <host> -p 8443 -u <Admin> -w <securityPwd>
```

At this point the SharePoint groups are available for use in defining components in Control Center. You can use the Lookup button to display all portal content in a tree structure, so you can browse it directly as you define portal components.

Related tasks

[Setting access permissions](#) on page 91

To enable the Control Center to connect to the SharePoint server, you need to create a service account and set it up with proper access level and security settings.

[Importing SharePoint certificates](#) on page 95

Control Center needs to connect to the monitored SharePoint servers in the network from time to time.

Related reference

[Command line arguments for Enrollment Manager](#) on page 45

Synchronizing SharePoint enrollments

The SharePoint connection file contains two parameters, ScheduledSyncTime and ScheduledSyncInterval, that control the auto-synchronization feature.

These work the same way here as for LDAP synchronization. However, it is important to understand that synchronization refers not to the whole content of a SharePoint site, but only to the group definitions of the site. This is because groups are the only data that is actually enrolled into Control Center.

Importing SharePoint certificates

Control Center needs to connect to the monitored SharePoint servers in the network from time to time.

If a SharePoint server is set up to use SSL encryption for such connections, Control Center's enrollment mechanism can support that as long as the SharePoint certificate has been imported into its truststore.

1. Locate the certificate on the server where SharePoint is running. It must be a valid X509 certificate.
2. Place a copy of the certificate in a convenient directory on the Control Center host.

3. On the Control Center host, open a command window and go to the <INSTALL_DIR>\ Control Center\server\certificates directory.
4. Run the Import command of the Keytool utility, using the following syntax:

```
\..\java\bin\keytool -import [-v] [-alias <alias>]
[-file <cert_file>] [-keypass <keypass>]
[-keystore <keystore>] [-storepass <storepass>]
[-storetype <storetype>] [-providerName <name>]
[-providerClass <provider_class_name> [-providerArg <arg>]]...
```

Here is an example command:

```
\..\java\bin\keytool -import -alias jedsharepoint -file "c:\Documents
and Settings\Administrator\Desktop\certs\pf-https-web.cer" -keystore dcc-
truststore.jks -keypass 123begemot! -storepass 123begemot! -storetype jks
```

The functions of all arguments are described in the following table.

Table 15: Certificate Keytool, Functions of Arguments

Argument	Function
[-v]	Optional flag; when present, utility runs in verbose mode
[-alias <alias>]	Required parameter; specifies the alias of your choice
[-file <cert_file>]	Required parameter; specifies the full path and name of the certificate file you copied from the SharePoint server (Step 2)
[-keypass <keypass>]	Required parameter; specifies the password for the key you are importing into the keystore. This is the Control Center SSL Password assigned during installation
[-keystore <keystore>]	Required parameter; specifies the name of the keystore file. By default, this is <code>dcc-truststore.jks</code> .
[-storepass <storepass>]	Required parameter; specifies the password required by the keystore file. This is the same as the <code>keypass</code> argument—the Control Center SSL Password assigned during installation.
[-storetype <storetype>]	Required parameter; specifies the keystore type, which should be <code>jks</code> , as in the example above.
[-providerName <name>]	Optional parameter; should not be used in standard installation.
[-providerClass <provider_class_name> [-providerArg <arg>]]	Optional parameters; should not be used in standard installation.

5. When prompted to “Trust this certificate”, type “y” and press **Enter**.

6. Restart the Policy Server:

For Windows:

- a. Go to **Control Panel > Administrative Tools > Services**.
- b. Find Control Center Policy Server in the list of services.
- c. Right-click the service name, then select **Start**.
- d. To stop Control Center, right-click the service name, then select **Stop**.

For RHEL:

- a. To stop Control Center, use `stop-policy-server.sh` located in `/<installation folder>/PolicyServer`.
- b. Type the one of the following commands: `sudo ./stop-policy-server.sh`
or
`sudo service CompliantEnterpriseServer stop`
- c. To check if the Tomcat process has started, type the following command: `ps -ef | grep tomcat`
- d. To start Control Center, use `start-policy-server.sh` located in `/<installation folder>/PolicyServer`.
- e. Type one of the following commands as root user:
 - `sudo ./start-policy-server.sh`
 - `sudo service CompliantEnterpriseServer start`

Related tasks

[Enrolling SharePoint users and groups on page 92](#)

Enrolling SharePoint groups is similar to the procedure for enrolling users and groups.

Enrollment utility password requirements

Some enrollment utilities require users to enter the password of the super user Administrator account, which is initially configured during Control Center installation. This password can be changed as needed.

Related concepts

[Enrolling other entities on page 82](#)

[Password and users on page 352](#)

Control Center uses several different passwords for various restricted activities, and these passwords are maintained in different ways.

Chapter

6

Defining custom properties

Topics:

- [User, hosts, and application properties](#)
- [About Property Manager](#)
- [Portal content properties](#)

This section describes the procedures for defining custom properties for various kinds of entities you define as components. You cannot define custom properties for Application or Action components.

Related concepts

[User, hosts, and application properties](#) on page 99

Each of the three LDAP entity types Users, Hosts, and Applications has a default set of properties (technically, LDAP attributes), which are displayed in the CONDITIONS section of Subject components.

[About Property Manager](#) on page 100

The Property Manager utility (`propertymgr.bat`) enables you to define custom properties for User and Computer components, and for user and host groups.

[Document properties](#)

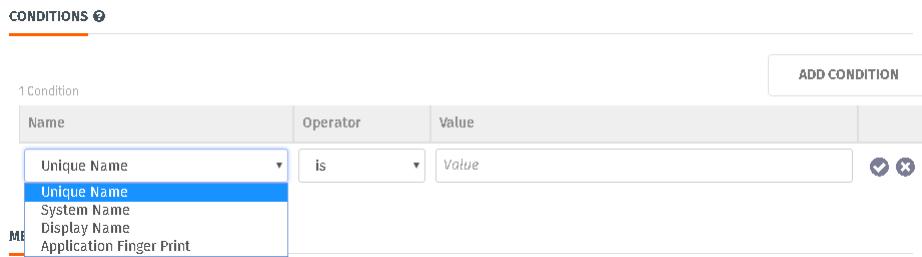
[Portal content properties](#) on page 103

Portal content properties are the properties that appear by default in the Property Name combo-box of the component definition.

User, hosts, and application properties

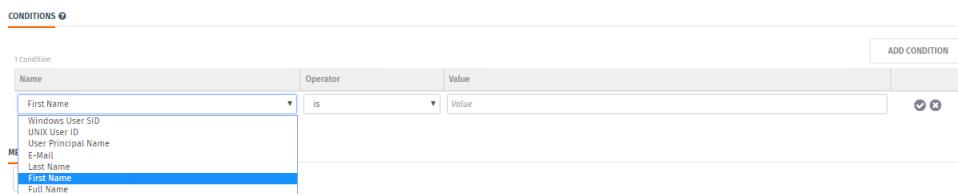
Each of the three LDAP entity types Users, Hosts, and Applications has a default set of properties (technically, LDAP attributes), which are displayed in the CONDITIONS section of Subject components.

These are stored in internal tables in the Information Network Directory. Each property has both a logical name, which is the actual column name in the database table, and a display name, which is the label that shows in the drop-down list. Properties are displayed in the drop-down list for all types of components as shown in the following figures.



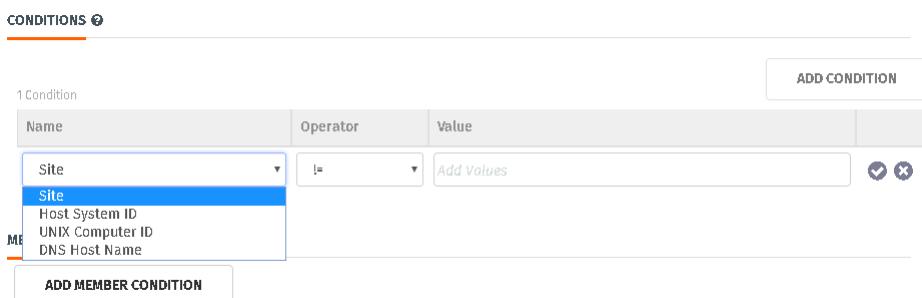
The screenshot shows the 'CONDITIONS' tab of a component configuration dialog. It displays a single condition row with the following fields: Name dropdown (set to 'Unique Name'), Operator dropdown (set to 'is'), and Value input field ('Value'). A small 'ADD CONDITION' button is located to the right of the row. Below the row, a dropdown menu lists several options: Unique Name, System Name, Display Name, and Application Finger Print. The 'Unique Name' option is currently selected and highlighted in blue.

Figure 19: Application component properties



The screenshot shows the 'CONDITIONS' tab of a component configuration dialog for users. It displays a single condition row with the following fields: Name dropdown (set to 'First Name'), Operator dropdown (set to 'is'), and Value input field ('Value'). A small 'ADD CONDITION' button is located to the right of the row. Below the row, a dropdown menu lists several options: First Name, Windows User SID, UNIX User ID, User Principal Name, E-Mail, Last Name, and Full Name. The 'First Name' option is currently selected and highlighted in blue.

Figure 20: User component properties



The screenshot shows the 'CONDITIONS' tab of a component configuration dialog for hosts. It displays a single condition row with the following fields: Name dropdown (set to 'Site'), Operator dropdown (set to 'is'), and Value input field ('Add Values'). A small 'ADD CONDITION' button is located to the right of the row. Below the row, a dropdown menu lists several options: Site, Host System ID, UNIX Computer ID, and DNS Host Name. The 'Site' option is currently selected and highlighted in blue. At the bottom left of the dialog, there is a button labeled 'ADD MEMBER CONDITION'.

Figure 21: Host component properties

The following table describes the pre-seeded attributes available for defining Computer components. Values for all default properties are case-insensitive, so you do not need to worry about capitalization as you type them.

The following table describes the pre-seeded attributes available for defining User components.

Table 16: User Component: Pre-seeded attributes

Attribute Name	Short Name	Data Type	Operators
Window User SID	windowsSid	String	is not, is
UNIX User ID	unixId	String	is not, is
User Principal Name	principalName	String	is not, is

Attribute Name	Short Name	Data Type	Operators
E-mail	mail	Multival	!=, includes, =, exactly matches
Last Name	lastName	String	is not, is
First Name	firstName	String	is not, is
Full Name	displayname	String	is not, is

Table 17: Host Component: Pre-seeded attributes

Attribute Name	Short Name	Data Type	Operators
Host System ID	windowsSid	String	is not, is
UNIX Computer ID	unixId	String	is not, is
DNS Host Name	dnsName	String	is not, is

Table 18: Application Component: Pre-seeded attributes

Attribute Name	Short Name	Data Type	Operators
Unique Name	uniqueName	String	is not, is
System Name	systemReference	String	is not, is
Display Name	displayName	String	is not, is
Application Finger Print	appFingerPrint	String	is not, is

Related concepts

[Defining custom properties](#) on page 98

About Property Manager

The Property Manager utility (`propertymgr.bat`) enables you to define custom properties for User and Computer components, and for user and host groups.

The Property Manager, like the Enrollment Manager, is automatically installed on the Control Center host, in the following location:

`Program Files\NextLabs\PolicyServer\tools\enrollment`

The following table describes the three functions of this tool.

Table 19: Property Manager: CLI commands

Command	Use	Full command string
-add	Add a custom property to users, hosts or groups	<code>propertymgr -add -s <server> -p <port> -u <couser> {-w <cepwd> -ew <encryptpwd>} -t <type> -l <logicalname> -i <displayname> -e <entitytype></code>
-delete	Delete a property from users, hosts or applications	<code>propertymgr -delete -s <server> -p <port> -u <couser> {-w <cepwd> -ew <encryptpwd>} -l <logicalname> -e <entitytype></code>

Command	Use	Full command string
-list	List all current properties of users, hosts and applications	propertymgr -list -s <server> -p <port> -u <ceuser> {-w <cepwd> -ew <encryptpwd>}
-h	Displays a help screen showing definitions of all valid command line arguments	propertymgr -h

The following table provides explanations of all arguments.

Table 20: Property Manager: Command arguments

Argument	Description
-s <server>	The name or URL of the host where your Control Center is running.
-p <port>	The port of the Control Center host—8443, by default.
-u <user>	Username of a Control Center user with administrator privileges.
-w <passwd>	The Control Center utility password.
-ew <encryptpwd>	The Control Center utility password that is encrypted. You can generate an encrypted password using the <code>mkgpassword.bat</code> utility.
-l <logicalname>	The logical name of the property you are adding or deleting. This is the actual column name in the internal data tables; it must be unique, and may not include spaces or special characters.
-i <displayname>	The display name you want to assign to the property you are adding. This is the string that displays in the conditions section. It is case-sensitive, and may contain spaces.
-t <datatype>	Specifies the data type of the property you want to add; valid values are STRING, CS-STRING, NUMBER or DATE. Used only with the Add Property command. These values are case-sensitive and must be capitalized as shown.
-e <entitytype>	Specifies the type of entity to which you want to add a property; valid values are USER, HOST, or APPLICATION. Used only with Add Property command. These values are case-sensitive and must be capitalized as shown.

Related concepts

[Defining custom properties](#) on page 98

[Utility Security Password](#) on page 353

The utility security password enables anyone to use several Control Center utilities.

Adding properties

You can use the `-add` command to add properties to users, hosts, and applications.

This procedure applies only to Control Center; any properties you add should also be supported by your LDAP back end. Adding a custom property requires two steps: first define the custom property, using the Property Manager CLI; and then map that property to an LDAP attribute in your Active Directory.

When you define your custom attribute, you must specify the entity type (that is, whether it is a new property of users, hosts, or applications) and the data type (string, cs-string, number or date), and also supply both a logical name and a display name. For custom properties, both property names and values are case-insensitive.

1. On the host where the Management Server component of the Control Center is installed, open a console window.

2. Change to the directory `<InstallDir>\tools\enrollment`. By default, this is:

```
\Program Files\NextLabs\PolicyServer\tools\enrollment
```

3. At the prompt, run the `Add` command as follows, filling in the appropriate values for the arguments:

```
propertymgr -add -s <server> -p <port> -u <ceuser> {-w <cepwd>| -ew <encryptpwd>} -l <logicalname> -i <displayname> -t <datatype> -e <entitytype>
```

The system displays a confirmation message.

4. To add the new property to the appropriate table in the Information Network Directory (the Control Center system dictionary), you have to edit the default definition file to include the new property as a new line in the Attribute Mappings section, as shown in the following figure. Each line in this section serves both to create a property in the dictionary, and to map that property to an LDAP attribute that is defined in your Active Directory. The left column defines the dictionary property; the right one maps it to an AD attribute.

 **Note:** The value you define in the left column must correspond to the property you added with the Property Manager's `Add` command, in the previous step.

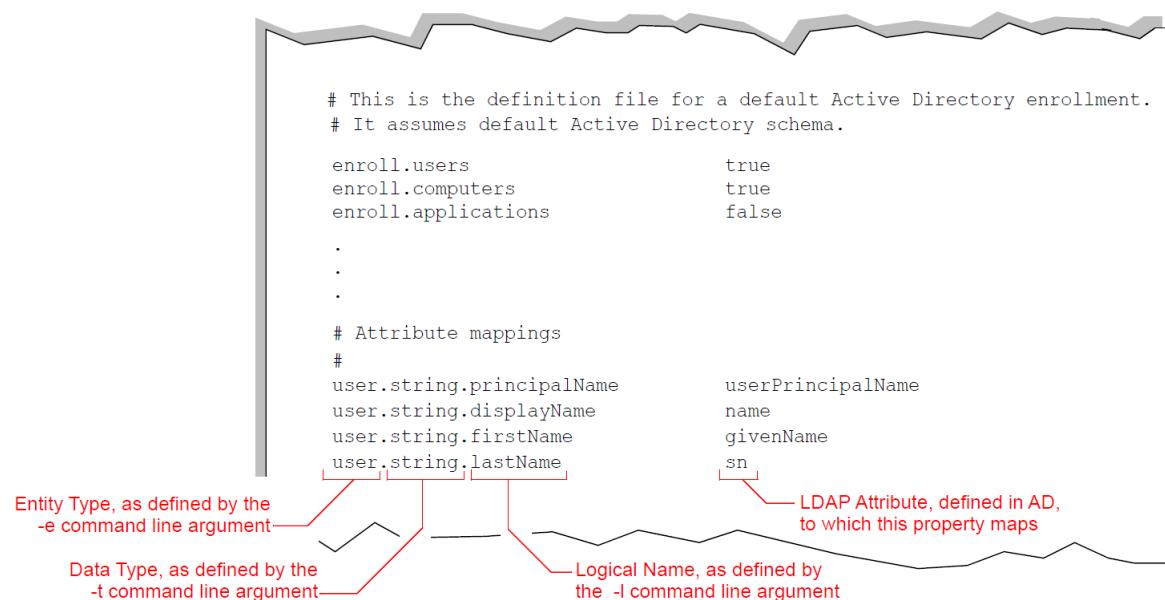


Figure 22: Editing Attribute Mappings (ad.default.def)

5. Next, run the Enrollment Manager's standard `-update` command. This step updates the table structure, but does not import the actual data values to for the new property.
6. To import the data values, run the Enrollment Manager's standard `-sync` command. When this finishes running, the new property is available for constructing components. You can confirm this by opening the CONDITIONS section under Subject components for a user, application, or a host component. Use the `List Properties` command to check your addition.

Related concepts

[Synchronizing enrollments](#) on page 73

The entities in network domains change frequently, and you need to ensure that any changes are updated into Control Center.

Related tasks

[Updating directory enrollments for a single domain on page 69](#)

You should use the `-update` command whenever you want to enroll different types of entities or a changed data structure from a domain you have already enrolled.

Deleting properties

You can use the `propertymgr -delete` command to delete a property from any type of entity. You must specify the logical name of the property, and the entity type.

1. On the host where the Management Server component of the Control Center is installed, open a console window.
2. Change to the `\enrollment` directory.
3. At the prompt, run the Delete command as follows, filling in the appropriate values for the arguments:

```
propertymgr -delete -s <server> -p <port> -u <ceuser> {-w <cepwd>|-ew <encryptpwd>} -l <logicalname> -e <entitytype>
```

When the properties have been deleted, a confirmation message appears.

To verify that properties have been deleted, use the `-list` command.

Listing properties

You can use the `propertymgr -list` command to display a list of all properties currently defined for all three entity types.

1. On the host where the Management Server component of the Control Center is installed, open a console window.
2. Change to the directory `<InstallDir>\tools\enrollment`. By default, this is:
`Program Files\NextLabs\PolicyServer\tools\enrollment`
3. At the prompt, run the `List Properties` command as follows, filling in the appropriate values for the arguments:

```
propertymgr -list -s <server> -p <port> -u <ceuser> {-w <cepwd>|-ew <encryptpwd>}
```

All properties are displayed in a single list, sorted by entity type. Each line shows the data type, logical name, display name, and data type. The display format is:

```
<TYPE>. <logicalName> - <display Name>: <data Type>
```

Portal content properties

Portal content properties are the properties that appear by default in the Property Name combo-box of the component definition.

The following table describes the default properties available for defining portal content components. Values for all default properties are case-insensitive, so you do not need to worry about capitalization as you type them.

Table 21: Portal content components: Default properties

Property	Description	Allowed operators
Created	The date when the content item was created.	'after', 'on or before'

Property	Description	Allowed operators
Created by	The user who initially created the content item.	'is', 'is not'
Description	A text description of the content item, if one has been supplied.	'is', 'is not'
File Size	The file size of the content item.	'=', '>', '!=>=', '<', '<=', '!='
Modified	The date when the content item was most recently modified.	'after', 'on or before'
Modified by	The user who most recently modified the content item.	'is', 'is not'
Name	The filename of the content element. Wildcards are supported.	'is', 'is not'
Sub-type	The Sub-type of the content item; valid only if Type property is Portlet. Valid values include List and Library. Not mandatory; if type is portlet and no subtype is specified, the component includes both subtypes.	'is', 'is not'
Title	The title string that has been applied to the content item. Wildcards are supported.	'is', 'is not'
Type	The type of content item. Valid values include: <ul style="list-style-type: none"> • Portal • Site • Page • Portlet (that is, library or list) • Item (that is, library or list item) 	'is', 'is not'

Related concepts

[Defining custom properties](#) on page 98

Defining custom portal properties

As with other component types, you can define custom properties for portal content components.

You define the properties as you work in the portal itself—for example, by adding a list or a library in SharePoint. (This topic is only relevant if you are using the Entitlement Management product, with SharePoint enforcers.)

In the course of day-to-day work in SharePoint, users may define large numbers of custom properties. (After all, the structural flexibility and decentralized control of portals is an important part of their collaborative value.) For this reason, you may not want to expose all custom properties, since the list in the CONDITIONS section could become too long. To help with this, the field is an open text input, where the user can type any property name he wants. This means that any custom property that has been defined in the portal can be used in policies, regardless of whether it is exposed in the list in the CONDITIONS section.

For whichever custom properties you do want to expose in the CONDITIONS section, follow this procedure.

1. Open the `configuration.xml` file on the Control Center host, locate the `<CustomAttributes>` section within `<DPS>`, and add a `<ResourceAttribute>` section to describe the attribute you are adding as shown in the following figure. This section has the following required elements:
 - `<Group>`: The set of properties in which the attribute appear. Specify `With Property` or `With Content`. The former can be used in all types of components; the latter is used for defining content-based components.
 - `<DisplayName>`: The name to be displayed in the CONDITIONS section. This name is case-sensitive; uppercase and lowercase letters are displayed as typed.
 - `<Name>`: The name of the attribute, as you defined it in the portal. You must prepend this value with `portal:` as shown in the following figure; this is required to differentiate between portal content properties and those of other types of resources, such as documents. (All kinds of resources are listed together in this section of the configuration file.) This name is case-insensitive.
 - `<Type>`: The data type of this attribute; currently `STRING` is the only supported type.
2. Close the configuration file, saving your changes, and restart Control Center.



Figure 23: Portal Content: Custom Properties

At this point when you open the Control Center web console, the new attribute is available to use in defining portal content components. To test it, create a portal content component and check the Name list in the CONDITIONS section.

You can define multiple custom attributes in the configuration file, for different kinds of resources. The previous example above has one resource for portal content. All resources appear in Control Center web console, for their respective component types.

Chapter

7

Managing policy enforcers

Topics:

- [About policy enforcers](#)
- [Adding policy enforcers](#)
- [About ICENet servers and policy enforcer profiles](#)
- [Load balancing](#)
- [Stopping policy enforcers](#)
- [Restarting policy enforcers](#)
- [Service user account permissions](#)
- [Uninstalling policy enforcers](#)
- [Reconfiguring policy enforcers](#)
- [Viewing policy enforcer status](#)
- [Managing policy enforcer profiles](#)
- [Database management](#)
- [Configuration tools](#)

Related reference

[About the console](#) on page 11

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

About policy enforcers

Policy enforcers are Control Center clients that monitor document access and use, and enforce policies at the point of use.

Each policy enforcer consists of a policy enforcement point (PEP); a policy decision point (PDP), also known as a Policy Controller; and a policy adapter.

There are several NextLabs policy enforcers that are predefined to work on specific platforms and enforce specific systems, such as Windows desktops, Microsoft Office Communicator, Microsoft Outlook, or Linux systems. In addition, SDKs enable developers to create PEPs that enforce policies on in-house legacy applications or third-party applications for which predefined enforcers are not available.

Related concepts

[Overview of PEP development](#) on page 256

Adding policy enforcers

If you need to add enforcers, you can do so at any time using managed installation methods, such as a login script or Windows Group Policy Object (GPO).

Control Center provides an MSI installation file that can be used with these installation methods. This section provides general instructions for performing a managed installation using GPO. For details about GPO, see the Microsoft documentation.

1. Use Microsoft Group Policy Object (GPO) to set the domain controller and register each target machine on which you want to install policy enforcer software.
2. Create an MSI transform file (MST file) to set the value of the required installation parameter `ICENET_SERVER_LOCATION` to the host where the ICENet server is installed. Use the format `machineName:port`. If you use the default port number 8443, you can omit it from the value and simply set the `machineName`.
3. Use a transformation tool to create the file. For details, see the transformation tool's documentation.
4. Configure GPO to run the appropriate MSI and MST files whenever each machine restarts:
 - Desktop or laptop PCs: `WindowsDesktopEnforcer-setup.msi`
 - File servers: `WindowsFileServerEnforcer-setup.msi`

On startup, the domain server automatically sends the binary to the target computer and runs the installer.

About ICENet servers and policy enforcer profiles

This section describes how policy enforcer profiles connect to and control ICENet servers.

Related tasks

[Defining policy enforcer profiles](#) on page 117

Initial registration

When you install an enforcer, the installation wizard prompts you for an ICENet server to connect to. This does not necessarily represent the ICENet server the enforcer always connects to, but rather the one it connects to the first time it starts up, to register itself with the Control Center.

When an enforcer registers, it informs the Control Center of its location and status, and acquires the operating parameters specified by its assigned profile. In most cases, this is the default profile for its type.

After registration, the enforcer connects to the Control Center through the ICENet server specified in its assigned profile. This might be the same ICENet server that was specified during enforcer installation or a different server. In the latter case, the ICENet server specified during installation is no longer used by that enforcer. The key point here is that although an ICENet server is specified during installation, the enforcer does not necessarily continue to connect through that ICENet server for anything but initial registration.

Relationships of profiles ICENet servers, and hosts

Enforcer profiles, ICENet Servers, and hosts have these relationships.

- Each enforcer host in the network, including file servers, desktops, and laptops, is a client of only one ICENet Server. You specify the server when you install the enforcer.
- Each profile can be associated with only one ICENet server, or with one cluster of load-balanced servers. This is the profile's parent ICENet Server, which is set as a property of the profile.
- A profile is available only to those enforcers running on hosts that are served by the ICENet server of which they are clients. In other words, an enforcer can be assigned only those profiles that belong to its ICENet server.
- Each ICENet server may have multiple profiles associated with it, just as it may have multiple enforcer hosts as clients.
- You can define a profile and assign it to one or more of the hosts belonging to the parent ICENet server, and you can reassign different hosts to a profile.
- ICENet servers can be assigned to existing profiles. When this is done, the profile is automatically made available for assigning to all the enforcer clients of that server. This is similar to moving the profile from one ICENet server to another. It removes the profile from all enforcers served by the previous ICENet server, replaces the profile with the appropriate default, and makes the profile available to assign to all enforcers served by the new ICENet server.

These relationships are illustrated in the following figure.

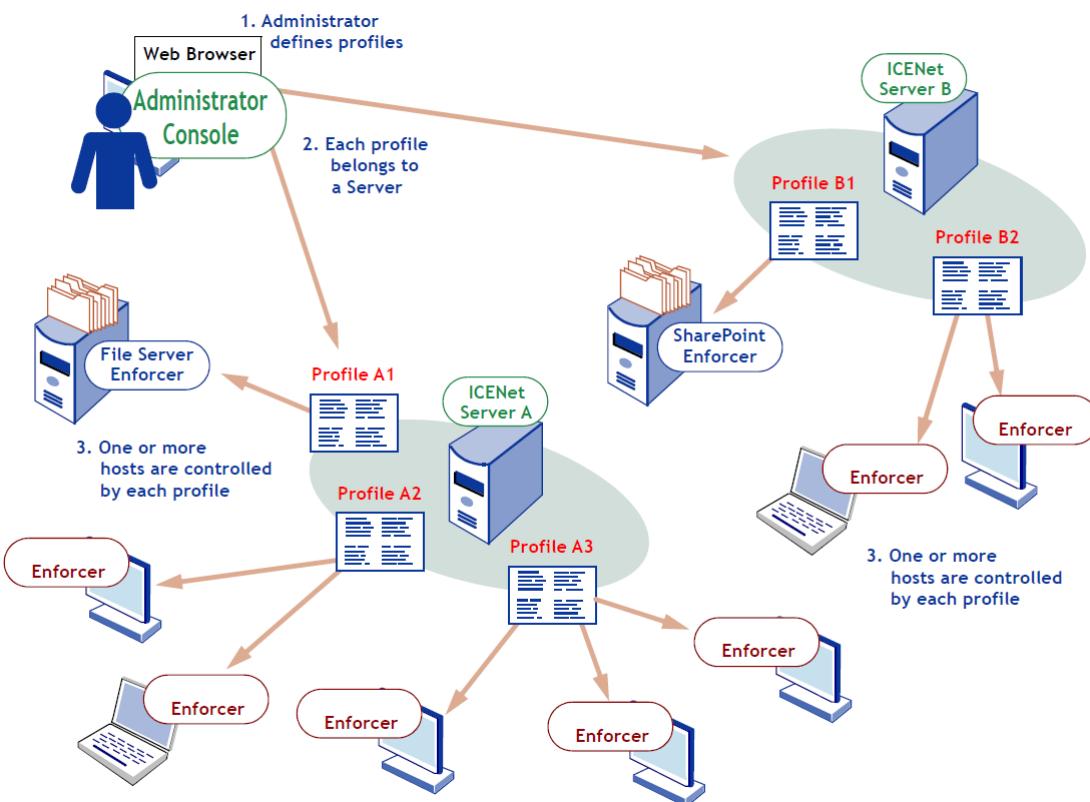


Figure 24: Profiles, ICENet servers, and hosts

Load-balanced or clustered ICENet servers

As far as profiles are concerned, each cluster of ICENet servers behaves like a single server.

You can associate a load balancer or cluster with a profile by entering its virtual IP address or virtual hostname in the OR field of the server settings as shown in the following figure. In this case, the profile is available to all hosts of all the ICENet servers in the cluster.

For information about setting up load sharing, see the Load balancing section. For information about setting up a failover cluster, see the NextLabs Control Center Installation and Upgrade Guide.



Figure 25: Load-balanced ICENet servers

Related concepts

[Load balancing](#) on page 109

Moving profiles between ICENet servers

To move a profile from one ICENet server to another, simply assign the new server to the profile. This removes the profile from all enforcers served by the previous ICENet server, and makes it available to be assigned to all enforcers served by the new ICENet server. However, it is not actually assigned to any of these clients until you assign it explicitly, by adding the clients to the list on the profile's Hosts tab.

Moving profiles in this way involves a delay, due to the reliance on heartbeat messages. If you ever decide to move the profiles associated with one server over to another server, you should leave both servers running long enough to allow all client hosts to receive instructions about the handover. Clients connected to the network receive the update with their next heartbeat message; but any mobile clients that are not connected to the network do not get this update until they connect. If the old server is shut down before mobile clients connect, the clients continue to enforce policies, but they are cut off from the Control Center system. Such clients cannot receive any new policy updates or send their audit data back to the Activity Journal.

Load balancing

The ICENet server controls all communication between the Control Center and the enforcers running on the network, and it handles a good deal of the inter-server data transfer within Control Center. For this reason, the ICENet server represents the most likely performance bottleneck in the system. To improve performance, you can install several ICENet Servers on distributed hosts. If you do this, you must use a load balancer to divide the workload among all ICENet servers.

You can set up load balancing any time by installing the load balancer between the devices on which policy enforcers are installed and the machines on which the ICENet servers are installed, as shown in the following figure. You can set up load balancing among many ICENet servers regardless of whether they are running on separate hosts, or many on the same host.

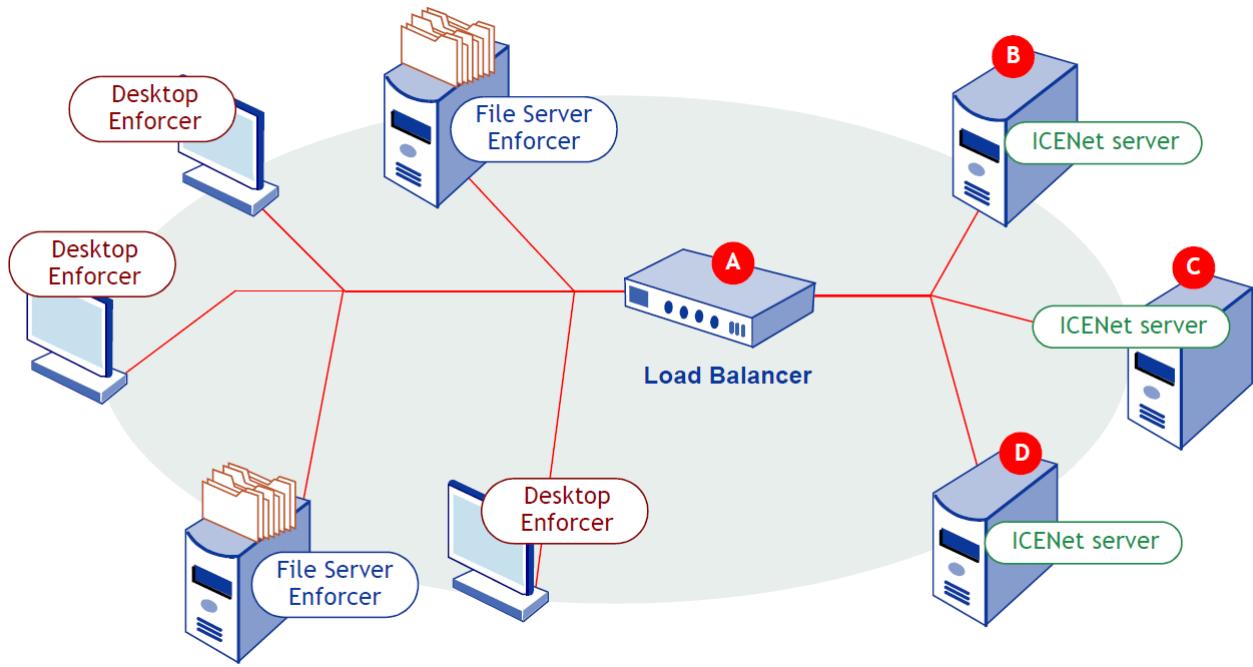


Figure 26: Load balancing ICENet servers

Related concepts

[Load-balanced or clustered ICENet servers](#) on page 109

As far as profiles are concerned, each cluster of ICENet servers behaves like a single server.

Setting up load balancing after Control Center installation

This section describes how to set up load balancing after Control Center installation.

1. Install the ICENet server components on several server hosts.
2. Set up the load balancer so that it refers to the ICENet servers.

 **Note:** For more information about installing the ICENet server components and setting up load balancer, see the NextLabs Control Center Installation and Upgrade Guide.
3. Configure the load balancer to use stateless (“non-sticky”) load balancing.
4. Use the Administrator console to modify all your currently defined policy enforcer profiles, changing the ICENet server location from its actual location to the load balancer’s virtual IP address or virtual hostname. Because the Administrator console cannot autodiscover the load balancer, it does not appear in the list in the ICENet servers settings. You must type it manually in the OR field, as shown in the following figure.



Figure 27: Specifying the load balancer location

! **Important:** If the load balancer location is incorrect, policy enforcers cannot find the load balancer, and cannot communicate with the Control Center. This causes serious problems in the Control Center system.

5. Monitor the policy enforcers to ensure that they all receive the new profile.

Stopping policy enforcers

There are occasions when you might want to stop the execution of the policy enforcer on a particular machine without uninstalling it.

This section explains how to stop policy enforcers on various systems.

Related concepts

[Enforcer logging](#) on page 151

Related tasks

[Uninstalling policy enforcers](#) on page 113

Before you can uninstall any enforcer, you must stop it.

Stopping desktop enforcers

To stop any kind of desktop enforcer, you can locally run a special executable called **Stop Policy Controller**. You launch this from the **Start > All Programs > NextLabs** menu on the host where the enforcer is running, as shown in the following figure.

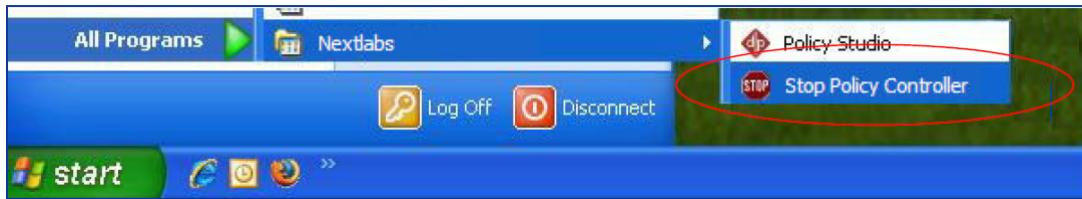


Figure 28: Stopping desktop enforcers

To run this executable, you need to provide the administrative password, which is set in the policy enforcer profile. Therefore, you must know which profile is being used on the machine so you can have the appropriate password ready.

Stopping file server enforcers and SharePoint enforcers

The executable that stops a Windows file server enforcer or a SharePoint enforcer is launched from Enterprise Data Protection group in the Windows **Start All > Programs** menu, as shown in the following figure. As with desktop enforcers, the executable prompts you for the password of the profile assigned to the enforcer that is being stopped.

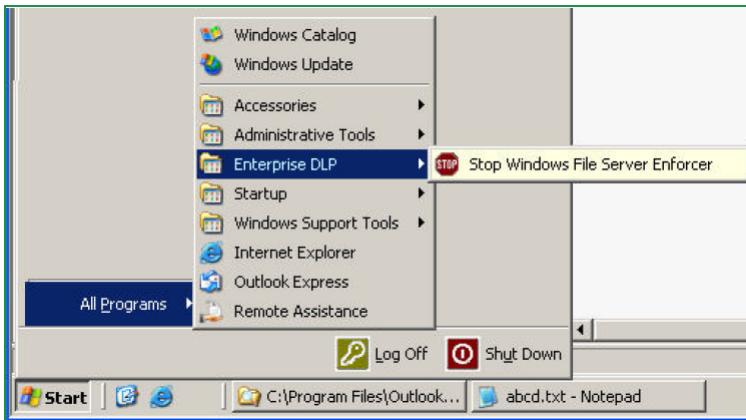


Figure 29: Stopping file server enforcers

1. Log into the local host and type the following command:

```
cefse_stop
```

2. Provide the superuser administrator's password when prompted.

Confirming policy enforcer status

After the executable runs, a notification window appears stating that the process was successfully terminated.

You can verify that the policy enforcer software has stopped as follows:

- All enforcers, remotely: Use the Administrator console to view the list of deployed policy enforcers. The policy enforcer you stopped appears in the list, but should not be sending heartbeats. See the Viewing policy enforcer status section.
- Desktop enforcers, locally: The CE icon appears in the Windows system tray, even after the enforcer is stopped. Right-click the icon, and select **About Desktop Enforcer**; the About screen should display this message: **Enforcer Status: Not Running**.
- Windows File Server and SharePoint enforcers, locally: Open the Services window using **Control Panel > System and Security > Administrative Tools**, and check the status of the service in the list on the Extended tab.
- Linux File Server enforcers, locally: To check the status of a Linux File Server enforcer, log in locally and type the following command:

```
cefse_status
```

Note: There are only two possible statuses: Stopped or Running. The enforcer should be running at all times, unless it has been manually stopped by an authorized administrator.

Related tasks

[Viewing policy enforcer status](#) on page 114

You can view the policy enforcer status in the **Administration > System Settings > Policy Enforcer Status** section.

Restarting policy enforcers

To restart a Windows enforcer of any type, open the Services window using **Control Panel > System and Security > Administrative Tools**, and restart the corresponding service. As with any Windows service, you must have local administrator privileges to do this. You can also restart enforcers by rebooting the host machine.

To restart a Linux File Server enforcer, log into the local host and use the following command:

```
service cefse start
```

Service user account permissions

All enforcers run as Windows services and are assigned a default user account at installation. This account must have read, execute, and create permission for the folder where the enforcer is installed. For example, by default, the Windows Desktop Enforcer is installed in `Program Files\NextLabs\Desktop Enforcer`, and is assigned the Local System user account, as shown in the following figure.

Services (Local)					
Control Center Enforcer Service	Name	Description	Status	Startup Type	Log On As
Computer Browser	Maintains an ...	Started	Manual	Local System	
Control Center Enforcer Service	Control Cent...	Started	Automatic	Local System	
Control Center Policy Server	Control Cent...	Started	Automatic	Local System	
Credential Manager	Provides secu...		Manual	Local System	

Figure 30: Enforcer service user account

- !** **Important:** Do not change the permission levels of the install directory, wherever it may be, or of the services user accounts, in such a way that the account does not have read, execute and create permission for the installation folder. If you do this, the account cannot access the service, and it does not restart automatically if it ever stops. For example, if you change the Local System user permission to extend only to Program Files without any child directories, the Windows Desktop Enforcer service cannot restart.

Uninstalling policy enforcers

Before you can uninstall any enforcer, you must stop it.

If the policy enforcers were installed using Group Policy Object (GPO), you can remove them using the same method. In the list of machines that are signed up for automatic managed installation, remove the names of the machines where you want to uninstall the software. The next time any of those machines restarts, a message appears stating that policy enforcer software is being uninstalled. When the uninstallation is complete, the user is allowed to log in.

If you want to remove the policy enforcer from a single machine, you can do so using the Add/Remove Programs functionality of the Windows Control Panel. During program removal, you are prompted for the policy enforcer administration password. This password is set as part of the policy enforcer profile. Before you can uninstall, you must know which profile is being used on the machine so you can have the appropriate password ready.

- !** **Important:** If the policy enforcer was installed via GPO, be sure to also delete the machine name from the GPO installation list. Otherwise, the enforcer is reinstalled automatically the next time the machine is rebooted.

To verify that the policy enforcer software has been uninstalled, use the Administrator console to view the list of deployed policy enforcers. The policy enforcer you uninstalled still appears in the list, but should no longer send heartbeats.

Related concepts

[Stopping policy enforcers](#) on page 111

There are occasions when you might want to stop the execution of the policy enforcer on a particular machine without uninstalling it.

Reconfiguring policy enforcers

Occasionally, you might need to move a desktop or file server enforcer from one Control Center to another, such as when moving a PC from one domain to another.

To do this, you need to reconfigure the enforcer so that it connects to a different ICENet server. To manually redirect an enforcer, use the following procedure.

1. Stop the enforcer you want to reconfigure.
2. On the enforcer host, delete the following four files from the install directory, as shown in the following figure:
 - bundle.bin
 - \config\registration.info
 - \config\security\agent-truststore.jks
 - \config\security\agent-keystore.jks

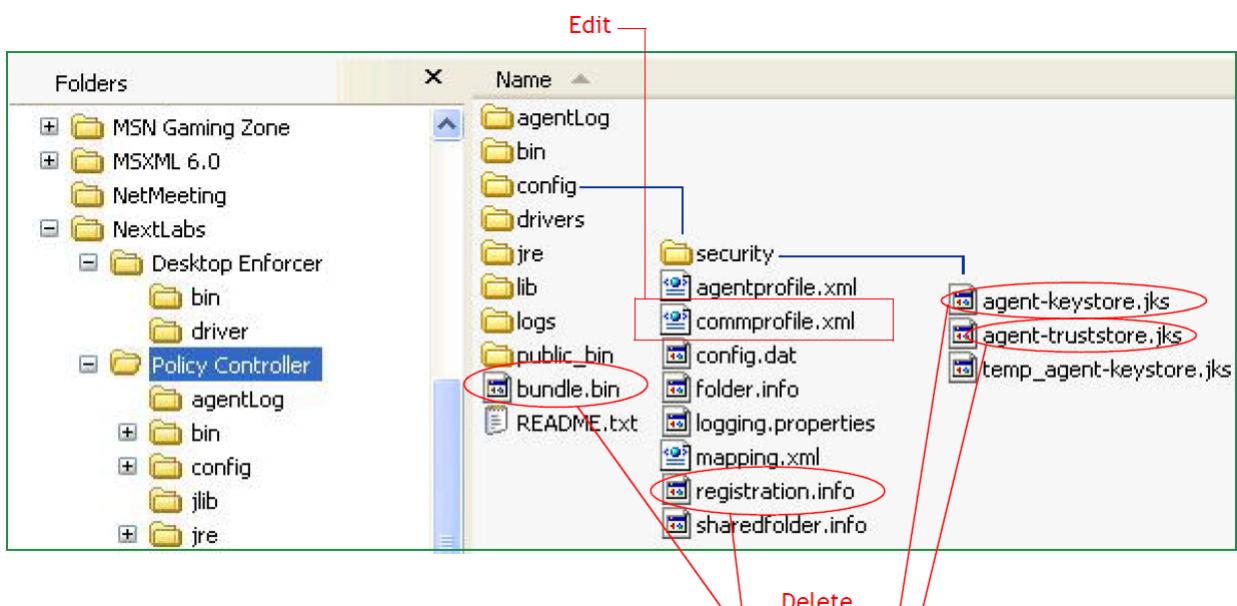


Figure 31: Configuring an enforcer to a new ICENet server

3. Open the configuration file \config\commprofile.xml, and find the DABSLocation element.
4. Replace the hostname within this element with the name of the new ICENet host you want to connect to. For example, change:

```
<DABSLocation value="https://Grande:8443/dabs"/>
```

to

```
<DABSLocation value="https://Guernsey:8443/dabs"/>
```

5. Restart the enforcer service.

Viewing policy enforcer status

You can view the policy enforcer status in the Administration > System Settings > Policy Enforcer Status section.

The Policy Enforcer Status section displays status information for each file server and desktop PC where a policy enforcer is running, as shown in following figure. The enforcers, desktops, and servers displayed depend on which products are installed. For example, NextLabs Entitlement Management products require server enforcers, and NextLabs Enterprise Data Protection products require desktop enforcers.

Figure 32: The policy enforcer status screen

1. Check the Status Summary indicator in the Status column.
 - Green: All policy enforcers are operating normally.
 - Warning (exclamation point icon): One or more of the policy enforcers listed has not sent a heartbeat in the past 24 hours. This does not necessarily indicate a problem, since certain policy enforcers, such as those on laptop computers used by remote personnel or computers that are turned off when not in use, might not connect as often as every 24 hours.
2. Optionally, you can select a filter in the Show drop-down list, to filter the rows displayed. Available filters include:
 - All Policy Enforcers
 - All Desktop Enforcers
 - All File Server Enforcers
 - All Portal Enforcers
 - All Active Directories

In addition, you can select Enforcers with warnings to display only the enforcers, of the type selected above, that have a warning status.

For example, if the Status Summary shows a Warning icon, and you want to find out which policy enforcers are triggering the warning, choose All Policy Enforcers and select Enforcers with warnings.

3. Read the detailed information about each policy enforcer in the list. The following table summarizes all information given for each policy enforcer:

Table 22: Information on policy enforcer status

Column	Description
Status	The current status of this enforcer, which is either of the following: <ul style="list-style-type: none"> • Green = Clear: the policy enforcer is sending normal heartbeats. • Exclamation point = Warning: the policy enforcer has not sent a heartbeat in the last 24 hours.
Host	The name of the machine where the policy enforcer is installed.
Type	The enforcer type: File Server, Desktop, or Portal.

Column	Description
Last Heartbeat	The time stamp of the last heartbeat generated by this enforcer. If the enforcer is running normally, this time corresponds to the configured heartbeat interval. However, a late heartbeat does not necessarily indicate a problem, since certain policy enforcers—in particular, those on laptop computers used by remote personnel or computers that are turned off when not in use—might not be able to send a heartbeat for an extended period of time even though they are operating normally.
Last Policy Update	The most recent deployment of a new or modified policy or component to this enforcer.
Policy Up to Date	Whether the enforcer has received the latest version of policies targeted for deployment to it. A check mark appears if the enforcer is up to date.
Profile Name	The enforcer profile assigned to this host. This profile determines behavior such as logging and heartbeat frequency.
Hide	Remove the host from the display. This is useful when the enforcer software has been uninstalled, and you no longer need to monitor that host. If a enforcer is ever reinstalled on this host, the host reappears on the list. If you click Hide on a host with an active policy enforcer, it reappears automatically the next time the enforcer sends a heartbeat.

Related concepts

[Confirming policy enforcer status](#) on page 112

After the executable runs, a notification window appears stating that the process was successfully terminated.

[Managing policy enforcer profiles](#) on page 116

Managing policy enforcer profiles

All policy enforcers are governed by a number of configuration settings that control such aspects as logging behavior, heartbeat rate, the tamper-prevention password, and network configuration. These are assigned default values when you first install an enforcer, but they can be changed at any time.

To simplify this, administrators can create named sets of configuration settings that can be assigned to one or more enforcers in the network. These are referred to as enforcer profiles, and they can be managed under the Policy Enforcer Configuration tab. The controls on the Policy Enforcer Configuration tab are used to define and manage policy enforcer profiles, as shown in the following figure.

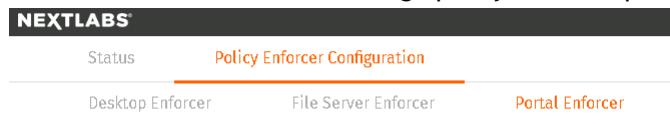


Figure 33: Policy Enforcer Configuration tab and sub-tabs

The following figure shows the default profile for the desktop enforcer.

Figure 34: The default profile for the desktop enforcer

On the File Server Enforcer page, profiles make no distinction between Windows- and Linux-based enforcers; you can assign the same profile to enforcers on either platform.

For each enforcer type, one default profile containing all default settings is available after installation, and is automatically applied to any enforcer you install. You can create additional profiles and assign them to various enforcers as needed. This is not mandatory, but it is strongly recommended that you at least change the administrative password, as a security precaution.

Administrators can work with enforcer profiles in several ways, including defining them, changing which enforcers belong to them, modifying them, and deleting them.

You can also define enforcer profiles to simplify the configuration process, and ensure that enforcers have the same configuration.

Related tasks

[Viewing policy enforcer status](#) on page 114

You can view the policy enforcer status in the **Administration > System Settings > Policy Enforcer Status** section.

Default profiles and passwords

If enforcer profiles are not explicitly defined, all enforcers in the system are automatically assigned the settings of the default profile that is automatically created in the Administrator console.

Although you do not technically need to change any settings, as a security precaution NextLabs recommends that you change the default password, which is password, for this default profile. This is the password you use to stop or uninstall any enforcer.

In addition, if there is more than one ICENet server in the system, one of those servers is assigned to each enforcer by default. It is strongly recommended that you examine the default profile, to confirm that the assigned ICENet server is the one you want to use.

You define enforcer profiles on the Policy Enforcer Configuration tab in the Administrator console. For details, see the About the console section.

Related reference

[About the console](#) on page 11

The web-based console of the Control Center enables administrators to manage policies, users, system configuration, and reports from a single interface.

Defining policy enforcer profiles

1. Specify which type of profile to create by selecting one of the three sub-tab links: Desktop Enforcer, File Server Enforcer, or Portal Enforcer.
2. In the left pane of the screen, click **New**.
3. Assign a descriptive name to the new profile.
4. On the Settings page in the right pane, specify the settings for the new profile. Typically, the settings that are most useful in distinguishing one profile from another are the ICENet server and the administrative password. The following table provides descriptions of all the settings.

Table 23: Enforcer Profile Settings

Setting	Description
ICENet Server	<p>Specifies the ICENet Server that this profile is associated with. That is, this profile is available to all the clients of the ICENet Server selected here.</p> <p>Choose one of the following:</p> <ul style="list-style-type: none"> From the drop-down list, select the URL where the ICENet Server component is installed. If you have set up load balancing or a failover cluster for the ICENet Server, type the URL of the load balancer or cluster in the OR field.
Heartbeat Frequency	<p>Specifies the frequency, in the time units selected, of the heartbeats the policy enforcer sends to the server. A heartbeat is a signal that lets the server know the policy enforcer is running and provides an opportunity to download a policy deployment or new policy enforcer profile to the policy enforcer.</p> <p>It is recommended that you enter the largest possible time increment here that meets your business requirements. This value should be set to at least one minute or more.</p>
Audit Log Upload Frequency	<p>Specify the interval, in minutes, at which policy enforcers send their activity logs to the server. The default value is 30.</p> <p>Frequent log uploads might affect the speed of your network. It is recommended that you enter the largest possible time increment here that meets your business requirements. This value should be set to at least one minute or more.</p>
Max Log Size	<p>Specify the maximum allowed size, in MB, of the current policy enforcement log is allowed to become. Default value = 2</p> <p>If the log reaches the Max Log Size before the scheduled upload time, logs are uploaded immediately, unless the machine is not connected to the network. In that case, logging is suspended, while policy enforcement continues.</p>
Enable Push	Select this option to enable Push Deployment of policy bundles to all enforcers using this profile.
Default Port	Specifies the default client-side port used for push bundle deployment, if enabled. Set to 2000 by default. If enabled, this port should not be blocked by firewalls on any enforcer hosts using this profile.

Setting	Description
Administrative Password	<p>Assign a password to tamper-protect policy enforcers installed under this profile. This password is required to stop or uninstall any policy enforcer covered by this profile.</p> <p>For both the Desktop Default Profile and the File Server Default Profile, and whenever you create a profile, this password is initially set to “password”. You should replace this default before you actually apply any profile to enforcers.</p>

 **Note:** If you are setting up a profile for a global audit, be sure to set the Document Activity Audit Level to Extended.

5. After you have configured settings, click **Save**.
6. On the Hosts tab, select all enforcers you want this profile to control. You can apply a profile to as many enforcers as needed.
7. When you have finished selecting hosts, click **Save**.

Related concepts

[About ICENet servers and policy enforcer profiles](#) on page 107

[About push deployment](#) on page 119

For any profile, you can enable the Push Deployment feature for all enforcers using that profile.

About push deployment

For any profile, you can enable the Push Deployment feature for all enforcers using that profile.

When this feature is enabled, policy analysts can use the Deploy Immediately option when deploying policies. With this feature, policies are sent without waiting for a heartbeat signal from the enforcers. In standard (non-Push) deployment, the Control Center waits until it receives a heartbeat signal from each enforcer, then checks if there are any policy bundles to send it, then sends them. Since policies are only deployed in response to a heartbeat, and the heartbeat interval may be set in hours or even days for some profiles, this can lead to unacceptably long (and unexpected) delays before new or updated policies go into effect. This feature is useful in test environments where you need to check the results of a policy right away.

For any enforcers with this feature enabled, the default port, 2000, must not be blocked by any firewalls on the client side. If you are running firewalls, be sure to define this as a port exception so that Control Center can connect through it.

Also, if you enable or disable this feature on an enforcer while it is running, the change does not take place until the enforcer sends its next heartbeat message, or until it is manually stopped and restarted.

Related tasks

[Defining policy enforcer profiles](#) on page 117

Using the document activity auditing level

For each profile you define, you can specify exactly which user or system activity is recorded in the Activity Journal for all enforcers governed by that profile.

You can choose one of four levels:

- Minimum: Records any attempts to tamper with the policy enforcer installed on this host, including all actions listed under Minimum Options in the Custom Journaling Options window. These events are always recorded; they cannot be disabled.
- Default: Records all events included under Minimum Options in the Custom Journaling Options window, plus those listed under Default Options.
- Extended: Records all events included under Minimum Options and Default Options in the Custom Journaling Options window, plus those listed under Extended Options.

- Custom: Records the combination of events you select in the Custom Journaling Options window. To open this window, click the **Custom** button. Check the boxes next to the actions that you want included in the log. If you leave all the boxes unchecked, the logging feature is set to Minimum.

Since the three types of enforcers work in different ways, the list of activities available to monitor is different for each enforcer type. shows the list in the Options window for each type of enforcer.

Using profiles for auditing

You can define a profile purely for the purpose of gathering data for what is referred to as a targeted audit.

To do this, simply create only one profile, select only those activities you want to audit (under Custom Journaling Options) and assign only those hosts you want to include in the audit. Or, if you prefer, you can use more than one profile, with certain actions audited on certain hosts and other actions on other hosts. You can then generate reports to analyze the data collected in the Activity Journal. The benefit of this strategy is that it requires less system resources than an unfiltered, global audit.

Removing a host from a policy enforcer profile

Every policy enforcer host in the network can belong to only one profile at any time. If you decide to change the policy enforcer settings for a particular host, you do it by simply assigning the host to a different profile.

In other words, the procedure for removing a host from a profile is the same as for assigning a host. Whenever you assign a host to a profile, it is automatically removed from any other profile to which it may have previously belonged.

Related tasks

[Deleting a policy enforcer profile](#) on page 120

You can delete any profile that is currently defined, which removes it from the Control Center system.

Modifying policy enforcer profiles

As conditions change over time, you might want to change policy enforcer profiles.

For example:

- You may want to change the security password.
- If you expand your hardware infrastructure and install policy enforcers on additional machines, and you don't want to use the default policy enforcer profiles which are assigned automatically, you must assign each of those new hosts to a policy enforcer profile.
- If you add a load balancer or configure a failover cluster for the ICENet server, you must change the ICENet server setting to specify the URL of the load balancer or cluster.

The modifications go into effect the next time the policy enforcer sends a heartbeat to Control Center.

1. Click the link for the type of policy enforcer profile you want to modify.
2. Click one of the profiles in the list to display its details in the editing pane.
3. Click the **Settings** or the **Hosts** tab, depending on what part of the profile you want to modify.
4. Modify the profile as needed.
5. Click the **Settings** tab, then click **Save**.

Deleting a policy enforcer profile

You can delete any profile that is currently defined, which removes it from the Control Center system.

When you delete a policy enforcer profile, any hosts that were assigned to that profile are reassigned to the default profile. You cannot delete the default profile.

If you simply want to remove a policy from a host, see the Removing a host from a policy enforcer profile section.

To delete a profile, select the profile and then click **Delete**, as shown in the following figure.

The screenshot shows the 'Policy Enforcer Configuration' interface under the 'Portal Enforcer' tab. In the 'Portal Enforcer Profiles' section, a profile named 'Protal Enforcer Profile 1' is selected. Below the list, there are buttons for 'New' and 'Delete'. The 'Delete' button is highlighted with a red box. To the right, there is a table with settings for the selected profile, including fields for Title, ICENet Server, Heartbeat Frequency, Audit Log Upload Frequency, Max Log Size, Enable Push (with a checked checkbox), Administrative Password, and Confirm Password.

Figure 35: Deleting policy enforcers

Related concepts

[Removing a host from a policy enforcer profile](#) on page 120

Every policy enforcer host in the network can belong to only one profile at any time. If you decide to change the policy enforcer settings for a particular host, you do it by simply assigning the host to a different profile.

Database management

This section describes the configuration tools available for configuring database access generally, and for managing and tuning the Activity Journal—the database where the Report Server maintains an archive of the data it collects.

The latter is important to the end user since it is the source of reports, and, because it is optimized to improve reporting efficiency, it requires periodical synchronizing and index regeneration.

Related concepts

[Configuring data access](#)

[About the four databases](#) on page 122

[Database management operations](#) on page 125

As the Report Server gathers data on user activity and policy enforcement, it temporarily stores it in OLTP tables before writing it to the Activity Journal database, where it is available for generating reports.

[Best practices](#) on page 130

The following section provides some guidance on how to use the configuration tools described above, to maximize the performance of your Activity Journal database, and of report generation.

[Other database tools](#) on page 132

In production environments, you should use an external database for all required data stores. Oracle, SQL Server, and PostgreSQL are supported; the installation default is PostgreSQL.

Configuring data access

The `configuration.xml` file also contains settings for connecting to the four data stores required by Control Center.

- Management Database
- Policy Master
- Activity Journal
- Information Network Directory

Depending on your configuration, all four may either use the internal database, or connect to an externally hosted Oracle, PostgreSQL, or SQL Server database. Each of these connections is controlled by a group of parameters, in one of the four <repository> sections of the file.

About the four databases

The four Control Center databases include:

- **Management Database:** The Management Database contains runtime information such as which components are running, how many policy enforcers have registered with the server, etc. The connection to this database is defined in the `management.repository` section of the file.
- **Policy Master:** The Policy Master contains policy component definitions and policy definitions created by Control Center users. The connection to this database is defined in the `policyframework.repository` section of the file.
- **Activity Journal:** The Activity Journal contains user events, policy enforcement data, and notifications, which are used by the Reporter application. This is always the same location as the Information Network Directory. The connection to this database is defined in the `activity.repository` section of the file.
- **Information Network Directory** The Information Network Directory stores information about directory entities—users, hosts, groups, and so on—that have been enrolled into the Control Information Network Directory. The Information Network Directory stores information about directory entities—users, hosts, groups, and so on—that have been enrolled into the Control Center system. This is always the same location as the Activity Journal. The connection to this database is defined in the `dictionary.repository` section of the file.

Related concepts

[Database management](#) on page 121

This section describes the configuration tools available for configuring database access generally, and for managing and tuning the Activity Journal—the database where the Report Server maintains an archive of the data it collects.

About connection pools

The four repository sections contain just a few settings—basically just the <name>, and a pointer to a connection pool section, where the actual connection information for that repository is stored.

Accordingly, there are four <ConnectionPool> sections, one associated with each of the repositories. The following figure illustrates this relationship.



Figure 36: Configuring data access: <Repositories> and <ConnectionPools>

Each <ConnectionPool> section contains six elements, whose functions are explained in the following table.

Table 24: Connection pool settings

Element	Description
Name	The name of the connection pool. This name should match the ConnectionPoolName setting in one of the <Repository> sections.
Username	The username of an account that can access the database specified in the Name property. This property is initially set during installation. This property can be changed as needed.
Password	The password associated with the account specified in Username. The value must be encrypted. For information about how to change the password after it has been set, see the Encrypting passwords section.
ConnectionString	Database connection information (host and port) in either Oracle or SQL Server format, depending on how the Control Center was installed. The host and port are initially set during installation, but you can change the machine and port in the connect string at any time after installation—for example, if you move the database server to a different machine.

Element	Description
DriverClassName	The name of the database driver used by this connection. Valid values are: Oracle: <code>oracle.jdbc.driver.OracleDriver</code> SQL Server: <code>com.microsoft.sqlserver.jdbc.SQLServerDriver</code>
MaxPoolSize	The maximum number of connections that are simultaneously available for accessing this data source from a given Control Center instance. See the Adjusting connection pool size section.

Related concepts

[Adjusting connection pool size](#) on page 124

The last element, <MaxPoolSize>, specifies the maximum number of simultaneous data connections available for each of the four databases.

Related tasks

[Encrypting passwords](#) on page 144

Whenever you generate a new password for the configuration file, you should encrypt it.

Adjusting connection pool size

The last element, <MaxPoolSize>, specifies the maximum number of simultaneous data connections available for each of the four databases.

By default, the values for this setting are not the same for each of the four connection pools. They are:

- `management.connection.pool` (for the Management DB): 8 connections
- `policyframework.connection.pool` (for the Policy Master): 8 connections
- `activity.connection.pool` (for the Activity Journal): 20 connections
- `dictionary.connection.pool` (for the Information Network Directory): 14 connections

The sum of these connections is 50, which is half the maximum supported by Control Center's internal PostgreSQL database. This is based on the common installation architecture of one physical host for the ICENet Server, and one for all other Control Center components: two server hosts x 50 connections apiece, matches the database's own default limit.

The way these 50 are divided among the four pools is based on the relative amounts of database activity you can anticipate for each repository, during normal use. If you need to adjust these settings you can do so by editing these connection pool definitions, but remember that you can only reallocate them from one pool to another. That is, if you increase the max for one pool, you should decrease it for another, so that the total does not exceed 50.

Related concepts

[About connection pools](#) on page 122

The four repository sections contain just a few settings—basically just the <name>, and a pointer to a connection pool section, where the actual connection information for that repository is stored.

Minimum pool size

There is a minimum as well as a maximum for each connection pool, but it is not configurable.

It is set to 10% of the maximum, rounded up to the next whole number. For example, if a max size is 18, the minimum is 2; if a max size is 24, the minimum is 3.

When to adjust

Whenever the operational requirements for one of the repositories begins to exceed the number of connections allocated, you need to reallocate connections to prevent database timeouts.

Database operations that time-out produce the following error message in the system log:

```
java.sql.SQLException: An attempt by a client to checkout a Connection has timed out.
```

The database connection bottleneck

Maximum connection values apply per each server instance, rather than to the Control Center as a whole.

If you install multiple ICENet Servers, each server has the same default number of connections to access each given repository. Therefore, the sum total of maximum available connections to a given data source is ($N * \text{MaxPoolSize}$), where 'N' is the number of ICENet Servers you are running. This does not mean that all these connections are used; some ICENet Servers might not use all the provided data sources. The total connections will be 50 times the number of servers you deploy.

However, the databases themselves support only a finite, maximum number of connections. If you are using an external PostgreSQL or Oracle database, it may support a different number. This means that if you deploy multiple ICENet Servers, their combined connections might exceed the actual capabilities on the database side. In such cases, the system log shows error messages such as:

```
org.postgresql.util.PSQLException: Backend start-up failed: FATAL: sorry, too many clients already.
```

If you are using an external database, you should know what its connection maximum is, and make sure that you do not exceed it by the aggregate connection pools of all ICENet servers in the system. You can do this either by increasing it on the database side, or reducing the `configuration.xml` connection pool sizes to something below the default levels.

Changing database connect strings

If you change the database name portion of the connect string, be extremely careful.

This part of the connect string must match the name of the database itself.

Database management operations

As the Report Server gathers data on user activity and policy enforcement, it temporarily stores it in OLTP tables before writing it to the Activity Journal database, where it is available for generating reports.

The following figure illustrates this activity. During this activity, the Reporter Server needs to synchronize the source and target tables to keep the data accurate. By default, sync operations are performed automatically according to settings in the `configuration.xml` file, and you can change these settings as needed.

Whenever this sync operation is performed, it fragments the table indexes of the Activity Journal, which are required for efficiently generating reports. For this reason, these tables need to be reindexed periodically. This operation is also performed automatically, and is controlled by settings in the configuration file.

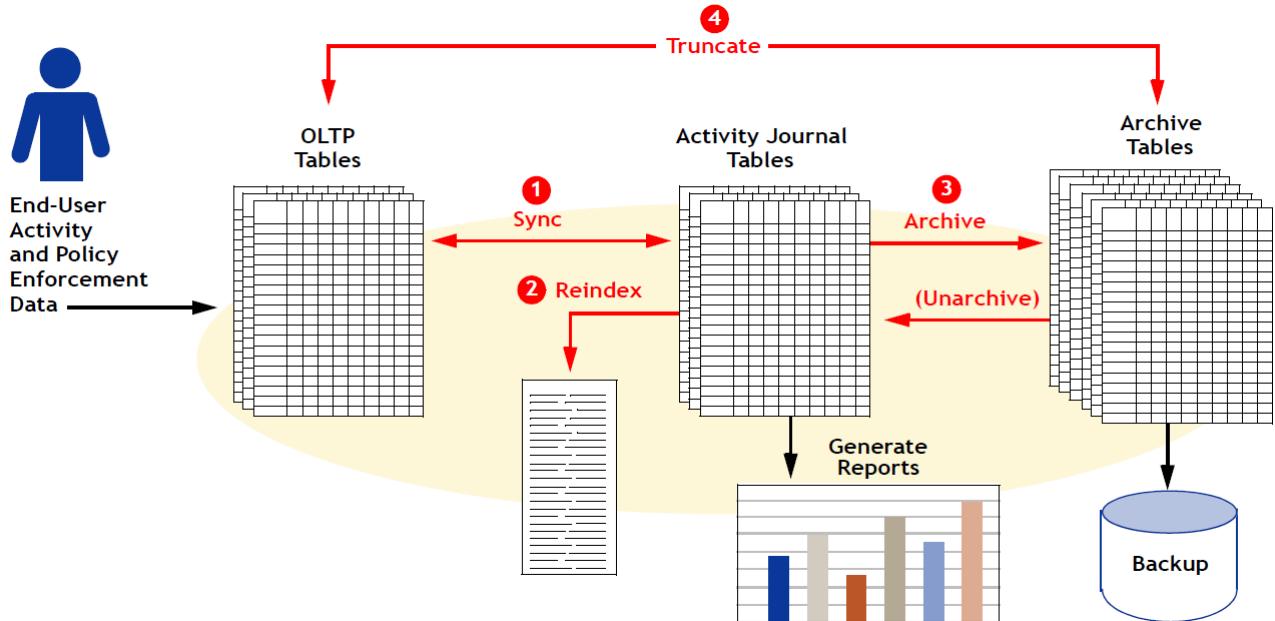


Figure 37: Activity data handling procedures

After a specified interval, 90 days, by default, all Activity Journal data is automatically moved to archive tables to improve efficiency for users generating activity reports. The archiving operation is configurable. The archive tables are stored in the same database as the Activity Journal, but they should be manually backed up periodically and deleted from the main database to ensure optimal performance.

The first three of these operations—syncing, reindexing, archiving—are controlled by a set of elements in the `<DAC> <ActivityJournalSettingConfiguration>` section of the `configuration.xml` file. The archive truncation procedure must be performed manually, using a supplied SQL script, and only after backing up.

Related concepts

[Configuring sync operations](#) on page 126

By default, the table sync operation is set to occur every 15 minutes, counting from the completion of the previous sync operation.

[Configuring Control Center components](#) on page 140

A set of sections in `configuration.xml` control the various components of the Control Center.

[Activity Journal archive considerations](#) on page 220

By default, the Activity Journal stores data for 90 days before archiving it, but this setting can be changed by the database administrator.

[Database management](#) on page 121

This section describes the configuration tools available for configuring database access generally, and for managing and tuning the Activity Journal—the database where the Report Server maintains an archive of the data it collects.

Configuring sync operations

By default, the table sync operation is set to occur every 15 minutes, counting from the completion of the previous sync operation.

You can change this interval so that the system syncs less frequently, or you can set the system to sync only once a day at a specific time, rather than on an interval basis. Also by default, the duration of the sync operation is limited to two hours, but you have the option of increasing or decreasing this value. As a baseline for planning, on recommended hardware the operation can be expected to synchronize one million log entries in about ten minutes.

Sync operation behavior is configured by three elements in the <DAC> <ActivityJournalSettingConfiguration> <SyncOperation> section of the configuration.xml file, as shown in the following figure.



Figure 38: Configuring table synchronization

- **TimeInterval:** The time interval, in minutes, between sync operations. At the default 15, the system synchronizes 15 minutes after the server starts, then again 15 minutes after that sync operation finishes, and so on. Each sync operation may itself take more or less than 15 minutes; this interval represents the time between the completion of one and the beginning of the next. Valid values = any positive integer.
- **TimeOfDay:** The sync operation to occur once a day at a specific time, rather than on an interval basis. This parameter is commented out by default; to use this option you must uncomment it, supply a time (or keep the default value, 23:30), and also comment out the TimeInterval parameter. That is, the system can only work one way or the other, and one parameter or the other must be commented out. Valid values = colon-delimited time in 24-hr. format, either with seconds or without (for example, 23:30 or 23:30:00 are both valid).
- **TimeoutInMinutes:** The maximum time, in minutes, that each sync operation can take. If this limit is reached, the sync operation terminates and any remaining data is synced during the next operation. Valid Values = any positive integer.

Related concepts

[Database management operations](#) on page 125

As the Report Server gathers data on user activity and policy enforcement, it temporarily stores it in OLTP tables before writing it to the Activity Journal database, where it is available for generating reports.

Configuring index rebuilding

Reindexing behavior is configured by four elements in the <DAC>

<ActivityJournalSettingConfiguration> <IndexesRebuildOperation> section of the configuration.xml file.

- **TimeOfDay:** The time the reindexing operation begins, on the day or days specified in the DaysOfWeek section. Valid values: Colon-delimited time in 24-hr. format, either with seconds or without (that is, either HH:MM or HH:MM:SS). If you set the TimeOfDay parameter for the sync operation, you should schedule the reindexing operation to occur after the sync operation completes. This ensures that indexes are rebuilt with the latest data.

- **DaysOfWeek:** The days of the week the reindexing operation is scheduled to occur, beginning at the time specified by the **TimeOfDay** element. You can specify more than one day of the week by adding a **<DayOfWeek>** element for each. Valid values = full name of day or three-letter abbreviation, must be in uppercase characters (such as SUNDAY, SUN, TUESDAY, TUE). Values are locale-specific. The value or values that you specify must match the regional format of the machine where Control Center is installed.
- **AutoRebuildIndexes:** The status of the reindexing feature. The feature is enabled when set to the default, True. Valid values: True, False.
- **TimeoutInMinutes:** The maximum time, in minutes, that each reindexing operation can take. If this limit is reached, the operation terminates at the completion of the index then in progress, and any remaining indexes remain fragmented until the next operation. Valid values: Any positive integer.



Figure 39: Configuring index rebuilding

Configuring data archiving

By default, the Report Server automatically runs a data archive operation once a day at 1:30 AM, and moves all data older than 90 days from the Activity Journal tables to the Archive.

After the data is sent to the Archive, the data is no longer available for displaying in Reporter. However, it can be unarchived if required for reporting purposes. As a baseline for planning, on recommended hardware, the operation can be expected to archive one million log entries in about fifteen minutes.

Auto-archiving behavior is configured by the four elements in the **<DAC>**

<ActivityJournalSettingConfiguration> **<ArchiveOperation>** section of the **configuration.xml** file as shown in the following figure.

- **TimeOfDay:** The time the auto-archiving operation is scheduled to begin each day. The frequency of this daily operation is not configurable. Valid values = colon-delimited time in 24-hr. format, either with seconds or without (that is, either HH:MM or HH:MM:SS). If you set the **TimeOfDay** parameter for the sync operation, you should schedule the archive operation to occur after the sync operation. This ensures that the archive contains the latest data. If you also schedule the reindexing operation, scheduling the archive operation after the reindexing operation reduces the load on the database.

- **DaysOfDataToKeep:** The number of days that data is kept in the Activity Journal tables before being automatically archived. The default is 90. Valid values: Any integer.
- **AutoArchive:** The status of the auto-archiving feature. The feature is enabled when set to the default, True. Valid values: True, False.
- **TimeoutInMinutes:** The maximum time, in minutes, that each auto-archiving operation can take. If this limit is reached, the operation terminates at the completion of the table then in progress, and the old data in any remaining tables is archived in the next day's operation. Valid values: Any positive integer.



Figure 40: Configuring data archiving

Related concepts

[Unarchiving data](#) on page 129

When data is moved from the Activity Journal to the Archive, it is no longer available for generating reports using the Reporter queries—these reflect only whatever data is in the Activity Journal tables.

Unarchiving data

When data is moved from the Activity Journal to the Archive, it is no longer available for generating reports using the Reporter queries—these reflect only whatever data is in the Activity Journal tables.

However, archived data can be unarchived at any time, at which time it becomes available for reporting. This operation should be performed only by an experienced database administrator.

You unarchive data based on a specified time range—for example if your Activity Journal only dates back to December 1, you can unarchive additional data from November 1 through November 31.

To unarchive data, a database administrator can use the Unarchive SQL scripts specific to the database type, shown in the following table. These scripts are included in the Control Center install directory, under \tools\datasync\sql. These scripts must be edited manually to specify the date range to unarchive.

Table 25: SQL Scripts for database administration

DB Type	Unarchive scripts	Truncate script
Microsoft SQL Server	unarchive-pa-mssql.sql	truncate-mssql.sql
	unarchive-ta-mssql.sql	
PostgreSQL	unarchive-pa-postgres.sql	truncate-postgres.sql
	unarchive-ta-postgres.sql	
Oracle	unarchive-pa-oracle.sql	truncate-oracle.sql
	unarchive-ta-oracle.sql	

There are two scripts for each operation, marked by PA and TA. The PA script handles data related to policy activity, and the TA one handles tracking activity data—that is, data on end-user activity that is not necessarily related to any policy evaluation or enforcement. You can run either script by itself if you only need reports on one kind of activity or the other, or you can run them both to unarchive both types.

Related concepts

[Configuring data archiving](#) on page 128

By default, the Report Server automatically runs a data archive operation once a day at 1:30 AM, and moves all data older than 90 days from the Activity Journal tables to the Archive.

Archive truncation

The fourth database management function involves truncating the archive tables, to prevent them from growing too large.

This can be done using the truncate script appropriate for your database type. When this script runs, it permanently drops all data from the archive tables. This means the process should only be performed after running a standard backup, so the archive data is preserved to another storage location. This backup can conform to your database administrator's standard procedure.

Once the archive tables have been backed up, you can run the truncate script. There is no requirement to edit this script before running it.

The truncate scripts not only drop all data from the archive tables, but also delete unneeded rows from the OLTP tables. Because of the importance of both the backup and truncate operations, they should be performed only by an experienced database administrator. On recommended hardware, the truncate operation should take a very short time—less than one minute—to complete.

Table names

Although none of the operations require that you specify individual table names, it may be useful for administrators to know that the schemas for the three sets of tables.

OLTP, Activity Journal, and Archive are basically the same, but can be identified by table name. The OLTP tables are not prepended; Activity Journal tables have the string `rpa_` prepended to their names, except for `report_obligation`; and the Archive tables begin with the string `apa_`, except for `archive_obligation`.

Best practices

The following section provides some guidance on how to use the configuration tools described above, to maximize the performance of your Activity Journal database, and of report generation.

Related concepts

[Database management](#) on page 121

This section describes the configuration tools available for configuring database access generally, and for managing and tuning the Activity Journal—the database where the Report Server maintains an archive of the data it collects.

Determining logging volume

To determine whether you need to reset any of the configuration parameters that control database management, you need to have some idea of how much log data your system is accumulating over time.

The simplest way to do this is through the Administrator console, whose main System Status tab displays (in the lower left corner) the System Statistics related to the Activity Journal. These represent events written to the system logs, and so the number gives you the actual number of log events—that is, database rows—generated in the last 24-hour period.

Setting sync times

As a rule, set the sync operation time interval to be as long as possible, given your reporting requirements.

There is an inherent trade-off between operational efficiency and the freshness of reporting data. This is because the sync operation fragments Activity Journal data indexes, and the more fragmented the indexes, the longer it takes to reindex them. In short, the more frequent the sync, the more up-to-date the data, but the longer it takes to run reports.

The default behavior of syncing frequently throughout the day ensures that all reports include the most recent data, but the frequent sync operations to place a processing load on the reporter server. If you are generating large amounts of activity data, set the sync interval to several hours, or once or twice a day, either by using the TimeOfDay parameter, or by setting the TimeInterval parameter to 12 or 24 hours. This greatly improves performance, although reports show data from the previous day. The appropriate sync frequency depends largely on whether you want your reports to show up-to-the-hour data.

Optimizing report generation

Large accumulations of data in the Activity Journal directly affect how fast reports can be generated.

The best rule of thumb for generating reports is to keep your Activity Journal tables as small as possible. Of course, the size is dependent on the length of the reporting period. If you need to report only the previous week's data, you can maintain less data in the Activity Journal than if you need reports on the previous month's or quarter's data. Similarly, if you need to run monthly reports but you only run them once a quarter, you can archive most of your data, and then unarchive it before you run the quarterly report.

Managing database size

Although the mechanisms described in this section automatically move data from one set of tables to another, databases eventually become unmanageably large if uncontrolled.

In every Control Center installation, the database must be backed up periodically so that the archive tables can be truncated to make room for constantly accumulating activity data. How often you do this depends on the size of your implementation and the amount of activity, both policy-related and not. For most enterprises, a daily backup is more frequent than necessary, while once a month would be too seldom. Something like once a week would be reasonable, more or less depending on your actual data accumulation.

The proper approach is to keep an eye on the size of your Control Center database, run a backup and truncate operation, and note how much that reduced the overall data volume.

Weekly backup procedure

The backup step of the process described above generally involves making copies of the archive tables, so they are available in case you ever need to restore them to another database so you can run reports on them.

Each time you do this, the backup includes the archive tables from the previous backup plus all new data since then. This means that each time you back up the (archive) database, you add the latest data for all tables. Assuming you make a backup once a week (a generally recommended interval), this backup data can quickly become unmanageably large.

A better approach is to use the following steps at the time of the weekly database backup.

1. Export the data from the archive tables to a set of files. These files can be marked with a timestamp as the filename, enabling you to restore the data from specific dates if necessary. You can do this by running a script during the weekly database maintenance window.
2. Back up the database in the standard way.
3. Run the Truncate script to remove unnecessary data from OLTP tables and clear out the archive tables.
4. Shrink the database.
5. Rebuild all indexes.

By following this procedure, you ensure that your database remains at a reasonable size, while also backing up the archived data to files that can be stored on a different media if needed, where it is available for restore operations as needed.

Exporting individual tables

Exporting data from individual tables to files requires tools that are specific to the database you are using. For example, Microsoft SQL Server provides the BCP tool for this purpose.

You can also use these specialized tools to export the content of archived data on a table-by-table basis. For example, the commands shown would export the content of the archive tables for an Microsoft SQL database called “Alpo.” This operation refers to the contents of a Windows batch file that could be run as a part of Step 1, above.

1. Call BCP “alpo.dbo.archive_policy_activity_log” out c:\Temp\ apalog.bcp -n -T
2. Call BCP “alpo.dbo.archive_policy_custom_attr” out c:\Temp\ apacustlog.bcp -n -T
3. Call BCP “alpo dbo.archive_obligation_log” out c:\Temp\apaoblog.bcp -n -T
4. Call BCP “alpo.dbo.archive_tracking_activity_log” out c:\Temp\atalog.bcp -n -T
5. Call BCP “alpo.dbo.archive_tracking_custom_attr” out c:\Temp\atacustattrlog.bcp -n -T

The file names should be appended with timestamps so that files from the previous exports are not overwritten. In addition, the directory where all these files should be backed up periodically.

Other database tools

In production environments, you should use an external database for all required data stores. Oracle, SQL Server, and PostgreSQL are supported; the installation default is PostgreSQL.

Related concepts

[Database management](#) on page 121

This section describes the configuration tools available for configuring database access generally, and for managing and tuning the Activity Journal—the database where the Report Server maintains an archive of the data it collects.

Using the VacuumDB utility

If you are using PostgreSQL, it is a good idea to clean up the database periodically, by running a utility called VacuumDB.

The more you use Control Center, the more often you should use this utility—for example, anywhere from once a week to once a month.

The Control Center installation package includes the vacuum executable, which is installed by default in the following location:

```
<InstallDirectory>\PolicyServer\Repository\bin\vacuumdb.exe
```

The recommended options to use are:

```
vacuumdb -a -U root -W <Administrator password>
```

You must supply the password of the Administrator account.

For more information about this utility, see the PostgreSQL online help, which you can open from the command line by typing:

```
vacuumdb.exe -?
```

The dbInit utility

The dbInit.bat utility included with Control Center can be used to upgrade the database for an installation of Control Center.

Using this utility is not required, but it offers several functions that may be of interest to database administrators coordinating with the Control Center administrators during installations or upgrades. This utility is installed in the following location:

```
<InstallDirectory>\PolicyServer\tools\dbInit
```

You can launch this utility by opening a command window, navigating to that location, and using the command dbInit. The following table lists all arguments available for this command. You can also view this information by typing the command:

```
dbInit -h
```

The dbInit utility should only be used by knowledgeable database administrators. For more information, contact NextLabs Technical Support.

Table 26: Available commands for the DBInit utility

Argument	Description
[-h]	Display a list of all commands and arguments. Cannot be used with any other arguments.
-i, -install	Install a new database.
-u, -upgrade	Upgrade from older installed database. Requires the following two arguments.
-f, -fromV <1.6 2.0 2.5>	Specifies upgrade From version, required only with Upgrade.
-t, -toV <1.6 2.0 2.5>	Specifies upgrade To version, required only with Upgrade.
-createschema	Generates a SQL statement describing the new schema, to a file specified by the -s argument, which is required. Used in the Install process.
-dropcreateschema	Generates a SQL statement describing the new schema, to a file specified by the -s argument, which is required, and also drops all the tables in the old schema. Used in the Upgrade process.
-updateschema	Generates a SQL statement describing any changes in new schema, to a file specified by the -s argument, which is required. Does not drop the tables in the old schema, but only updates them. Used in the Upgrade process.
-processsql	Imports a SQL statement from the file specified by the -s argument, which is required, and creates the schema based on that. Used in the Upgrade process.
- s -schema <schema path>	Specifies the path and name of the to schema file used by the above four arguments (either as output or as input)
-c -config <config path>	Specifies the path to the path of Java classes used for importing Seed Data.

Argument	Description
<code>-C -connection <configuration.xml path></code>	Specifies the path to the configuration files, required by the database.
<code>-L, -libraryPath <libraryPath></code>	Specifies the path to the mapping files required by the Hibernate function.
<code>[-q, -quiet]</code>	Activates the Hibernate function between the Java code and the database; default is false.

Configuration tools

This section describes the tools available for modifying the configuration of Control Center after it is installed.

None of the available configuration settings are strictly mandatory, in that all have default settings that you do not necessarily need to change. However, you may want to change some settings either during your initial installation, or later.

This section describes the configuration settings you might want to modify, under certain specified conditions. You should not make any changes to settings that are not explicitly described here. All configuration files contain many settings that you should not change; doing so could seriously disrupt the operation of your Control Center system.

Related concepts

[About configuration files](#) on page 134

The Control Center configuration files are XML files that contain configuration settings you can view or modify to tune your system's performance, scale the system by adding or removing modules, and address various administrative issues.

[Configuration settings](#) on page 136

The main Control Center configuration file, `configuration.xml`, is copied to the host server when you first install the Control Center.

[Tomcat settings](#) on page 146

Control Center's UI applications work through the Tomcat web-based application server, included in every Control Center installation.

[Policy Controller configuration](#) on page 150

Related tasks

[Editing configuration files](#) on page 136

You can change any aspect of your configuration.

[Adding policies](#) on page 175

About configuration files

The Control Center configuration files are XML files that contain configuration settings you can view or modify to tune your system's performance, scale the system by adding or removing modules, and address various administrative issues.

The following table describes these configuration files.

Table 27: Configuration files

Filename	Location	Description
configuration.xml	<InstallDir>\server\configuration\configuration.xml	Main Control Center configuration file.
server.xml	<InstallDir>\server\configuration\server.xml	Tomcat Application Server.

To modify the initial settings, or to access settings that are not included in the installation wizards, you can edit the configuration files directly. Each feature is associated with a section in the file, and focuses on the functions of each information element (marked by its XML tag) in that section. An element may or may not have one or more Properties sections within it, which control aspects of the corresponding feature's operation. The following figure shows an example of one of these sections.



Figure 41: Sample section from the configuration.xml file

Related concepts

[Configuring data access](#)

[Configuration tools on page 134](#)

This section describes the tools available for modifying the configuration of Control Center after it is installed.

[Configuration settings on page 136](#)

The main Control Center configuration file, configuration.xml, is copied to the host server when you first install the Control Center.

[Tomcat settings on page 146](#)

Control Center's UI applications work through the Tomcat web-based application server, included in every Control Center installation.

Editing configuration files

You can change any aspect of your configuration.

1. Open the file with a text editor.
2. Find the setting you are interested in by searching for its tag.
3. Change the value between the open and close tags.
4. Close the file, saving your changes.

Related concepts

[Configuration tools](#) on page 134

This section describes the tools available for modifying the configuration of Control Center after it is installed.

Configuration settings

The main Control Center configuration file, `configuration.xml`, is copied to the host server when you first install the Control Center.

It controls the behavior of all Control Center components throughout your system, even if they are installed on different hosts. During installation, the file is placed in the following directory:

`<InstallDir>\server\configuration\configuration.xml`

You can use this file to change the settings that control various functions of the system. These include the configurations of the following:

- The User Repository, the logical database where information on users and user groups is stored
- User authentication, external domain authentication, and trusted domain configuration
- Individual Control Center software components
- Connections between Control Center and its four data stores
- The performance of the ICENet log queue manager
- Definitions of all custom obligations
- Special actions that are specific to individual enforcers

Related concepts

[Configuration tools](#) on page 134

This section describes the tools available for modifying the configuration of Control Center after it is installed.

[About configuration files](#) on page 134

The Control Center configuration files are XML files that contain configuration settings you can view or modify to tune your system's performance, scale the system by adding or removing modules, and address various administrative issues.

[Configuring authentication](#) on page 137

Authentication is the process of verifying that a person who is trying to log in to Control Center is authorized to do so.

[Configuring the external domain authentication](#) on page 138

The `<ExternalDomainConfiguration>` section contains several properties that are used to configure information about the external domain for user and group information retrieval, and for remote authentication of users logging into Reporter and Administrator.

[Configuring trusted domains](#) on page 139

During normal operation, Control Center tracks users and hosts according to the domain where they are enrolled, for the purposes of preparing policy bundles. Control Center needs this information so that it

can include, in policy bundles, only the information that is relevant to a particular policy enforcement point.

[Configuring Control Center components](#) on page 140

A set of sections in `configuration.xml` control the various components of the Control Center.

[Configuring data access](#)

[Configuring the ICENet log queue](#) on page 145

[Registering custom obligations](#)

[Configuring the User Repository](#)

The settings that capture information about the internal directory server hosting Control Center user information are organized in the `<UserRepositoryConfiguration>` section.

The following table describes the elements in this section.

Table 28: User Repository configuration elements

Element	Description
<code>server.name</code>	The name of the machine hosting the User Repository server. This is the same as the machine hosting the DMS (Management Server).
<code>server.port</code>	The port number of the User Repository server. This is set by the installer.
<code>useSSL</code>	Specifies whether the server being accessed is SSL enabled.
<code>login.dn</code>	The distinguished name or user principal name (for example, <code>jdoe@mycompany.com</code>) of the account used to gain access to the user repository. This property is initially set through the Control Center installation wizard.
<code>login.password</code>	Password for the user in <code>login.dn</code> . This property is the same as the Administrator password, which is specified during Control Center installation. The value must be encrypted. For information about how to change the value at any time after your initial installation, see the Encrypting passwords section.

Related tasks

[Encrypting passwords](#) on page 144

Whenever you generate a new password for the configuration file, you should encrypt it.

[Configuring authentication](#)

Authentication is the process of verifying that a person who is trying to log in to Control Center is authorized to do so.

Control Center integrates with the existing security infrastructure, eliminating the need to manage user account information in multiple systems.

Related concepts

[Configuration settings](#) on page 136

The main Control Center configuration file, `configuration.xml`, is copied to the host server when you first install the Control Center.

[Choosing local or remote authentication](#)

Control Center supports the following types of authentication.

- Local authentication: Users are authenticated directly against Control Center's local Information Network Directory.

- **Remote authentication:** Users are authenticated against a Kerberos server in the organization's network.
- **Hybrid authentication:** Users are first authenticated against the manually created users within Control Center. If no such user is found or if the credentials do not match, the credentials are verified remotely (as in Remote authentication) if an imported user is found with the same login name.

The type of authentication used is controlled by the <AuthenticationMode> element, which looks like this:

```
<ApplicationUserConfiguration>
<AuthenticationMode>Hybrid</AuthenticationMode>
...
</ApplicationUserConfiguration>
```

This is initially set during installation. It is set to Local if the user skips the Application User Authentication screen during installation, and to Hybrid if the user provides the requested information in that screen. You can change it manually any time after installation by changing this value.

Configuring the external domain authentication

The <ExternalDomainConfiguration> section contains several properties that are used to configure information about the external domain for user and group information retrieval, and for remote authentication of users logging into Reporter and Administrator.

During Control Center installation, this section is commented out if the Application User Authentication screen of the installation wizard is skipped.

In this section, the name of the security domain, authentication server, and connectivity information are configured, as discussed in the following table.

Table 29: External domain configuration settings

Element	Description
<DomainName>	The name of the security domain against which users are authenticated. User login names must exist within this domain. You can specify only one domain.
<AuthenticatorConfiguration>	Contains two properties relevant to domain authentication.
java.security.krb5.kdc	Kerberos Domain Controller. The name of the primary domain controller for the domain specified in the java.security.krb5.realm property. The primary domain controller contains credentials for all web application users.
java.security.krb5.realm	The domain name. Should be the same as specified in the <DomainName> subsection described earlier, except that it must be in all capital letters.
<UserAccessConfiguration>	This element has four properties that control connectivity to the Active Directory server hosting the user and group information. These settings enable Control Center to display and enroll external users and groups from your organization's domain into Control Center.
server.name/server.port	Name and port of the Active Directory server hosting the information.
useSSL	Whether to use SSL when connecting to the Active Directory server.

Element	Description
root.dn	The distinguished name of the root node from which to start retrieval of user/group data entries.
login.dn	The principal name of a user with read privileges on the external domain.
login.password	The encrypted password for the user entered in login.dn. You can encrypt the password using the mkpassword.bat program found at C:\Program Files\NextLabs\PolicyServer\tools\crypt.

Related concepts

[Configuration settings](#) on page 136

The main Control Center configuration file, configuration.xml, is copied to the host server when you first install the Control Center.

Related tasks

[Encrypting passwords](#) on page 144

Whenever you generate a new password for the configuration file, you should encrypt it.

Configuring trusted domains

During normal operation, Control Center tracks users and hosts according to the domain where they are enrolled, for the purposes of preparing policy bundles. Control Center needs this information so that it can include, in policy bundles, only the information that is relevant to a particular policy enforcement point.

For example, policy bundles deployed to file servers in the test.widgetco.com domain need to contain users and hosts only from that domain. Ordinarily, the information about users and hosts from one domain, widgetco.com, is irrelevant to file servers in another, test.widgetco.com, because the operating system denies logon to users from widgetco.com—making it impossible to perform any further actions on file servers of the test.widgetco.com domain.

However, there is one important exception. When there is a trust relation between the widgetco.com and test.widgetco.com domains, file servers from test.widgetco.com must know about hosts and users of the widgetco.com, and vice versa. Control Center distributes policy bundles correctly across domains in such cases, but only if you inform Control Center about any such trust relationships. To do this, you use a configuration setting, <MutuallyTrusted>, found in the DABS section of the configuration file and shown in the following figure.



Note: This configuration step is required for Domain Group enrollment.

```

<DABS>
    <HeartbeatRate>30</HeartbeatRate>

    <MailServerConfiguration>
        .
        .
        .
    </MailServerConfiguration>

    <TrustedDomainsConfiguration>
        <MutuallyTrusted>publishing.widgetco.com,test.widgetco.com,widgetco.com</MutuallyTrusted>
        <MutuallyTrusted>partners.cyberdyne.com,widgetco.com</MutuallyTrusted>
    </TrustedDomainsConfiguration>

</DABS>

```

Figure 42: Sample trusted domains configuration

Each element in this section can contain two or more domains, and represents a trust relationship between or among them. Note the following syntactical points about this section:

- Domains with mutual trust relations are specified as comma-separated lists within the `<MutuallyTrusted>` tags.
- Each of these lists is logically separate: a domain with more than one trusted relationship may be included in multiple lists, without joining the lists. For example, the sample configuration above creates trust relationships between `widgetco.com` and `partners.cyberdyne.com` and between `widgetco.com` and `publishing.widgetco.com`, but it does not create a trust relationship between `partners.cyberdyne.com` and `publishing.widgetco.com`.
- Empty lists, or lists with only one domain, are allowed but ignored.
- Blank spaces around commas are ignored; however, blank spaces around dot separators produce errors.

Related concepts

[Requirements for Domain Group enrollment](#) on page 44

An additional requirement applies for Domain Group enrollment.

[Configuration settings](#) on page 136

The main Control Center configuration file, `configuration.xml`, is copied to the host server when you first install the Control Center.

Related tasks

[Enrolling Domain Groups](#) on page 65

There are a few concepts and requirements unique to Domain Group enrollment.

Configuring Control Center components

A set of sections in `configuration.xml` control the various components of the Control Center.

All instances of each software component share the configuration settings of a single section in the configuration file. For example, if you have a distributed system where several ICENet Servers share the traffic from the policy enforcers in your organization, all the ICENet components use the same configuration. The following table lists these sections and the elements and properties in each, and summarizes their functions.

Table 30: Configuring Control Center components

Section	Purpose	Element or property	Function
<DMS>	Governs the operation of the Management Server	<HeartbeatRate>	Specifies the frequency, in seconds, of the Management Server's own heartbeat signal. This value should not be changed. Minimum = 15; default = 60
<DCSF>	Governs the operation of the core server framework, which relays information across machines between components, relays license key enforcement, and performs other tasks.	<HeartbeatRate>	Specifies the frequency, in seconds, of the heartbeat signal sent to the Management Server. Minimum = 15; default = 60
<DABS>	Governs the operation of the ICENet Server, which manages communication between the Control Center and all policy enforcers.	<HeartbeatRate> <TrustedDomains Configuration>	Specifies the frequency, in seconds, of the heartbeat signal the ICENet Server sends to the Management Server. Minimum = 15; default = 60 Allows you to specify trust relationships among domains in your network. For the trusted domains listed here, policy bundles applying to a file server in one domain include user and host information for the other domain. This enables Control Center to enforce policies across domains (this happens only in the case of trusted domains).
<DPS>	Governs the operation of the Policy Management Server.	<HeartbeatRate> <DeploymentTime>	Specifies the frequency, in seconds, of the heartbeat signal the Policy Management Server sends to the Management Server. Minimum = 15; default = 60 Specifies the default deployment time for policy enforcers. Value is case-insensitive. Required syntax: [(ordinal last)] [day of week] HH24:MM

Section	Purpose	Element or property	Function
		<CustomAttributes>	For storing definitions of custom attributes for components. This section is not present until it is explicitly added, to define custom properties.
<DAC>	Governs the operation of the Intelligence Server, which is the back-end server for Reporter.	<HeartbeatRate>	Specifies the frequency, in seconds, of the heartbeat signal the Intelligence Server sends to the Management Server. Minimum = 15; default = 60
		<ActivityJournal SettingConfiguration>	Configures the database management operations related to the Activity Journal, which stores data on policy enforcement activity.
		<numberofExtendedAttrs>	Specifies the number of extra columns for dynamic attributes in the main reporting table. If the number of attributes exceeds the specified value, attributes are stored in a child table. default = 99
<Management Console>	Governs the operation of Administrator.	<HeartbeatRate>	Specifies the frequency, in seconds, of the heartbeat signal Administrator sends to the Management Server. Minimum = 15; default = 60
<Message Handlers>	Contains settings that control the ICENet Server's connection to your mail server, for email notification purposes.	<Server> <Port> <Username> <Password> <Default_From>	Specifies the name of the network host where the mail sever is running. The port where the ICENet connects to the mail server. The username required for this connection. If your SMTP server does not require authentication, this may be left blank. The password required for this connection. If your SMTP server does not require authentication, this may be left blank. Specifies the content of the From field of email notifications sent by this server, used in cases when no sender is specified in the policy. This value is supplied during Control Center installation.

Section	Purpose	Element or property	Function
		<Default_To>	Specifies the content of the To field of email notifications sent by this server, used in cases when no recipient is specified in the policy. This value is supplied during Control Center installation.
<Reporter>	Governs the operation of the Reporter.	<HeartbeatRate> use.past.data.for.monitoring	Specifies the frequency, in seconds, of the heartbeat signal Reporter sends to the Management Server. Minimum = 15; default = 60 Specifies whether monitors evaluate policy activity data that is logged prior to the creation of the monitors. Type a value of true or false.
		monitorExecutionInterval	Specifies the frequency, in minutes, at which a background job evaluates all active monitors. By default, the job evaluates active monitors every 2 minutes.

The following table provides examples of values for the DeploymentTime property described in the previous table.

Table 31: Examples of DeploymentTime settings

Deployment Time Example	Explanation
00:00	Next midnight (default value of <DeploymentTime>)
Wednesday 1:20	1:20 AM every Wednesday
first Monday 1:15	1:15 AM on the first Monday of each month
Last Saturday 22:30	10:30 PM on the last Saturday of each month
3rd FRIDAY 0:10	10 minutes past midnight on the third Friday of each month
fourth SATURDAY 12:30	12:30 PM on the fourth Saturday of each month
fifth Monday 23:59	11:59 PM on the fifth Monday of each month

If the required day of the current month has already passed, the next month is used. If the next month does not have the specified day (for example, the month of February does not have a fifth Sunday until 2032) the following month is selected.

Related concepts

[Configuration settings](#) on page 136

The main Control Center configuration file, configuration.xml, is copied to the host server when you first install the Control Center.

[Database management operations](#) on page 125

As the Report Server gathers data on user activity and policy enforcement, it temporarily stores it in OLTP tables before writing it to the Activity Journal database, where it is available for generating reports.

Related tasks

[Creating a monitor on page 237](#)

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

MessageHandler settings

During the Control Center installation, the install wizard prompts you for several parameters connected with the email notification feature.

These are stored in the `<MessageHandlers>` section of the configuration file. They include:

- The server name and port of your network mail server
- A username and password with which the ICENet Server can connect to the mail server
- Default From and To addresses. These are used in email notification messages whenever that obligation is included in a policy, but no sender or recipient is specified in the definition of the policy.

The following figure shows the `MessageHandlers` section. If you need to change any of these settings, you can manually edit the configuration file at any time.



Figure 43: Email message handler configuration

Encrypting passwords

Whenever you generate a new password for the configuration file, you should encrypt it.

1. Open a console window.
2. Change directory to `<InstallDir>/tools`.
3. Run the following command, where `<password>` is the new password you want to place in the configuration file:
`mkpassword.bat -w <password>`
4. An encrypted version of the password appears on the screen. The encrypted value is random; if you run the command again for the same password, a different encrypted version is generated.
5. Copy the encrypted password from the screen and paste it into `configuration.xml` at the desired location.

 **Note:** You can use the command `mkpassword.bat -h` to display a help screen with instructions for using this utility.

Related concepts

[About connection pools](#) on page 122

The four repository sections contain just a few settings—basically just the `<name>`, and a pointer to a connection pool section, where the actual connection information for that repository is stored.

[Configuring the User Repository](#) on page 137

The settings that capture information about the internal directory server hosting Control Center user information are organized in the `<UserRepositoryConfiguration>` section.

[Configuring the external domain authentication](#) on page 138

The `<ExternalDomainConfiguration>` section contains several properties that are used to configure information about the external domain for user and group information retrieval, and for remote authentication of users logging into Reporter and Administrator.

[Configuration file structure](#) on page 146

Related tasks

[Securely exporting and importing objects using the Control Center interface](#) on page 203

Configuring data access

The `configuration.xml` file also contains settings for connecting to the four data stores required by Control Center.

- Management Database
- Policy Master
- Activity Journal
- Information Network Directory

Depending on your configuration, all four may either use the internal database, or connect to an externally hosted Oracle, PostgreSQL, or SQL Server database. Each of these connections is controlled by a group of parameters, in one of the four `<repository>` sections of the file.

Configuring the ICENet log queue

As enforcers send activity data to the ICENet Server, it buffers that data in a queue and periodically writes logs to the Activity Journal. If your system's enforcement activity is so heavy that activity data is being uploaded from enforcers to the ICENet Server faster than that server can write logs to the Activity Journal database over an extended period, then the binary log data can start to accumulate in the file system queue. This queue is located at `<INSTALLDIR>\server\logqueue`.

The most immediate impact in such cases is a risk of running out of disk space, which can drastically reduce the performance of the ICENet Server host. Some secondary impacts can include the loss of activity log data uploaded from enforcers, and failure to log system events in the server logs.

The DABS section of the configuration file contains two configuration options, shown in the following table, that can help cope with such situations. They are located in a section called `<FileSystemLogConfiguration>`.

Table 32: ICENet queue management options

Element	Description
ThreadPoolMaximumSize	Specifies the maximum number of threads available for the ICENet Server to writing logs into persistence—in effect, clearing data from the queue. Default value: 8
QueueManagerUploadSize	Specifies the maximum size of logs the ICENet Server writes for each log entry it sends to the database. The larger this number, the more data can be cleared from the queue per each database write. Default value: 1MB

Under normal conditions, these default values should provide adequate performance. If you experience noticeable slowdowns in one or more ICENet Servers, you can increase these values one at a time. You should tune the system by increasing the Thread Pool Size moderately (say, to 12) and checking the performance results, then increasing the log upload size to 2M, and rechecking.

Related concepts

[Configuration settings](#) on page 136

The main Control Center configuration file, `configuration.xml`, is copied to the host server when you first install the Control Center.

Tomcat settings

Control Center's UI applications work through the Tomcat web-based application server, included in every Control Center installation.

During installation, all required configuration settings are supplied by prompts in the install wizard, and all optional ones are assigned with default values. For example, the file contains `<Service>` sections that provide settings for two services, CE-Core and CE-Apps. You must not delete either of these sections, but you can modify some of their properties. This section provides details about all required and optional configuration settings.

If you install Control Center components in a distributed architecture rather than on a single host, Tomcat is installed on each of the host machines.

Related concepts

[Configuration tools](#) on page 134

This section describes the tools available for modifying the configuration of Control Center after it is installed.

[About configuration files](#) on page 134

The Control Center configuration files are XML files that contain configuration settings you can view or modify to tune your system's performance, scale the system by adding or removing modules, and address various administrative issues.

Location

The settings described in this section can be changed by opening and editing the Tomcat configuration file.

During installation, this file is placed in the target host, at the following location:

`<InstallDir>\Nextlabs\PolicyServer\server\configuration\server.xml`

If you distribute the components of Control Center, this host is the one where the Policy Management Server is installed.

Configuration file structure

The Tomcat configuration file contains two Service sections with the following names:

- CE-Core contains settings for the core server framework, which relays information across machines between components, relays license key enforcement, and so on.
- CE-Apps contains settings for Control Center's web applications (Administrator and Reporter).

You can find these sections in the file by searching for the tags <Service name="CE-Core"> and <Service name="CE-Apps">. Both contain the following subsections, containing additional settings:

- <Connector>
- <Host configuration>
- <Default Context for Components>
- <Component Context Settings>

The following table describes the function and use of the settings available in each of these sections.

Table 33: Connector section properties

Setting	Description
port	<p>To change the port number later, you must propagate the change consistently to any other settings that refer to the same port. Be sure to update the port property in every Tomcat configuration file throughout your organization. You must also update the value wherever it is used in other properties throughout the file (for example, DACLocation) so that other components can start to communicate using the new port. It is recommended that you do this by examining each property one by one, rather than by using a global “search and replace” strategy. Remember that you must restart Tomcat for the new settings to take effect.</p> <p>In addition, if you are using load balancing and you change the application port number, you need to inform the load balancer of the new location.</p> <p>If you do not properly reconfigure all components, the fact that one or more is unable to connect should be apparent from a message in the log file. Similarly, if you choose a port number that is already in use by a different application, and a port conflict arises, this should be reflected in the log file.</p>
keystoreFile	<p>The path to the keystore file where the security certificate is stored. In the DCC-Apps section, keystoreFile, keystorePass, and keystoreType are useful when you want to exchange the supplied Web application security certificates for certificates of your own. It is not recommended that you change this setting within the DCC-Core section, which configures the security certificates for communication among components of Control Center. For information about security certificates and how to change them, see the SSL certificates section.</p>
keystorePass	<p>The password used to access the server certificate from the specified keystore file. This password is encrypted during installation. If you later change the password, you must run the encryption utility to re-encrypt the password.</p>
keystoreType	<p>The type of keystore file to be used for the server certificate. The value should always remain set to “JKS” for Java Key Store.</p>
truststoreFile	<p>A list of trusted entities. Do not change this setting; however, you might need to look at the value to find out where the truststore file is located.</p>

Setting	Description
truststorePass	The password used to access the server certificate from the specified truststore file. This password is encrypted during installation. If you later change the password, you must run the encryption utility to re-encrypt the password.
truststoreType	The type of truststore file to be used for the server certificate. The value should always remain set to “JKS”.
clientAuth	To keep your server secure, the connector for the web service port should always have this attribute set to true.
maxThreads	When true, the clients of this service that make an application request must be authenticated (by verification of a valid certificate chain) before a connection is made. This ensures that only trusted clients ever make requests to Control Center’s internal applications.
minSpareThreads	The maximum number of request processing threads to be created, which determines the maximum number of simultaneous requests that can be handled.
maxSpareThreads	The number of request processing threads created when this connector is first started. The connector ensures that it has the specified number of idle processing threads available.
enableLookups	The maximum number of unused request processing threads that can exist at any given time. If this number is exceeded, the thread pool stops the unnecessary threads.
acceptCount	Performs DNS lookups and return the actual hostname of the remote client if the value is set to true. By default, for performance improvement, this value is set to false.
connectionTimeout	The maximum queue length for incoming connection requests when all possible request processing threads are in use. Any requests received when the queue is full are refused.
scheme	The number of milliseconds this connector waits, after accepting a connection, for the request URI line to be presented.
secure	The name of the protocol to be used. To keep the server secure at all times, the value should always be “https” and should not be changed.
Host configuration	A flag that indicates whether the connection is secure or unsecure.
Name	The <Host> section within each <Service> section contains the following properties to configure the host of the service:
autoDeploy	The virtual hostname. This value should remain set to “localhost” at all times.
unpackWARs	Determines whether new Web applications added to the appBase directory while Tomcat is running should be automatically deployed. Since no new Web applications are expected while Tomcat is running, this value can remain set to false.
	Unpacks web application archive (WAR) files in the appBase directory into a corresponding disk directory structure when this property is set to true. Runs web applications directly from a WAR file when this property is set to false. Default: false.

Setting	Description
xmlValidation	Enables validation of XML configuration files when set to true. Default: false.
appBase Default Context for Components	The <DefaultContext> section configures server resources shared by all Control Center components. You should never modify any of these settings.

The configuration file contains one <Context> section for each Control Center component that is installed on this host; normally there is only one. The file may contain other <Context> sections that are commented out. You can add more Control Center components later simply by uncommenting the appropriate <Context> sections.

 **Note:** Do not comment out the <Context> section for DCSF. Every machine must have this component. The following table describes the Component Context section.

Table 34: The component context section

Setting	Description
path	<p>The context path of this Web application, which is matched against the beginning of each request URI to select the appropriate Web application for processing. For example, when path=“/administrator”, this <Context> section contains settings for the Administrator application.</p> <p>Each context path value must be unique. In the DCC-Core service, the context path values should not be changed, as Control Center expects these values to be present. In the DCC-Apps service, you can change the Web application context paths to different values; for example, “/console” instead of “/administrator”.</p> <p>When path=“/reporter”, this <Context> section contains settings for the Reporter application. This <Context> section has no ComponentName property, because the Reporter application is not always part of the Control Center installation; it could be replaced by a custom application built using the Control Center API. Therefore, Reporter can not necessarily be monitored using the Administrator application, and therefore does not need a display name for use on screen in that application.</p>
reloadable	Monitors classes in /WEB-INF/classes/ and /WEB-INF/lib for changes and automatically reloads the web application if a change is detected when set to true. To optimize the server performance, this value should be set to false at all times.
docBase	Absolute path name to the application Web Archive (WAR) file. This value is set at installation based on the Control Center installation location. This value should not be changed.
workDir	Path name to a scratch directory to be provided by this context for temporary read-write use by servlets within the associated Web application. This value is set at installation based on the Control Center installation location. This value should not be changed.

Setting	Description
ComponentName Parameter	The value, in the format machine_component, is the component name displayed in Administrator. A default name is provided during installation. You can change this value, but it must be unique over all Tomcat configuration files. That is, no two Control Center component instances can have the same component name.
Location Parameter	The location where the component can be reached. This value is set at installation and should not be modified.
DACLocation Parameter	Used only where path="/reporter". Specifies the location of the Intelligence Server (DAC) as part of the settings for the Reporter application. If you have installed multiple instances of the Intelligence Server and are using a load balancer to distribute Reporter requests among them, change the value of DACLocation to the location of the load balancer.

Related concepts

[SSL certificates](#) on page 354

All internal communication among various software components of the Control Center platform is conducted using SSL encryption.

Related tasks

[Encrypting passwords](#) on page 144

Whenever you generate a new password for the configuration file, you should encrypt it.

Policy Controller configuration

Policy Controller configuration includes the following tasks.

Related concepts

[Configuration tools](#) on page 134

This section describes the tools available for modifying the configuration of Control Center after it is installed.

[Event log settings](#) on page 150

Control Center has an extensive event logging mechanism that can capture all events relevant to policy enforcement.

[Thread pool size settings](#) on page 152

You can tune the performance of the Policy Controller by optimizing the number of threads.

[How to set up the REST API](#) on page 286

To use the REST API, the Java Policy Controller must be installed and configured. For more information, see the NextLabs Control Center Installation and Upgrade Guide.

Related tasks

[Memory settings](#) on page 152

If your system includes custom plug-ins for Policy Controllers, you might need to increase the JVM memory.

[ICENet server settings](#) on page 153

In the event that the location of the ICENet server is misconfigured or requires reconfiguration, you can reset it.

[Setting up user authentication for the REST API](#) on page 154

To use the REST API with Java Policy Controllers, you need to set up user authentication.

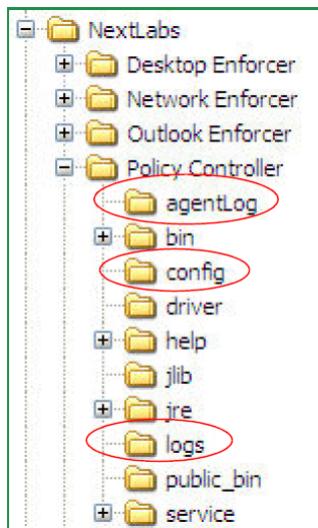
Event log settings

Control Center has an extensive event logging mechanism that can capture all events relevant to policy enforcement.

Logging is performed on the enforcer side, not at the Control Center. Each log is associated with one installation of a policy enforcer, and all events are captured in a log file. By default, these files are maintained on each enforcer host, at the following location:

\NextLabs\Policy Controller\agentLog

Logs are maintained on each local enforcer host, but are sent back to the Control Center periodically so their event content can be added to the Activity Journal database.



Since you use Reporter to generate reports on system events and the performance of enforcers, you usually do not need to use these files directly. One exception might be during troubleshooting, in which case you should follow instructions from NextLabs technical support representatives.

Related concepts

[Policy Controller configuration](#) on page 150

Enforcer logging

If you do not change any log settings for enforcers, the default behavior is as follows:

- Level of verbosity = Severe
- max size of file = 500K
- To change any of this behavior at any time, edit the file `logging.properties`, in the `config` directory of the Policy Controller's install directory. The available levels, in order of increasing verbosity, are:
 - Severe
 - Warning
 - Info
 - Fine
 - Finest

To edit the files, you must first stop the enforcer. Configuration files cannot be opened when an enforcer is running.

Related concepts

[Stopping policy enforcers](#) on page 111

There are occasions when you might want to stop the execution of the policy enforcer on a particular machine without uninstalling it.

The enrollment log

Control Center maintains a separate log for events connected to enrolling data into the system dictionary.

It is called `Enrollment.N.log`, where N is an integer from 0-9. It is stored in the same location as the event log.

Memory settings

If your system includes custom plug-ins for Policy Controllers, you might need to increase the JVM memory.

If you get Out-of-Memory errors, this indicates that you need to increase the memory. For Policy Controller for Java, set the JVM memory options at the Tomcat or JBoss level. For Endpoint and Server Policy Controllers, which run on Windows systems, you set the memory option in the Windows Registry.

1. Stop the Policy Controller. From the Start menu, select Stop Policy Controller.
2. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type `regedit`, and press **Enter**.
3. In the Registry Editor, navigate to the following folder:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services  
  \ComplianceEnforcerService\Parameters
```

4. Open `AppParameters` and look for the following setting. Increase the value as needed.
`-Xmx1024m`

Related concepts

[Policy Controller configuration](#) on page 150

Thread pool size settings

You can tune the performance of the Policy Controller by optimizing the number of threads.

If your performance testing indicates that the number of threads should be changed, use one of the procedures in this section to set the number of threads.

Related concepts

[Policy Controller configuration](#) on page 150

Changing the number of threads through the Windows Registry

You can perform this task either through the Windows Registry or the command line.

The following procedure explains how to make the change in the Windows Registry.

1. Stop the Apache Tomcat service. To do so, select **Start > Control Panel > Administrative Tools > Services**. Locate the Tomcat service and select **Stop**.
2. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type `regedit`, and press **Enter**.
3. In the Registry Editor, navigate to the following folder:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Apache Software Foundation\Procrun  
  2.0\Tomcat8\Parameters\Java
```

4. Right-click **Options** then select **Modify**.
5. Insert the following command, or modify the command if it already exists:
`-Dnextlabs.evaluation.threadpoolsize=<number of threads>`

Where: `<number of threads>` is the number of threads to configure.

Changing the number of threads through the command line

1. Stop the Apache Tomcat service. Select **Start > Control Panel > Administrative Tools > Services**. Locate the Tomcat service and select **Stop**.

2. Right-click Apache Tomcat X.X and click **properties**. On the **General** tab, take note of the service name.
3. Open a Command prompt, and navigate to Tomcat's installation directory.
4. Run the following command:

```
tomcat8w //ES//<service name>
```

Where: <service name> is the name of the Tomcat service you noted in step 2.

5. In the Apache Tomcat <service name> Properties window, select the **Java** tab.
6. In Java Options, insert the following command, or modify the command if it already exists:
`-Dnextlabs.evaluation.threadpoolsize=<number of threads>`
<number of threads> is the number of threads to configure.
7. Click **OK**.

Changing the number of threads for server Policy Controllers

You perform this task through the Windows Registry.

1. Stop the Policy Controller. From the Start menu, select **Stop** Policy Controller.
2. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type `regedit`, and press **Enter**.
3. In the Registry Editor, navigate to the following folder:
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\ComplianceEnforcerService\Parameters`
4. Open AppParameters, and insert the following command, or modify the command if it already exists:
`-Dnextlabs.evaluation.threadpoolsize=<number of threads>`
Where: <number of threads> is the number of threads to configure.

ICENet server settings

In the event that the location of the ICENet server is misconfigured or requires reconfiguration, you can reset it.

1. Stop the Policy Controller on the host where the reconfiguration is required. This requires the Administrator password.
2. Navigate to `\NextLabs\Policy Controller\` and delete the `bundle.bin` file.
3. Locate the Policy Controller configuration folder at: `\NextLabs\Policy Controller\config`
4. Locate and delete the file `registration.info`.
5. Locate and open the subfolder `\security`. Delete the following files:
 - `agent-keystore.jks`
 - `agent-secret-keystore.jceks`
 - `agent-truststore.jks`

 **Note:** Do not delete `\config\security\temp_agent-keystore.jks`. Doing so causes errors for system startup.
6. In the `\config` folder, locate and edit `commprofile.xml`, and update the location of the DABS location. Be sure to supply the value of “`full path:port/dabs`” as shown in the following figure.

```

<?xml version="1.0" encoding="UTF-8"?>
<comm-profile-dTO push-enabled="true" default-profile="false" id="4">
<created-date xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
time="2008-09-15T17:35:10.360-07:00" xsi:type="gregorian-calendar"/>
<modified-date xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
time="2011-01-18T15:26:51.001-08:00" xsi:type="gregorian-calendar"/>
<custom-activity-journaling-settings><name>DESKTOP:Policy Testing</name>
<logged-activities><action>
CHANGE_ATTRIBUTES,CHANGE_SECURITY,COPY,DELETE,MOVE,PRINT,OPEN,EDIT,RENAM
E,EMAIL,RUN,IM</action></logged-activities></custom-activity-journaling-
settings><default-push-port value="2000"/><log-limit value="2"/>
<password-hash>ZF92ai19g0yU80LTr9/im8mo+Og=</password-hash><DABSLocation
value="https://NXT-CE10.nextlabs.com:8443/dabs"/><current-activity-
journaling-settings><name>Default</name><logged-activities><action>
CHANGE_ATTRIBUTES,CHANGE_SECURITY,COPY,MOVE,PRINT,RENAME,EMAIL,RUN,IM
</action></logged-activities></current-activity-journaling-settings>
<log-frequency><time-unit>seconds</time-unit><time value="30"/></log-
frequency><heart-beat-frequency><time-unit>minutes</time-unit><time
value="15"/></heart-beat-frequency><name value="ALPO Debug Profile"/>
</comm-profile-dTO>

```

Figure 44: Updating the location of the ICENet server

7. Close and save the `commprofile.xml` file.
8. Start the Policy Controller.

Related concepts

[Policy Controller configuration](#) on page 150

Setting up user authentication for the REST API

To use the REST API with Java Policy Controllers, you need to set up user authentication.

1. For policy controllers installed on servers running Tomcat, disable the OAuth filter:
 - a. On the Tomcat server, go to the `conf` folder and open `server.xml` in a text editor.
 - b. Find the XML element `Context` with attribute `path` value `/dpc`.
 - c. Change the parameter `EnableJWTAuthenticationFilter` value to `false`.
2. For policy controllers installed on servers running JBoss, change the parameter `EnableJWTAuthenticationFilter` value to `false` in the `dpc.properties` file.
3. Specify the following parameters:
 - For Java PEP SDK specify the following parameters in the `openaz-pep.properties` file:

```

#-----
#
#                               NEXTLABS OpenAZ PEP Properties
#
#-----
#
# PDP Engine configurations
#      - Embedded PDP:      com.nextlabs.openaz.pdp.EmbeddedPDP Engine
#      - REST/ CloudAZ PDP: com.nextlabs.openaz.pdp.RestPDP Engine
#-----
# PDP Engine class, when using embedded PDP, set to
# "com.nextlabs.openaz.pdp.EmbeddedPDP Engine"
nextlabs.pdp.engine.name=com.nextlabs.openaz.pdp.EmbeddedPDP Engine

#
# When using Embedded PDP configure the properties below

```

```

# path to dpc folder: Embedded PDP requires dpc folder path which
# includes all
# the resources required for embedded pdp
#-----
# E.g. For Windows Environment: nextlabs.dpc.root=C:/nextlabs_embedded_pdp/dpc
#                               or nextlabs.dpc.root=C:\nextlabs_embedded_pdp\\\dpc
# For Linux Environment:      nextlabs.dpc.root=/home/nextlabs_embedded_pdp/dpc
# nextlabs.dpc.root=<path_to_embeddedPDP_dpc_folder>

# Payload type. json or xml
#-----
#nextlabs.cloudaz.payload_content_type=application/xml
nextlabs.cloudaz.payload_content_type=application/json

#
# When using REST/ CloudAz PDP configure the properties below

# The host of CloudAz server eg: saas-jpc.cloudaz.com
#-----
nextlabs.cloudaz.host=<CloudAz REST API host>

# The port on which the CloudAz service is listening on the server
#-----
nextlabs.cloudaz.port=443

# Whether the CloudAz service is over https (true or false)
#-----
nextlabs.cloudaz.https=true

# The authentication settings to connect with the REST/CloudAz service
# Two authentication type are available to use
#   - No authentication required: NONE
#   - Use with OAuth2 provided authentication service: OAUTH2
#-----
nextlabs.cloudaz.auth_type=OAUTH2

#
# OAuth2 Related configurations
#   only if nextlabs.cloudaz.auth_type is OAUTH2

# The OAuth2 Authorization Grant Type
#   available grant type is
#     - client_credentials (default)
#-----
nextlabs.cloudaz.oauth2.grant_type=client_credentials

# The OAuth2 server configurations
#   Default oauth2 service is provided by control center server, in that
#   case
#   configure control center server host and port
#-----
nextlabs.cloudaz.oauth2.server=<control center host>
nextlabs.cloudaz.oauth2.port=<control center port>
nextlabs.cloudaz.oauth2.https=true

# Client Id to identify the client connecting using OAuth2
#-----
nextlabs.cloudaz.oauth2.client_id=<CLIENT_ID>

# Client secret for OAuth2 client credentials grant

```

```

#-----
nextlabs.cloudaz.oauth2.client_secret=<CLIENT_SECRET>

# Oauth endpoint to get the token for the client credential grant
#   - CloudAZ endpoint: /oauth/token
#   - REST endpoint : /cas/token
#-----
nextlabs.cloudaz.oauth2.token_endpoint_path=/cas/token

# Ignore HTTPS self signed certificates error, if using self signed
# certificates
#-----
nextlabs.cloudaz.ignore_https_certificate=true

# OpenAZ api configuration
# no need to change this unless required
#-----
xacml.pdpEngineFactory=com.nextlabs.openaz.pdp.PDP EngineFactoryImpl

# Mapper classes used internally to map requests
#-----
pep.mapper.classes=com.nextlabs.openaz.pepapi.RecipientMapper,com
.nextlabs.openaz.pepapi.DiscretionaryPoliciesMapper,com.
nextlabs.openaz.pepapi.HostMapper,com.
nextlabs.openaz.pepapi.ApplicationMapper

```

- For the JavaScript PEP SDK, specify the following parameters in the `openaz-pep.json` file:

```
{
  "nextlabs.cloudaz.port": "58080",
  "nextlabs.cloudaz.https": false,
  "nextlabs.cloudaz.auth_type": "OAUTH2",
  "nextlabs.cloudaz.oauth2.grant_type": "client_credentials",
  "nextlabs.cloudaz.oauth2.client_id": "<CLIENT_ID>",
  "nextlabs.cloudaz.oauth2.client_secret": "<CLIENT_SECRET>",
  "nextlabs.cloudaz.ignore_https_certificate": true,
  "nextlabs.cloudaz.oauth2.server": "<Control Center Host>",
  "nextlabs.cloudaz.oauth2.port": "443",
  "nextlabs.cloudaz.oauth2.https": true,
  "nextlabs.cloudaz.oauth2.token_endpoint_path": "/cas/token"
}
```

- In the Control Center console, create a user account for the PEP client with the following settings.
 - Username = The `client_id` specified in the properties file
 - Password = The `client_secret` specified in the properties file
 - Add the user attribute `jwt_passphrase`. The value of this attribute can be anything except for the password (`client_secret`).
- After the account has been created, log in to the account and supply the appropriate `client_secret` password. This is the password specified in the properties file.
- For security, ensure that all users change their passwords when they log in to their accounts for the first time.

Related concepts

[Policy Controller configuration](#) on page 150

[About managing user accounts](#) on page 26

Control Center user accounts enable users to log in to the Control Center web console using a specified username and password.

Related tasks

[Setting up the Java SDK](#) on page 266

If you are developing Java PEPs, add the required JAR libraries to your Java project to set up the Java SDK.

[Setting up the JavaScript SDK](#) on page 280

To use the JavaScript client library, set up the JavaScript SDK.

Chapter

8

Policy and Policy Model overview

Topics:

- [About policies](#)
- [Overview of policy implementation](#)
- [Managing Policy Model resource types](#)
- [Managing policy components](#)

This section provides an overview of policies and explains how to create and manage Policy Model resource types and policy components.

Related concepts

[Configure Control Center on page 11](#)

To use NextLabs Control Center to enforce policies, the system must be configured for your environment, and policies must be designed and deployed.

[About policies on page 159](#)

Policies are statements that describe resource usage situations and specify the actions to be taken when those situations arise.

[Overview of policy implementation on page 159](#)

Implementing policies involves several tasks.

[Managing Policy Model resource types on page 160](#)

Policy Model resource types are the templates that include information about attributes, actions, and obligations.

[Managing policy components on page 167](#)

Policy components are the abstract building blocks or raw material of information control policies.

Related tasks

[Adding policies on page 175](#)

About policies

Policies are statements that describe resource usage situations and specify the actions to be taken when those situations arise.

In other words, policies define a set of rules controlling how categories of users, hosts, or applications in a given environment are allowed to use categories of resources. Policy designers construct policies as combinations of components that are linked with operators and other logical constraints, and then further refined by contextual conditions, such as time of day.

Typically, organizations construct all the policies required to cover all the potential business situations where information must be controlled, or where an event, such as displaying a reminder message, should be triggered in response to a user action. Each policy comprises a set of predefined building blocks that are combined according to the syntax shown in the following figure.

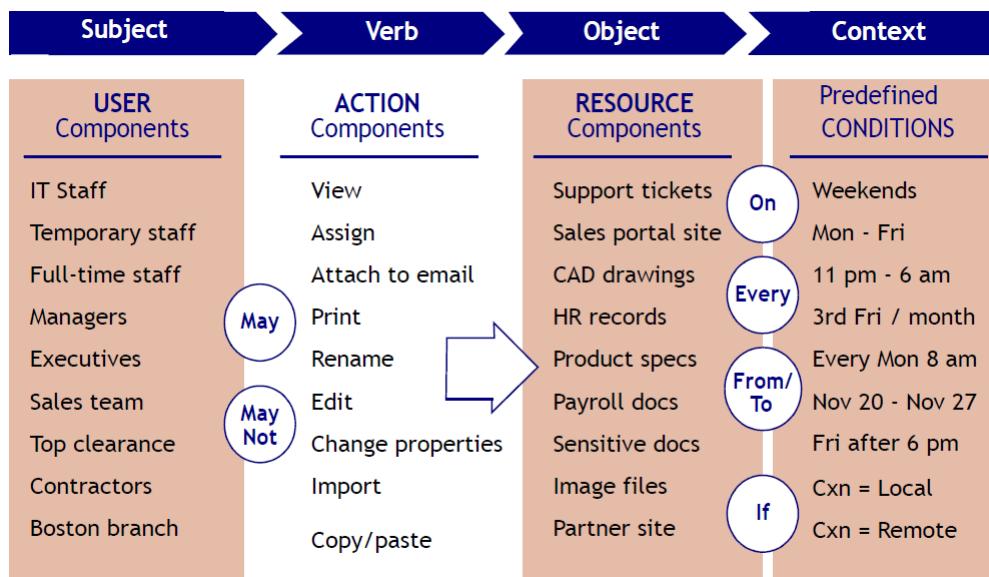


Figure 45: Policy components and syntax

Each enforcement event is caused by a single user performing a specific action on a particular resource that is covered by a policy. For example, consider a Human Resources assistant attempting to open a Microsoft Word document that contains an employment offer letter. If there is a policy that allows only the HR director access to this type of document, access is blocked, and a message appears on the assistant's device explaining that he is not authorized to open the file, and, if the policy requires it, sending a notification email to the HR director.

- ! **Important:** Policies are enforced only after policy enforcement points have been configured and policies have been deployed. For more information about configuring policy enforcement points, see the user guide for each NextLabs enforcer.

Related concepts

[Policy and Policy Model overview](#) on page 158

Overview of policy implementation

Implementing policies involves several tasks.

- **Creating a Policy Model:** The Policy model contains three pre-defined and non-removable subjects.
 - User (An actor whose attributes may be referenced in a component)
 - Host (A computer whose attributes may be referenced in a component)

- Application (An application program whose attributes may be referenced in a component)

These records may contain pre-seeded attributes from enrollment as well. Policy authors can add custom attributes to these policy models and use in component creation.

The Policy Model determines the kind of subject types (users, hosts, and applications) and resource types (documents and other resources) that can be covered by policies. To create a Policy Model, you create resource types, and specify the attributes, actions, and obligations available to each resource type during policy creation.

- Defining policy components: Policy components represent categories or classes of entities, such as users, and actions, such as open or copy. These components can be thought of as the parts of speech used to construct policy statements. For example, “noun” policy components might include All employees in the human resources department or Any file with an XLS extension, and “verb” components might include Copy, Print, and Rename File.
- Using components to construct policies: Policies use components as building blocks to represent rules that control information access and use. For example, the components in the following policy, “Allow Users in IT to Edit and Reassign Tickets in their Product Area,” are “IT Department,” “Edit,” “Assign,” and “Tickets in User’s Product Area.” Policies are stored in a Policy Master database for distribution to the network components that enforce them. Conditions are policy elements that change a policy’s effect based on dynamic comparisons, evaluations, or contextual factors. Complex conditions can be written directly in ACPL (Active Control Policy Language), the language Control Center uses internally to represent, store, and manage components and policies. Policies can also include obligations, such as displaying notifications to end users and sending email messages to administrators when policies are enforced.
- Deploying policies to policy enforcement points (PEPs): Deploying policies means distributing policies to the appropriate PEPs on devices such as desktop computers, laptops, and file servers throughout the organization. Policies are enforced only after they are deployed and the appropriate PEPs are configured.
- Fine-tuning policies after they have been deployed: After policies have been deployed, they can be monitored and updated as needed to ensure that they are performing as required.

Related concepts

[Policy and Policy Model overview](#) on page 158

[Managing Policy Model resource types](#) on page 160

Policy Model resource types are the templates that include information about attributes, actions, and obligations.

[Constructing policies](#) on page 175

Policy designers construct policies to control information access and use in an organization.

[Deploying and managing objects](#) on page 194

[Managing policies and components](#) on page 195

Managing Policy Model resource types

Policy Model resource types are the templates that include information about attributes, actions, and obligations.

These items are available to policy designers when they create components and policies. Administrators should add resource types before components are created.

Related concepts

[Policy and Policy Model overview](#) on page 158

[Overview of policy implementation](#) on page 159

Implementing policies involves several tasks.

Adding and editing subject types

You can add or edit subject types at any time. To make attributes, actions, and obligations available to policy designers, resource types should be added before components are created.

1. Log in to the console with an account that has permission to add resource types.
2. On the left navigation bar, click **Policy Model**.

The Policy Models page appears displaying three pre-defined and non-removable subjects (User, Host, Application).

The screenshot shows the 'Policy Models' page in the NextLabs Control Center. The left sidebar has a 'Policy Model' section selected. The main area displays four subject types: 'User', 'Host', 'Application', and 'FSO'. Each subject type has a checkbox, a delete icon, and a 'Modified' timestamp. A 'SORT BY' dropdown is visible on the right. The subjects are categorized as 'SUBJECT' (User, Host, Application) and 'RESOURCE' (FSO). The 'User' subject is described as 'An actor whose attributes may be referenced in a component'.

Figure 46: The Policy Models page

3. Click the name of a subject type you want to edit.
4. In the SUBJECT TYPE INFORMATION section, you can edit the subject type information. This information is displayed on the Policy Models page and identifies the subject type in the system:

Table 35: Resource type information

Option	Description
Name	The display name of the subject type. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Short name	The internal identifier associated with the subject. You cannot edit this field.
Description	A description of the item. This field is useful for explaining how the item is used and for documenting changes to the item. Descriptions can include multibyte characters, such as those used by Asian languages.

Option	Description
Tags	<p>Tags are labels that identify or classify objects. For example, you could apply a tag named Team1 to all documents related to the Team1 project. You could then search for documents with the Team1tag, or set access requirements for documents with the Team1 tag.</p> <p> Note: Tags used in components are available only to components; component tags are not available to policies, and policy tags are not available to components.</p>

- Click the **ATTRIBUTES** tab, then add the attributes that you want to make available to policy designers who select the resource type:

Table 36: Attribute information

Option	Description
Attribute Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Short Name	The internal identifier associated with the item. The system uses short names to reference items in the database. Short names cannot contain spaces or special characters, and they cannot be changed after they are added. In addition, short names must be unique in the system. Developers should keep a record of the short names because they are required for API calls.
Data Type	The kind of information in the attribute. Data types include strings or text, numbers, and collections. Collections are sets of data, such as locations or vendors.

Option	Description
Operators	<p>The symbol selected for the operation. The available operators are determined by the data type selected.</p> <p>MULTIVAL</p> <ul style="list-style-type: none"> not equal (!=): The item does not match the attribute multival. equal (=): The item matches the attribute multival. exactly matches: The item matches all of the items in the attribute multival. includes: The item contains one or more items in the attribute collection. <p>NUMBER</p> <ul style="list-style-type: none"> equals (=): The item matches the attribute number. not_equals (!=): The item does not match the attribute number. less than (<): The item is less than the attribute number. less than equals (<=): The item is less than or equal to the attribute number. greater than (>): The item is greater than the attribute number. greater than equals (>=): The item is greater than or equal to the attribute number. <p>STRING</p> <ul style="list-style-type: none"> is: The item matches the attribute string. is not: The item does not match the attribute string.

- Click the **PRE-SEEDED ATTRIBUTES** tab to view the pre-seeded attributes that are added when users are enrolled from an LDAP server or HR system.
- Click **SAVE** at the upper right of the page. The subject type is updated to the Policy Model. It is available for selection during policy component creation.

Adding and editing resource types

You can add or edit resource types at any time. To make attributes, actions, and obligations available to policy designers, resource types should be added before components are created.

- Log in to the console with an account that has permission to add resource types.
- On the left navigation bar, click **Policy Model**.
- Do one of the following:
 - Click **ADD RESOURCE TYPE**.
 - Click the name of an existing resource type to edit it.
- Provide Resource Type Information. This information is displayed on the Policy Models page and identifies the resource type in the system:

Table 37: Resource type information

Option	Description
Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.

Option	Description
Short name	The internal identifier associated with the item. The system uses short names to reference items in the database. Short names cannot contain spaces or special characters, and they cannot be changed after they are added. In addition, short names must be unique in the system. Developers should keep a record of the short names because they are required for API calls.
Description	A description of the item. This field is useful for explaining how the item is used and for documenting changes to the item. Descriptions can include multibyte characters, such as those used by Asian languages.
Tags	<p>Tags are labels that identify or classify objects. For example, you could apply a tag named Team1 to all documents related to the Team1 project. You could then search for documents with the Team1tag, or set access requirements for documents with the Team1 tag.</p> <p> Note: Tags used in components are available only to components; component tags are not available to policies, and policy tags are not available to components.</p>

- Click the **ATTRIBUTES** tab, then add the attributes that you want to make available to policy designers who select the resource type:

Table 38: Attribute information

Option	Description
Attribute Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Short Name	The internal identifier associated with the item. The system uses short names to reference items in the database. Short names cannot contain spaces or special characters, and they cannot be changed after they are added. In addition, short names must be unique in the system. Developers should keep a record of the short names because they are required for API calls.
Data Type	The kind of information in the attribute. Data types include strings or text, numbers, and collections. Collections are sets of data, such as locations or vendors.

Option	Description
Operators	<p>The symbol selected for the operation. The available operators are determined by the data type selected.</p> <p>MULTIVAL</p> <ul style="list-style-type: none"> not equal (!=): The item does not match the attribute multival. equal (=): The item matches the attribute multival. exactly matches: The item matches all of the items in the attribute multival. includes: The item contains one or more items in the attribute collection. <p>NUMBER</p> <ul style="list-style-type: none"> equals (=): The item matches the attribute number. not_equals (!=): The item does not match the attribute number. less than (<): The item is less than the attribute number. less than equals (<=): The item is less than or equal to the attribute number. greater than (>): The item is greater than the attribute number. greater than equals (>=): The item is greater than or equal to the attribute number. <p>STRING</p> <ul style="list-style-type: none"> is: The item matches the attribute string. is not: The item does not match the attribute string.

6. Click the **Actions** tab, then add the actions that you want to make available to policy designers who select the resource type. Actions are the operations that can be performed on the resource type:

Table 39: Resource type actions

Option	Description
Action Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Short Name	The internal identifier associated with the item. The system uses short names to reference items in the database. Short names cannot contain spaces or special characters except the colon (:) character, and they cannot be changed after they are added. In addition, short names must be unique in the system. Developers should keep a record of the short names because they are required for API calls.

 **Note:** The User subject type includes pre-seeded attributes that are added when users are enrolled from an LDAP server or HR system. Actions are not available for the User subject type.

If you create an action from within a resource type, the system automatically creates a corresponding action component. The new action component has these properties:

- The name of the action component is the name given to the action.

- The description of the action component indicates that it was generated automatically by the system.
 - Tags currently assigned to the resource type are automatically assigned to the action component.
 - If you delete the action from within the resource type deletes the action component provided that the component is not being used elsewhere. Components can be deleted only if they are not in use.
7. Click the **Obligations** tab, then add the obligations that you want to make available to the resource type:

 **Note:** Obligations are additional instructions the PDP might return to the PEP together with the authorization decision. For example, an obligation might instruct the PEP to notify the user why they are not allowed to perform a certain operation, and the message itself might be included in the obligation. In other cases, obligations might instruct the PEP to record the allowed user action in an audit database; encrypt a value before allowing the user to store that value; or inject a security filter in a SQL query to restrict the rows being returned, before allowing the user to run the query against a given database table.

Table 40: Resource type obligations

Option	Description
Obligation Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Short Name	The internal identifier associated with the item. The system uses short names to reference items in the database. Short names cannot contain spaces or special characters, and they cannot be changed after they are added. In addition, short names must be unique in the system. Developers should keep a record of the short names because they are required for API calls.

Option	Description
Parameters	<p>Information used to prepopulate fields in obligations. For example, if the obligation is to send email, an obligation parameter could include an email address. This simplifies the policy authoring experience by prepopulating values.</p> <p>Parameters information includes:</p> <ul style="list-style-type: none"> • Name: The name of the parameter. • Short Name: The identifier used to reference the parameter in the database. Underscores (_) are allowed, but short names cannot contain spaces or special characters, cannot be changed, and must be unique in the system. Developers should make a record of short names because short names are required for API calls. For example, to reference attributes in a request, you would use this format: resource.pat_rec.age, where pat_rec is the short name for Patient Record, and age is the short name for Age. • Type: The category of the parameter. Categories have differing properties: - Single Row: A single value. - Multiple Row: Items that appear in multiple rows. - List: A range of values. • List value: For List type parameters, the values to be displayed in the list. Separate each value with a comma. • Default Value: The value that appears by default. The default value might be an instruction, such as <Enter the reason why access is denied> to be replaced during policy component creation. Or the default value might include variables and expressions. • Hidden: Whether or not the parameter is displayed during policy component creation. • Editable: Whether or not the parameter can be edited during policy component creation. • Mandatory: Whether or not the parameter must be specified during policy creation.

- Click **SAVE** at the upper right of the page. The resource type is added to the Policy Model. It is available for selection during policy component creation.

Create additional resource types, or add policy components.

Related concepts

[Using Advanced Conditions](#) on page 181

Advanced Conditions are ACPL (Active Control Policy Language) expressions that can be used to include parameters that cannot be conveyed by policy components, such as user and resource components.

[Managing policy components](#) on page 167

Policy components are the abstract building blocks or raw material of information control policies.

Related tasks

[Adding and editing policy components](#) on page 169

You can add policy components to define subjects, resources, and actions in your environment any time as needed.

Managing policy components

Policy components are the abstract building blocks or raw material of information control policies.

They represent categories and classes, either of physical entities that play a role in information control policies, or of actions that involve those entities. Because components are categories and classes rather than physical entities, they provide a layer of abstraction that insulates an organization's information control policies from changes to the physical entities in the environment.

For example, employees might join or leave the organization, documents might be created and deleted, computers are purchased and retired, and SharePoint sites or pages change on a daily basis. Despite these changes, you can write policies without needing to know specifically about the underlying users, documents, or servers. The policies remain in effect, covering all appropriate network entities, even as those entities change.

Related concepts

[Policy and Policy Model overview](#) on page 158

[Deploying and managing objects](#) on page 194

Related tasks

[Adding and editing resource types](#) on page 163

You can add or edit resource types at any time. To make attributes, actions, and obligations available to policy designers, resource types should be added before components are created.

[Cloning components](#) on page 186

Duplicating components is referred to as component cloning. All types of components can be cloned.

About component types

You create components by specifying a component type and defining the component's properties.

The following categories of components are supported:

- Subject
- Resource
- Action

After you save and deploy components, you can use them to construct policies that govern not some specific user, host, or application in your organization, but any user, host, or application belonging to the category that the component describes.

For example, rather than listing every manager in your organization by name, you could create a user-type component that represents the category, "all users with the job title of Manager." Then you could use that component as a building block to construct policies that control whether managers can access and use the specified information. This means that whoever designs policies using policy components does not need any direct knowledge of who the managers are, or of who gains or loses the manager title in the future. They simply need to know how, as a class, managers should be allowed to access and use information.

Action components are a combination of one or more of the available basic actions defined in the resource type policy model. Action components are generated automatically when they are added from within the resource type.

You can use components in various combinations to construct the policies you need for managing information access and use. For example, you might define a user component "Contractors," and an action component "Copy, Move, Print, or Send via Email or IM." You could then combine these to implement an "eyes-only" rule: "All independent contractors may read company documents but may not transmit them in any way." When this rule is distributed to all enforcement clients in your network, it is enforced regardless of the individual contractor or document involved.

The process of defining components involves constructing a model of component categories that is comprehensive enough to represent all the actual entities in your organization that need to be covered by policies. You can either make a component model based on your knowledge of the network entities, regardless of how they may need to be used in policies; or based on the requirements of a set of logical policies that have already been designed.

 **Note:** You might find it expedient to construct a set of policies first, and then define the components you need for those policies.

Related concepts

[Constructing policies](#) on page 175

Policy designers construct policies to control information access and use in an organization.

Adding and editing policy components

You can add policy components to define subjects, resources, and actions in your environment any time as needed.

After policy components are defined, they become available to policy designers for use in policies.

Set up a policy model by creating resource types for use in components.

1. Log in to the interface with an account that has permission to add components.
2. On the left navigation bar, in the Components section, select a component type:
 - Subject
 - Resource
 - Action
3. Do one of the following:
 - Click **ADD COMPONENT**.
 - Click the name of an existing component to edit it.
4. Select the component type.
 - For subject component: Select one of the following subject type:

Subject Type	Description
Application	A Application program whose attributes may be referenced in a component.
Host	A computer whose attributes may be referenced in a component.
User	An actor whose attributes may be referenced in a component.

- For resource or action component: Select the resource type to use as the template for this component.

 **Note:** The list is empty if no resource types have been added.

5. Provide the component information. This information is displayed on the Components page and used to identify the component in the system:

Table 41: Component information

Option	Description
Display Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Description	A description of the item. This field is useful for explaining how the item is used and for documenting changes to the item. Descriptions can include multibyte characters, such as those used by Asian languages.

Option	Description
Tags	<p>Tags are labels that identify or classify objects. For example, you could apply a tag named Team1 to all documents related to the Team1 project. You could then search for documents with the Team1 tag, or set access requirements for documents with the Team1 tag.</p> <p> Note: Tags used in components are available only to components; component tags are not available to policies, and policy tags are not available to components.</p>

6. Do any of the following:

- Select the conditions to be used with the component. Conditions appear only if they have been added to the selected resource type. If you have multi-value attributes, you can specify multiple values by entering the value and pressing **Enter**.



CONDITIONS

1 Condition

Name	Operator	Value
First Name	is	John

+ Add Condition ADD CONDITION

You can click **PREVIEW** to display all of the individual entities that would be included in a component as it is presently defined. It displays at the right side of the window.

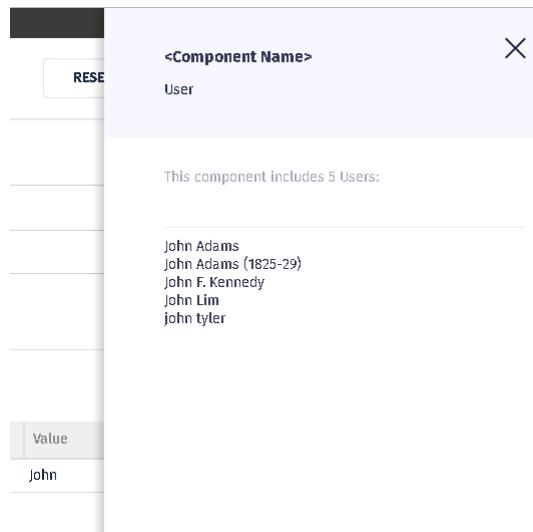


Figure 47: Preview component

- Select the actions to be available with the component. Actions appear only if they have been added to the selected resource type.
- Select the members you want to include or exclude in the component.
 - Click **ADD MEMBER CONDITION**.
 - Choose one of the available operators:
 - In
 - Not In
 - Specify the members by performing one of the following actions:
 - To search for members, begin typing in the field, then select the members.

- To add a member, type a name, select the check box next to the name, which is identified as (NEW), then click **APPLY**.
- (For subject type components only) To search for members, begin typing in the field, then click **LOOKUP**.

Figure 48: Members: Filter

A list of enrolled members appear. Select the members from the list of enrolled members. Based on the subject type you select, you have the following filters available:

Subject Type	Filters
Application	Components, Applications
Host	Components, Hosts, Host Groups. After selecting a host group, you can click the host group to view all members of the group and search for members of the group.
User	Components, Users, User Groups. After selecting a user group, you can click the user group to view all members of the group and search for members of the group.

d. Click **APPLY**.

To view the details, click the user, component, or the user group.

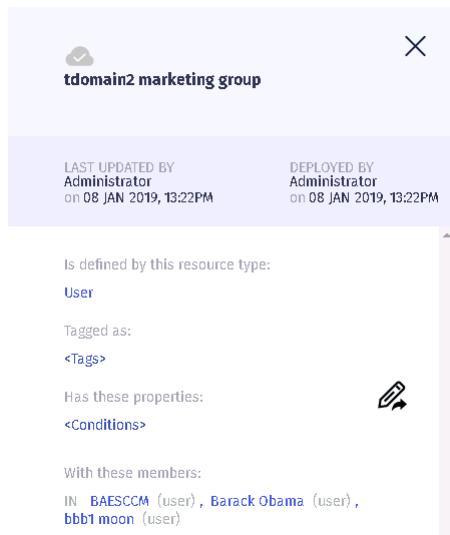


Figure 49: Component details

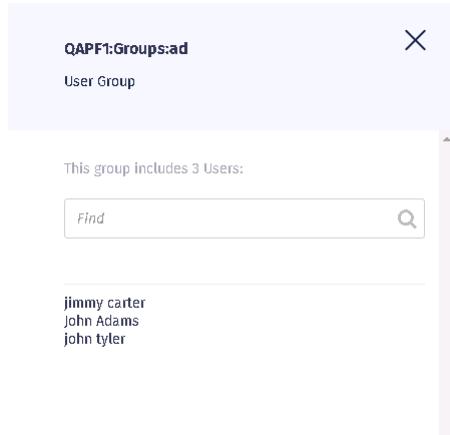


Figure 50: User group details

7. Do one of the following:

- Click **SAVE** to save the component in the Draft state.

 **Note:** If you chose **SAVE**, you must deploy the component before policies that use it can be enforced.

- Click **SAVE & DEPLOY** to make the component available for deployment with policies.

The Deploy dialog appears.

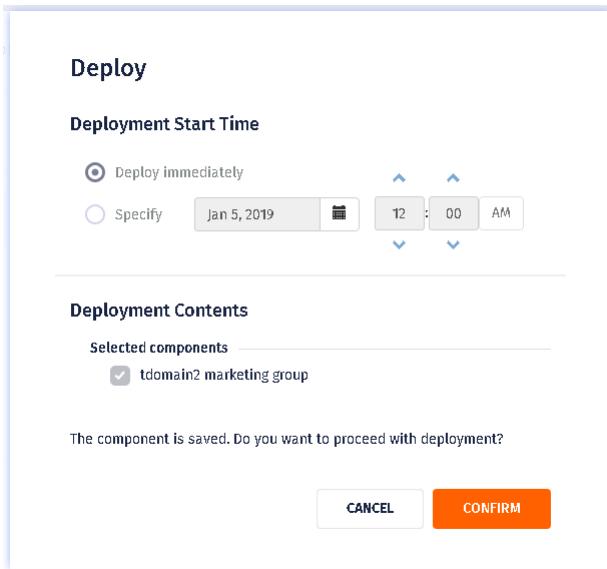


Figure 51: Deployment start time

8. If you selected **SAVE & DEPLOY**, select one of the following options:
 - Deploy immediately: The component is deployed immediately.
 - Specify: The component is deployed at the time that you specify in the future. The components state is Scheduled for Deployment.
 9. In Deployment Contents, look for the Modified components category. If present, it shows policy components that are needed by the object you are trying to deploy and that have been modified and resubmitted for deployment. Check the box next to any of these components that you want to schedule for deployment at the same time.
- Note:** In the Deployment Contents section, if you choose to clear any check box under Modified components, the system ignores the changes done to that component and uses the previously deployed component.

10. Click CONFIRM.

Create additional components or construct policies that use the components.

Related concepts

[Constructing policies](#) on page 175

Policy designers construct policies to control information access and use in an organization.

Related tasks

[Adding and editing resource types](#) on page 163

You can add or edit resource types at any time. To make attributes, actions, and obligations available to policy designers, resource types should be added before components are created.

[Deploying components](#) on page 195

Components used in policies must be deployed before policies that use those components can be enforced.

[Adding policies](#) on page 175

Chapter

9

Constructing and testing policies

Topics:

- [Constructing policies](#)
- [Using Advanced Conditions](#)
- [Managing subpolicies](#)
- [Viewing and editing policy and component information](#)
- [Cloning policies and components](#)
- [Using audit policies](#)
- [Constructing Delegated Administration policies](#)
- [Testing policies](#)

This section describes how to construct policies. Policies are rules designed to control information use in an organization.

They consist of components combined with logical operators, variables, obligations, and time-based contexts.

Constructing policies

Policy designers construct policies to control information access and use in an organization.

For example, a designer might construct the following policy:

Allow Users in IT to Edit and Reassign Tickets in their Product Area

The components used in this policy include:

- Subject component: IT Department (User)
- Action components: Edit and Assign (Support Tickets)
- Resource component: Tickets in User's Product Area (Support Tickets)

Policy designers use conditions to change a policy's effect based on dynamic comparisons, evaluations, or contextual factors. In addition, policies can include notification obligations, such as notifications to the end user or email warnings to an administrator whenever a policy is enforced anywhere in the network.

Related concepts

[Overview of policy implementation](#) on page 159

Implementing policies involves several tasks.

[About component types](#) on page 168

You create components by specifying a component type and defining the component's properties.

Related tasks

[Adding and editing policy components](#) on page 169

You can add policy components to define subjects, resources, and actions in your environment any time as needed.

Adding policies

Configure a Policy Model and configure policy components.

1. Log in to the console with an account that has permission to add policies.
2. On the left navigation bar, click **Policies**.
3. Click **ADD POLICY**.
4. Provide the policy information. This is used to identify the policy on the Policies list and in the system:

Table 42: Policy information

Option	Description
Name	The display name of the item. Names cannot be longer than 255 characters, and cannot include the reserved characters \$, /, ?, \, &, or *. Names can include multibyte characters, such as those used by Asian languages.
Description	A description of the item. This field is useful for explaining how the item is used and for documenting changes to the item. Descriptions can include multibyte characters, such as those used by Asian languages.

Option	Description
Tags	<p>Tags are labels that identify or classify objects. For example, you could apply a tag named Team1 to all documents related to the Team1 project. You could then search for documents with the Team1 tag, or set access requirements for documents with the Team1 tag.</p> <p> Note: Tags used in policies are available only to policies; policy tags are not available to components, and component tags are not available to policies.</p>

5. Select the policy effect and subject components:

-  **Note:** If you do not specify a subject component, the policy applies to all subject components.

Table 43: Policy effect and subject components

Option	Description
Policy Effect	<p>The action taken when conditions in the policy are met.</p> <ul style="list-style-type: none"> Allow: Permit the listed Subjects to perform the task specified in the policy. Allowing a set of Subjects to perform a task does not mean that others are blocked from performing that action. Allow policies can never result in Deny responses. Allow policies can only result in Allow or Not Applicable responses. Deny: Do not permit the listed Subjects to perform the task specified in the policy, but allow all others to do so. Deny policies can result in Deny, Allow, or Not Applicable responses.
Subject Components (Add Condition)	<p>The subjects (users) to be governed by the policy.</p> <p>Conditions are expressions of predicates that refine the context in which policy actions are performed. Conditions can be selected only if attributes have been added for the subject type policy component.</p> <p>Using conditions, you can create policies that change their effect based on dynamic comparisons, evaluations, or contextual factors. For complex contextual factors, use Condition Expressions, which are written directly in ACPL. Click the plus icon next to the condition that you want to set.</p> <p> Important: If no subject components are specified, the policy applies to all subject components.</p>

Option	Description
And Recipient	<p>The recipients of communications to which this policy will apply. You can specify one or more recipients of communications. Recipients are represented as User components.</p> <p>If you leave this field blank, the policy will apply to all recipients, so if it is a Deny policy it will block users from sending the specified communications to any user.</p> <p>In order to support this feature, User components have two special properties, E-mail Address and Domain. You can use them to define a user component based individual email address or domain, just as you might use other properties like Location or Job Title. This is useful for creating user components representing, for example, an email blacklist (or whitelist), which you then use as the And Recipient field of your communication policy.</p> <p> Note: You can use wildcards in defining email addresses as user properties, but not with domains. When defining users based on domains, you must supply one or more whole domain strings.</p>

6. Select resource components. These are the information resources, such as documents, spreadsheets, to be governed by this policy. Do any of the following:

- To select from all existing resource components, click in the field, then select a resource component.
- To search for resource component, begin typing in the field, then select a resource component.
- To add a resource component, type a name, select the check box next to the name, which is identified as (NEW), then click **APPLY**. An empty resource component is added to the policy. The new resource component also appears on the Resource Components list. Edit the resource component to define its properties.
- To add multiple resource components, click **ADD CONDITION**, then select or add a resource component.
- To remove a resource component, click the X next to the resource component name.
- For an action that logically involves a From location and a To location—Moved, Renamed, or Copied—you can specify a target location for the action of each policy. This control is used to specify a distinct location where you want to either permit or prohibit copying or moving all resources covered by the policy. If you do not specify a target, the policy will apply whenever the specified From action occurs, without regard to a target location.

To specify a target location:

- a. Click **Target Location (Moved, Renamed, or Copied)**.
- b. From the drop-down list, select an operator:
 - Into: Use Into to specify a particular destination; for example, when the specified users attempt to copy a file into a particular directory.
 - Outside: Use Outside to specify anywhere but a particular destination.
- c. Specify the target location:
 - To search for file names, directories, storage locations (file servers and folders), or document or portal content components, begin typing in the field, then select the target location.
 - To add a target location, type a name, select the check box next to the name, which is identified as (NEW), then click **APPLY**.

7. Select action components. These are the actions to be governed by the policy. If you include more than one action, an OR operator is assumed. If you do not specify any action component, the policy is in effect for all actions.
8. Optional: In the Advanced Conditions section, add any PQL expressions you want to use in policy evaluation. Advanced conditions extend policy evaluation by defining expressions that cannot be defined using components. For example, if a third-party system needs to provide input into policy evaluation, you could construct an expression that invokes the external system using the following syntax:

```
call_function("Service_Name", "Function_Name", <arguments>)
```

For example:

```
call_function("SECURITY", "Check classification", user.classification, "TOP SECRET") = "yes"
```

In addition, you can use variables in expressions. For more information about conditional expressions.

- Connection type: This is the type of connection between the policy server and the enforcement point. If you set this to Remote, the policy is enforced only if users are connected to the computer remotely, via Remote Desktop Protocol (RDP). This is useful for restricting the actions remote users can take—use only certain applications, or access resources only on certain network servers.



Note: This condition can only be used in policies that are enforced by desktop-level enforcers. If you include it in a policy that is enforced on the server side—for example, by the Windows File Server or SharePoint Server enforcer—it does not work properly, since server-side enforcers cannot detect the end user's connection type.

- Heartbeat: This enables you to define policies that are triggered on the basis of how many minutes have passed since the Control Center Server has received a heartbeat message from some enforcer. When the value specified here is exceeded, the policy is enforced. The most obvious use of this feature is to protect the data on lost or stolen notebook PCs: once they have been disconnected from the network for a suspiciously long time, a policy might be enforced that, for example, encrypts or deletes all sensitive files on the local drive.

9. Choose the policy effective duration. This is the period of time when the policy is to be enforced. Setting a duration is useful if you have a policy that you only want to enforce during a particular time period. For example, allowing users to access resources during working hours only, or for the duration of a specific project:

Table 44: Policy effect duration

Option	Description
Always	Enforce the policy continuously.
Specific Days	Enforce the policy according to the specified schedule: <ul style="list-style-type: none"> • Specify a date range to enforce the policy during a specified period. • Specify a Start date to begin enforcing the policy on a specified date. • Specify the days of the week on which the policy is to be enforced. For example, select SUN and SAT to enforce the policy on weekends only (Sunday and Saturday).



Note: It is important to understand the difference between these time condition settings. If a user does something to a resource and it is within this specified time period, the policy is enforced; if it is outside that period, the policy is not enforced. However, the policy is evaluated in either case. Time is evaluated based on the clock running on the Policy Management Server, a Control Center component, which is determined by the time zone

where the Policy Management Server host is located. For example, if you are using the web console in a branch office in Buffalo, and the Control Center is installed on a server at company headquarters in Tulsa, the time you specify is interpreted as Tulsa time.

10.Select the obligations you want the system to perform when the situation described in the policy occurs. Choose **Audit activity** to log the event in the Activity Journal. Choose **Notify** to send an email alert to an administrator.

- The **To:** field, where you supply the email address or addresses to which notification messages are to be sent. Separate multiple addresses using a comma.
- The **Message:** field, where you supply the message that goes into the body text of the notification email. In addition to whatever you specify here, the body of each email also includes details about that instance of enforcement: the user, the location, the document name, and so on.
- All notification email messages automatically include the following information:
 - User who triggered the policy enforcement
 - Host where user was working
 - Application involved
 - Source file being used or accessed
 - Action attempted
 - Effect (allow/deny/when)
 - Policy name
 - Policy description
 - Date and time of enforcement
- The **from** address: By default, the **From** address field in all email notifications contains a placeholder value. The Control Center administrator can replace this with a real address by manually editing the configuration file. You can define only one **From** address, which is included in all email notifications for all policies.

11.In the DEPLOYMENT TARGET section, select one of the following:

- Auto Deployment: The policy is distributed to all relevant enforcers in the system.
- Manual Deployment: The policy is distributed to only the enforcers that you specify.

12.Do one of the following:

- Click **SAVE**.

If you chose **SAVE**, you must deploy the policy to distribute it to enforcers. Policies are enforced only after they are deployed and the appropriate enforcers are configured.

- Click **SAVE & DEPLOY**.

The Deploy dialog appears.

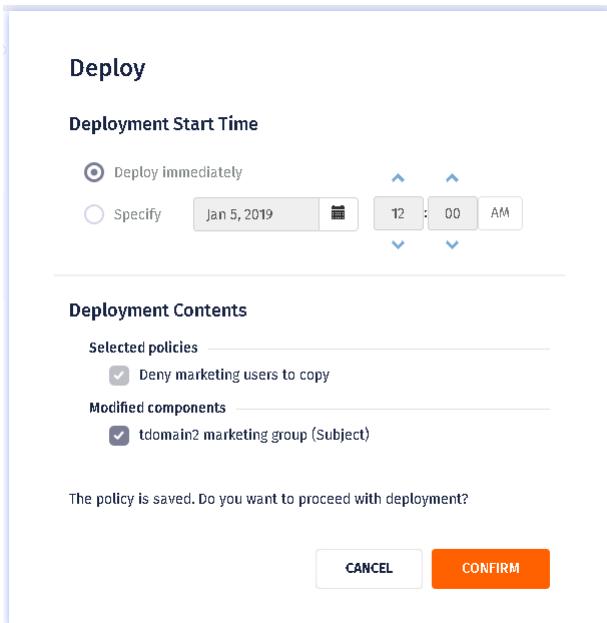


Figure 52: Deployment start time

13.If you clicked **SAVE & DEPLOY**, select one of the following options:

- Deploy immediately: (Default). Deploys the policy right away, without waiting for heartbeat messages from the enforcers. This is called push deployment.
- Specify: Displays the current time by default, but you can use the calendar and time widgets to set any specific minute you want deployment to begin. The policy state is Deployment Pending.

14.In Deployment Contents, look for the Modified components category. If present, it shows policy components that are needed by the object you are trying to deploy and that have been modified and resubmitted for deployment. Check the box next to any of these components that you want to schedule for deployment at the same time.

 **Note:** In the Deployment Contents section, if you choose to clear any check box under Modified components, the system ignores the changes done to that component and uses the previously deployed component.

15.Click **CONFIRM**.

Related concepts

[Policy and Policy Model overview](#) on page 158

[Using Advanced Conditions](#) on page 181

Advanced Conditions are ACPL (Active Control Policy Language) expressions that can be used to include parameters that cannot be conveyed by policy components, such as user and resource components.

[Configuration tools](#) on page 134

This section describes the tools available for modifying the configuration of Control Center after it is installed.

[Deploying and managing objects](#) on page 194

Related tasks

[Adding and editing policy components](#) on page 169

You can add policy components to define subjects, resources, and actions in your environment any time as needed.

[Adding subpolicies](#) on page 185

You can add subpolicies for any policy.

[Cloning policies](#) on page 186

Duplicating policies is referred to as policy cloning. When you clone a policy, any subpolicies within the policy are also cloned.

Using Advanced Conditions

Advanced Conditions are ACPL (Active Control Policy Language) expressions that can be used to include parameters that cannot be conveyed by policy components, such as user and resource components.

These conditions can be used to change a policy's effect based on relationships between resource, subject, and host attributes that are dynamically returned on a policy query.

For instance, a condition might compare the attribute values returned for a resource (the Product Line a document is associated with) with the attribute values returned for a subject (the Product Line a user is associated with). The syntax for this condition would be:

```
resource.<resource type>.<attribute> = subject.<attribute>
```

The Advanced Condition would be:

```
resource.ProductLine = user.ProductLine
```

In addition, policy designers can combine multiple attribute-to-attribute match conditions, as well as other matches (attribute-to-string, attribute-to-constant, attribute-to-integer). Matches can also be defined using different methods, listed in the following table. Advanced Conditions use Boolean logic, so all Match Operators can be reversed to "Not." Where multiple matches are combined, they can be linked using a logical "AND" or "OR". Finally, partial and full wildcards can be used to replace all or parts of strings. Examples with explanations are shown in the following table.

Table 45: Match operators for advanced conditions

Name	ACPL Operator	Description
Equal or Multi-value equal (match any)	=	When a single value is present for each attribute, A = B when A and B values match. When multiple values are present for attributes, {A} = {B} when any value in set {A} matches any value in set {B}.
Multi-value unordered equal	equals_unordered	When multiple values are present for each attribute, {A} equals_unordered {B} when set {A} and set {B} are identical, where order is not relevant.
Set includes	includes	Where multiple values are present for attributes, {A} includes {B} when all values in set {B} are present in set {A}. This can also apply where {B} is an empty set. Where multiple values are present for set {A} and a single value is present for B, set {A} includes {B} when the single value in {B} is present in set {A}.
Greater than/equal to, >, >=, <, <= less than/equal to	>, >=, <, <=	A matches B, where the value returned for attribute A is greater than; greater than or equal to; less than; or less than or equal to the value of B.

Related tasks

[Adding and editing resource types](#) on page 163

You can add or edit resource types at any time. To make attributes, actions, and obligations available to policy designers, resource types should be added before components are created.

[Adding policies](#) on page 175

Using wildcards in Advanced Conditions

You can use wildcard characters in attribute values. Generally they are useful to compare URL strings, filenames, and other attributes with patterns.

The following table shows supported wildcards.

Table 46: Wildcards used in Advanced Conditions

Wildcard character	Description
*	Matches zero or more characters, not including the path separator “/”
**	Matches zero or more characters, including the path separator “/”
d	Matches any single-digit numeral, 0-9
D	Matches any numeral, one or more digits
a	Matches any single letter, either upper- or lowercase
A	Matches one or more letters, either upper- or lowercase
c	Matches any single alphanumeric character or special character, such as %, &, _, etc., except for backslashes, which are reserved
C	Matches one or more alphanumeric characters or special characters, such as %, &, _, etc., except for backslashes, which are reserved
s	Matches a single space
S	Matches one or more spaces

With the exception of the standard * and **, all wildcards consist of a letter preceded by an escape character. Two escape characters are supported, a question mark and an exclamation point:

- ?<wildcard> means Matches the wildcard. This is useful for general purposes, such as matching policies to existing resources.
- !<wildcard> means Does Not Match the wildcard. Beyond general matching purposes, this is useful to ensure that new filenames conform to required conventions.

If the character following the ? or ! is not one of the wildcard characters listed above, it is assumed to represent itself. Thus ?z is just the letter z. More usefully, ?? is a literal question-mark and ?! is a literal exclamation point.

Different letters represent different kinds of characters, and each has two versions: lowercase stands for a single character of a certain type, and uppercase stands for one or more of those characters. This wildcard convention enables you to define wildcard-based conditions with a great deal of precision. For example:

- `http://www.mycompany.com*.html` matches any html file at the top level of `http://www.mycompany.com`
- `http://www.mycompany.com/**/*.*.html` matches any html file anywhere in the URL
- `**.txt` matches any resource ending in ".txt"
- `**/secret*` matches any resource with a name that starts with "secret"
- `**/*secret*` matches any resource with a name that contains the word "secret" anywhere.
- `Q?drevue.doc` matches any resource with a name that starts with "Q", ends with "revenue.doc", and has exactly one single-digit number in between. For example, `\documents`

- \accounting\2revenue.doc would match, while \\documents\accounting\4-98revenue.doc would not match.
- s?c?c?c.* matches any resource with a name containing exactly three characters of any kind, in either upper- or lowercase.

Example condition for equal and multi-value equal

The following condition includes an attribute-to-attribute multi-value equal AND an attribute-to-string match.

```
(resource.fso.ProductLine = user.ProductLine) AND (resource.fso.ReviewStatus = "Approved")
```

In this example, the user attribute ProductLine must match the resource attribute ProductLine, while multiple attributes are possible for both. For a multi-value match, the “=” operator evaluates to true when any of the values match. In other words, if a document belongs to ProductLine A, B, and C, and the user belongs to ProductLine C, D, and E, the condition is met. The attribute-to-string match specifies that the resource attribute ReviewStatus must match the string “Approved.” Because a logical AND links the two conditions, both must match for the entire condition to be true.

Example condition for multi-value equals_unordered operators

The “equals_unordered” operator should be used in cases where all attribute values must match between two multi-value sets, where the order of values is not relevant.

A typical use case is for a policy that allows access only when a document is not export controlled, OR the user requesting access to the resource has ALL the proper export licenses associated with that resource.

```
(resource.fso.ExportControl="NO") OR (resource.fso.ExportLicense equals_unordered user.ExportLicense)
```

Example:

In the following example, there are four enrolled users with a multi-value attribute named Country.

User	Multi-value Attribute	Attribute Values
User 1	Country	Finland, France
User 2	Country	Finland, Germany
User 3	Country	France, Germany, Brazil
User 4	Country	Brazil, France

The following table shows the policy evaluation results for all these operators: =, !=, includes, and equals_unordered.

Example	Result
user.country = “Finland”	User 1, User 2
user.country = “Germany”, “France”	User 1, User 2, User 3, User 4
user.country = “Brazil”, “France”	User 1, User 3, and User 4
user.country != “France”	User 2
user.country includes “Brazil”, “France”	User 3 and User 4
user.country equals_unordered “Brazil”, “France”	User 4

Example condition for the includes operator

The “includes” operator should be used in cases where set {A} should be included within set {B}.

This is different from “equals_unordered,” as set {B} might have additional values not present in set {A}. This is also different from “=” as set {B} can be an empty set.

A possible use case for this operator could be where a user must have all the appropriate class (or security) attributes as a resource to be granted access to that resource. In other words, user set {A} must include resource set {B}, where the user has all the proper clearances associated with the resource. Notice that this use case is different than equal or multi-value equal, as set {B} can be empty. If the resource has no security settings associated with it, user set {A} can include the empty resource set {B}, and the user should be able to access the resource.

```
resource.fso.DocSecurity includes user.SecurityAccess
```

Example condition for greater than/less than

You can also use greater/less than, and greater/less than or equal to operators in Advanced Condition.

Some examples could include policies that match dynamic attributes to dates (“date last accessed” or “today’s date,” or the date a license or policy is still valid), or integers (number of times accessed).

```
resource.fso.LicenseEndDate >= current_time
```

Managing subpolicies

Subpolicies are policies that can be included in other policies; they are the children of parent policies.

Subpolicies are typically used to specify additional policy enforcement criteria in the form of exceptions. For example, if a parent policy specifies that access to sensitive data is denied to all, a subpolicy can specify an exception to that rule. The subpolicy might state that users with special clearance can access the sensitive data. Subpolicies are especially useful for policies that include or exclude many classes of subjects. Separating these exceptions into subpolicies makes it easier to read and manage policies.

Examples of subpolicies

The following examples, expressed in English, show typical ways to use subpolicies.

In the following example, a parent policy denies access to confidential resources. Three subpolicies specify exceptions to the parent policy, allowing three types of users—licensed users, executives, and members of the legal department—to access confidential resources.

```
Deny Access to Confidential Resources
    Allow Licensed Users
    Allow Executives
    Allow Legal Department
```

The next example shows a hierarchical set of policies. The top-level policy denies access to sensitive financial documents. The first subpolicy specifies an exception to the top-level policy: Members of the finance group are allowed access. The second subpolicy specifies an exception to the first subpolicy: contractors in the finance group are denied access. The third subpolicy specifies an exception to the second subpolicy: contractors in the finance group with special clearance are allowed access.

```
Deny Access to Sensitive Financial Documents
    Allow Finance Group
        Deny Contractors
            Allow Contractors with Special Clearance
```

As the examples show, subpolicies provide an intuitive way to express complex policies that specify multiple conditions or exceptions. The examples also show the best way to design policies: Start with a policy that applies broadly, and then enumerate specific exceptions to that general policy. This is an important design concept because it enables you to define policies that are modular and extensible.

Adding subpolicies

You can add subpolicies for any policy.

1. On the left navigation bar, click **Policies**.
2. Locate the policy to which you want to add a subpolicy.
3. In the policy row, open the action menu on the right, then select the **Add Subpolicy** button.
4. Define the properties of the subpolicy. Because a subpolicy is simply a policy within a policy, you define and deploy a subpolicy in the same way you define a parent policy.

When the subpolicy has been added, it appears on the policy list. It also appears in the Policy hierarchy section of the parent policy.

Related tasks

[Adding policies](#) on page 175

Viewing and editing policy and component information

You can view information related to policies and components, including the hierarchy among related policies or components, their change history, and their properties.

Viewing and editing policy and component hierarchy

You can view and edit the relationships between policies and components in the policy hierarchy. This is especially useful for viewing and editing policies that have subpolicies and subcomponents.

1. On the left navigation bar, do one of the following:
 - Click **Policies**.
 - Click a component type.
2. Click the name of the policy or component whose hierarchy you want to view.
3. In the left pane, click the hierarchy button.

 **Note:** The hierarchy button is available only for policies and components that have been saved.

Viewing policy and component version history

You can view the version history of policies and components in the policy properties.

This information includes the date and time of changes and the name of the user who made the changes.

1. On the left navigation bar, do one of the following:
 - Click **Policies**.
 - Click a component type.
2. Click the name of the policy or component whose history you want to view.
3. In the left pane, click the history button.

 **Note:** The history button is available only for policies and components that have been saved.

Viewing and editing policy and component properties

You can view and edit policy and component properties as needed.

1. To view or edit policy or subpolicy properties, follow these steps:

- a. On the left navigation bar, select **Policies**, then click the name of the policy or subpolicy you want to view or edit.
 - b. View or edit the policy properties as needed, then click **Save**.
2. To view or edit component properties:
 - a. On the left navigation bar, select the type of component whose properties you want to view or edit, then click the name of the component.
 - b. View or edit the component properties as needed, then click **Save**.

Cloning policies and components

When you need to create policies and components that are similar to existing policies and components, it is often more efficient to make a copy of existing items and then make the desired changes in the duplicate, rather than build the new policy and components from scratch.

Cloning policies

Duplicating policies is referred to as policy cloning. When you clone a policy, any subpolicies within the policy are also cloned.

1. On the left navigation bar, click **Policies**.
2. Locate the policy you want to clone.
3. In the policy row, open the action menu on the right, then select the **Clone** button. The cloned policy is saved in Inactive form and appears on the policy list.
4. Edit the cloned policy as needed.

 **Note:** The components used by cloned policies are not themselves cloned; rather, the cloned policy and the source policy share the original components. To change the definition of one or more components as well as the policy, you need to clone the components as well.

Related tasks

[Adding policies](#) on page 175

[Cloning components](#) on page 186

Duplicating components is referred to as component cloning. All types of components can be cloned.

Cloning components

Duplicating components is referred to as component cloning. All types of components can be cloned.

1. On the left navigation bar, click **Policies**.
2. In the Components section, select a component type, then locate the component you want to clone.
3. In the component row, open the action menu on the right, then select the **Clone** button. The cloned component is saved in Inactive form and appears on the component list.
4. Edit the cloned component as needed.
5. Click **Save** to clone the selected components.

If the component is based on any other components (these are called leaf components), those components are not cloned. Rather, the cloned copy and the source component share the original leaves. To change the definition of the leaf components, you need to clone the leaf components as well, using the same procedure.

Related concepts

[Managing policy components](#) on page 167

Policy components are the abstract building blocks or raw material of information control policies.

Related tasks

[Cloning policies](#) on page 186

Duplicating policies is referred to as policy cloning. When you clone a policy, any subpolicies within the policy are also cloned.

Using audit policies

There are two available enforcement types.

- Deny: Prevent the activity defined by the policy from happening
- Allow: Permit the activity defined by the policy to happen

You can use Allow to create “audit policies”—that is, cases where you do not need to block a certain action, but you do want to track the details about it, every time it occurs. This is a powerful tool for any kind of auditing: either during initial network auditing to discover what kinds of policies you need to implement, or in ex-post facto cases, such as proving that certain employees did or did not use specific documents in certain ways.

Creating audit policies

In many cases you may want to define pure notification policies with a logging obligation only, since real-time email may not be required for the purposes of establishing an audit trail. Similarly, in such cases you likely do not need a notification message to pop up on the subject’s PC. However, all three of these obligation options are available in any case.

You can use Allow policies in creative ways, such as to remind users about policies, approaching deadlines, or any other matter that may or may not be connected with resource use and access. Especially when used in policies with time-based contexts, this feature offers great flexibility in sending automatic notifications in response to specific user actions.

Remember that just because a user is allowed to open or use a resource that is the subject of an audit policy, it does not necessarily mean the policy was enforced in that case. For a policy to be enforced, all of its parts must evaluate to true: subject, action, resource, and time context.

Allow policy obligations

When you define a new policy with Allow, you can only associate On Allow, Monitor obligations with it.

Allow policies can never result in Deny responses. Allow policies can only result in Allow or Not Applicable responses.

About enforcement logging

Enforcement logging can be configured as an obligation, associated with each individual policy you construct.

By default, it is enabled for On Deny in all policies; this cannot be disabled because it is good practice to log every instance of a Deny enforcement.

By default, no obligations are enabled for On Allow, Monitor obligations for all policies. You can enable one or all for any given policy as needed.

Logging vs. email

To reduce email traffic and clutter in administrator mailboxes, avoid enabling email notifications for policies unless there is a real need to do so.

For the same reason, you should consider how often a policy is likely to be enforced before adding email notifications to it.

Logging is another matter. It should not present a problem to enable logging obligations for every policy you construct. These are stored in the Activity Journal, which should have adequate storage capacity for extensive logging. In fact, you are unlikely ever to define a policy whose enforcement you do not want recorded in the Activity Journal.

Constructing Delegated Administration policies

Delegated Administration policies enables administrators to manage user access to the Control Center interfaces as well as policies, and policy objects using rules based on user attributes and conditions.

Using attributes and conditions mirrors the way policies are used to control access to resources and eliminates the need to create and apply roles to users or groups of users.

Related concepts

[Manage access to the console](#) on page 20

Testing policies

After policies are saved, you can test them using the Control Center console.

Testing policies in the console simulates sending authorization requests and enables you to view responses without building PEPs (policy enforcement points) or setting up application interactions. This reduces testing time and enables you to:

- Construct authorization requests by selecting attributes.
- View example request and response payloads in JSON and XML formats.
- Verify that policies function as expected. For example, you can verify that PEPs pass all of the necessary attributes and actions in requests, and that PDPs (Policy Decision Points) provide the appropriate responses.
- Select multiple policies for testing to ensure that policies function correctly in combination. For example, one policy might allow users with the attribute Suppliers to access a customer list. Another policy might deny access to Suppliers unless their territory matches the territory of the customer list. Testing both policies at once enables you to verify that the policies perform as expected when combined.



Note: To test policies, you must use an account with permission to view the Policy Tester, policies, and Policy Model components.

Also, policies that are in the Inactive state, that is, policies that have previously been deployed but that are currently deactivated, cannot be tested until they are deployed again.

Before you begin, you must be familiar with the policies you want to test and the attributes specified in those policies. You need to select those attributes to generate authorization requests.

1. On the left navigation bar, click **Policy Tester**.

2. In the Select Policy section, choose the policies to be tested:

- **All Policies:** All policies you have access to, provided they are in the Inactive Draft, Deployed, or Deployed Draft state. Policies that are in the Inactive state, that is, policies that have previously been deployed but that are currently deactivated, cannot be tested until they are deployed again. Select this option to test for conflicts among all policies.
- **Select Policies:** Only the policies specified in this section. After you choose **Select Policies**, type the name of the policies you want to test to search for matching policies.
 - **Include Sub-Policies:** If you choose Select Policies, choose whether to include sub-policies when searching for the policies you want to select.
 - **Saved Search Filter:** If you have searched for policies on the Policy Management page, and you have saved a search, you can select it to include all the policies in the search results.

3. In the Request section, add the attributes you want to include in the test. For the first test, select attributes that match the policy.

For example, Allow Users in IT to View All Support Tickets, a sample policy included in the QuickStart Guide, has the Subject attribute IT Department.

Allow Users in IT to View All Support Tickets

POLICY EFFECT AND SUBJECT COMPONENT

Policy Effect: Allow

Subject Components: In IT Department

ADD CONDITION

LAST UPDATED BY Administrator on 13 JUL 2017, 14:59PM

Is defined by this resource type: User

Tagged as: Helpdesk

Has these properties: department is IT

And it includes the action View, which is defined by the resource type Support Tickets.

Allow Users in IT to View All Support Tickets

Resource Components

In Search for Resource

ADD CONDITION

ACTION COMPONENTS

Action Components: View Support Tickets

ADVANCED CONDITIONS

LAST UPDATED BY Administrator on 13 JUL 2017, 14:59PM

Is defined by this resource type: Support Tickets

Tagged as: Helpdesk

Has these actions: VIEW_TKTS

Includes these components: <Components>

To verify that the Allow Users in IT to View All Support Tickets policy works as expected, add the Subject attribute, Department, with the value IT.

Policy Tester

RESET

TEST REQUEST

Add SUBJECT attributes

Name	Value	ADD ATTRIBUTE
subject-id	chris.webber × Add Values	Delete
department	IT × Add Values	Delete

+ Add Attribute

Add RESOURCE attributes

Resource Type: Support Tickets

Action: View

4. To view the JSON build request, click the JSON button under the Request tab.

REQUEST RESPONSE LOGS

REQUEST **JSON** XML

```
{
  "Request": {
    "ReturnPolicyIdList": true,
    "Category": [
      {
        "CategoryId": "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject",
        "Attribute": [
          {
            "DataType": "http://www.w3.org/2001/XMLSchema#string",
            "AttributeId": "urn:oasis:names:tc:xacml:1.0:subject:subject-id",
            "IncludeInResult": false,
            "Value": "chris.webber"
          },
          {
            "DataType": "http://www.w3.org/2001/XMLSchema#string",
            "AttributeId": "urn:oasis:names:tc:xacml:1.0:subject:undefined",
            "IncludeInResult": false,
            "Value": "IT"
          }
        ]
      },
      {
        "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-category:resource",
        "Attribute": [
          {
            "DataType": "http://www.w3.org/2001/XMLSchema#string",
            "AttributeId": "urn:nextlabs:names:evalsvc:1.0:resource:resource-type",
            "IncludeInResult": false,
            "Value": "support_tickets"
          }
        ]
      }
    ]
  }
}
```

5. To view the XML build request, click the XML button under the Request tab.

```

<?xml version="1.0" encoding="UTF-8"?>
<Request
  xmlns='urn:oasis:names:tc:xacml:3.0:core:schema:wd-17'
  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' xsi:schemaLocation='urn:oasis:names:tc:xacml:3.0:core:schema:wd-17 urn:oasis:names:tc:xacml:1.0:subject-category:access-subject'
  <Attributes Category='urn:oasis:names:tc:xacml:1.0:subject-category:access-subject'>
    <Attribute IncludeInResult='false' AttributeId='urn:oasis:names:tc:xacml:1.0:subject:subject-id'>
      <AttributeValue>chris.webber</AttributeValue>
    </Attribute>
    <Attribute IncludeInResult='false' AttributeId='urn:oasis:names:tc:xacml:1.0:subject:undefined'>
      <AttributeValue>IT</AttributeValue>
    </Attribute>
  </Attributes>
  <Attributes Category='urn:oasis:names:tc:xacml:3.0:attribute-category:resource'>
    <Attribute IncludeInResult='false' AttributeId='urn:nextlabs:names:evalsvc:1.0:resource:resource-type'>
      <AttributeValue>support_tickets</AttributeValue>
    </Attribute>
    <Attribute IncludeInResult='false' AttributeId='urn:oasis:names:tc:xacml:1.0:resource:resource-id'>
      <AttributeValue>resource-name</AttributeValue>
    </Attribute>
  </Attributes>
  <Attributes Category='urn:oasis:names:tc:xacml:3.0:attribute-category:action'>
    <Attribute IncludeInResult='false' AttributeId='urn:oasis:names:tc:xacml:1.0:action:action-id'>
      <AttributeValue>VIEW_TKTS</AttributeValue>
    </Attribute>
  </Attributes>
</Request>

```

6. To see the PDP response, click **Test Request.**

Policy Tester

REQUEST RESPONSE LOGS

RESET TEST REQUEST

Decision : **ALLOW**

Evaluation Time : 1ms

OBLIGATIONS

The Evaluation Time is the amount of time the PDP took to make the decision. The Obligations section shows obligations required by the policy.

For another test, change the value of an attribute so that it does not match the policy. In the Allow Users in IT to View All Support Tickets example, change the value of the Department attribute to something other than IT, such as HR.

7. For another test, change the value of an attribute so that it does not match the policy. In the Allow Users in IT to View All Support Tickets example, change the value of the Department attribute to something other than IT, such as HR.

REQUEST RESPONSE LOGS

Add SUBJECT attributes

Name	Value	
subject-id	chris.webber × Add Values	Delete
department	HR × Add Values	Delete

+ Add Attribute

8. To view the response, click **Test Request**.

SELECT POLICIES

Select Policies

Allow Users in IT to View All Support Tickets ×

Search for Policies

 Include Sub-Policies

REQUEST RESPONSE LOGS



Decision :

NOT APPLICABLE

Evaluation Time :

1ms

OBLIGATIONS

Note: In production environments, NextLabs recommends that you deny access to resources when the response is Not Applicable.

9. For information about how the policy was applied, click the **Logs** tab.

REQUEST RESPONSE LOGS

```
New log created on : Fri Jul 14 2017 09:39:24 GMT-0700 (Pacific Daylight Time)

Policies Tested
  Policies with Allow Effect:
    ROOT_244/Allow Users in IT to View All Support Tickets
  Policies with Deny Effect:
    None
  Policies that did not Match:
    None

Request Parameters
  application
    name: application
  host
    inet_address: 2130706433
  environment
  subject
    id: chris.webber
    department: IT
  fromResource
    urn:nextlabs:names:evalsvc:1.0:resource:resource-type: support_tickets
    id: null
    ce::id: resource-name
    ce::destinytype: object
  action
    name: VIEW_TKTS
  policies
    ignoredefault: true
  Ignore obligation = false
  Process Token = 0
  LogLevel = 0
```

10.To clear the policy selection and results, click **Reset**.



Note: Test results do not appear in the Activity Log, and results are cleared when you refresh the browser.

Chapter

10

Deploying and managing objects

Topics:

- [About deployment](#)
- [Deploying components](#)
- [Deploying policies](#)
- [Managing policies and components](#)

This section explains how to deploy and manage objects, such as policies and components.

Related concepts

[Overview of policy implementation](#) on page 159

Implementing policies involves several tasks.

[Managing policy components](#) on page 167

Policy components are the abstract building blocks or raw material of information control policies.

Related tasks

[Adding policies](#) on page 175

About deployment

Deployment places objects into a state where policies can be sent to policy enforcement points (PEPs) at the next heartbeat and enforced throughout the network.

Organizations might have specific requirements for controlling changes to computer systems, so a single person is typically assigned to control policy deployment.

Policies are enforced only after they are deployed and distributed to PEPs on PCs or servers where policy enforcement is required.

Deploying components

Components used in policies must be deployed before policies that use those components can be enforced.

1. On the left navigation bar, select **Components**, then select a component type.
2. On the component list, do one of the following:
 - Locate the component you want to deploy, then select the deploy button in the Action menu at the right of the component row.
 - Select the check boxes next to the components you want to deploy, then click the deploy button on the left above the list.

The components are deployed and available for deployment in policies.

Related tasks

[Adding and editing policy components](#) on page 169

You can add policy components to define subjects, resources, and actions in your environment any time as needed.

[Modifying policies and components](#) on page 197

As conditions change over time, you might want to change existing policies and components.

Deploying policies

Policies must be deployed before they can be enforced.

1. On the left navigation bar of Control Center, select **Policies**.
2. On the Policies list, do one of the following:
 - Locate the policy you want to deploy, then select the deploy button in the Action menu at the right of the policy row.
 - Select the check boxes next to the policies you want to deploy, then click the deploy button on the left above the list.

Control Center automatically performs a dependency check to see whether any components used in the policy are in the draft state and must be deployed before the policy is deployed.

3. If dependencies are identified, verify that all components used in the policy are deployed. Policies can be enforced correctly only after all components used in the policy are deployed.

Managing policies and components

Policy management tasks include:

- Searching and filtering policy and component lists
- Modifying policies and components
- Changing user access to components
- De-activating policies and components

Related concepts

[Overview of policy implementation](#) on page 159

Implementing policies involves several tasks.

Searching and filtering policy and component lists

Control Center enables you to search for policies, components, and Policy Models, which is especially useful for finding items quickly when you have long lists of policies, components, or Policy Models.

In addition, search criteria can be used as filters. When search criteria are applied to a list, the system displays only those items that match the criteria.

 **Note:** By default, the All Tags check box is enabled and the system displays both tagged and untagged items. You can select specific tags to filter your search results. If you select the No Tags check box, the system only displays untagged items.

1. Log in to the Control Center interface with an account that has permission to edit policies or components.
2. To search for and filter policies:
 - a. On the left navigation bar, click **Policies**, then click the magnifying glass button in the upper right of the page.
 - b. Select search criteria, then click **Apply**. The page shows policies that match the criteria. When you close the search panel, the list continues to be filtered according to the search criteria and **Filters Applied** appears at the top of the list.
 - c. If a filter is applied, click the X next to **Filters Applied** to remove it and display all list items.
3. To search for and filter components:
 - a. Select a component type on the left navigation bar, then click the magnifying glass button in the upper right of the page.
 - b. Select search criteria, then click **Apply**. The page shows policies that match the criteria. When you close the search panel, the list continues to be filtered according to the search criteria and **Filters Applied** appears at the top of the list.
 - c. If a filter is applied, click the X next to **Filters Applied** to remove it and display all list items.
4. To search for Policy Model components (resource types and subject types):
 - a. On the left navigation bar, click **Policy Model**, then click the magnifying glass button in the upper right of the page.
 - b. Select search criteria, then click **Apply**. The page shows policies that match the criteria. When you close the search panel, the list continues to be filtered according to the search criteria and **Filters Applied** appears at the top of the list.
 - c. If a filter is applied, click the X next to **Filters Applied** to remove it and display all list items.
5. To save a search:
 - a. Enter search criteria, then click **Apply**.
 - b. Click **Save this search**.
 - c. Provide a name and description for the search, then click **Save**. The search criteria is saved, and you can access it any time by selecting it in the Saved Searches drop-down list.
6. To access a saved search:
 - a. On the left navigation bar, select the type of item the search relates to, then click the magnifying glass button in the upper right of the page.

- b. In the drop-down list at the top of the page, select a saved search. Only those searches related to the selected item type appear on the list.

Viewing policy version information

You can view the following version policy information:

- Creation date
 - Modification date
 - Deployment date
1. Log in to the Control Center interface with an account that has permission to view policies.
 2. On the left navigation bar, click **Policies**. The modification date for each policy is displayed.
 3. Click the name of a policy.
 4. On the policy detail page, in the left pane, click the version history icon. The policy creation date and deployment dates are displayed.

Modifying policies and components

As conditions change over time, you might want to change existing policies and components.

For example, if a policy allows only Human Resources personnel to view offer letters, and the company CEO expresses an interest in viewing such letters, the policy would need to be modified to enable access for the CEO.

1. Log in to the Control Center interface with an account that has permission to edit policies or components.
 2. On the left navigation bar, click **Policies or Components > <Component type>**.
 3. Click the name of the policy or component.
-  **Note:** If the name is grayed-out, you do not have sufficient privileges to modify the object.
4. On the policy or component detail page, make the desired changes, then click **Save** or **Save & Deploy**.

If you chose Save, you must deploy the object before it can be used in policies or enforced.

Related tasks

[Deploying components](#) on page 195

Components used in policies must be deployed before policies that use those components can be enforced.

Changing user access to components

User access to components, policies, and other objects is determined by delegation policies.

Related concepts

[Manage access to the console](#) on page 20

De-activating policies and components

You deactivate policies and components to un-deploy them. When you deactivate a policy, it is removed from deployment and is no longer in active use.

If you deactivate a top-level policy, all subpolicies are also deactivated.

You might want to deactivate:

- A component that was never used in any policy.
- A component that represents a concept that is no longer useful in your company; for example, a User Group component that represents the employees in a division that has been sold.
- A policy that is no longer useful—for example, one covering documents that were confidential during project development, but have been made public since the project finished.

When policies are deactivated, they are removed from policy enforcement points at the next heartbeat.

1. Log in to the Control Center interface with an account that has permission to edit policies or components.
2. On the left navigation bar, click **Policies or Components**.
3. Locate the policy or component you want to deactivate, then select the deactivate button from the Action menu on the right side of the row. This deactivates the object. Policies that have been deactivated are removed from policy enforcers at the next heartbeat.

Deleting objects

After you deactivate a policy or component, it remains in the system so that you can reference it later. If you are sure you do not want to redeploy an item, you can delete it permanently.

1. Log in to the Control Center interface with an account that has permission to view policies.
2. On the left navigation bar, click **Policies or Components**.
3. Locate the policy or component you want to deactivate, then select the trash can button.

Chapter

11

Exporting and Importing Policies

Topics:

- About exporting and importing policies and other items
- Exporting and importing objects using the Control Center interface
- Exporting and importing policies securely
- Using versions
- About object states

This section describes how to export and import policies.

About exporting and importing policies and other items

Exporting and importing policies enables you to share policy objects defined in one instance of Control Center with other instances of Control Center.

This is helpful in collaborative environments where policy construction responsibilities are distributed among several policy designers. It is also useful for transferring policies and components from isolated test environments to live production environments.

Policies, and any components used in policies, are exported as BIN files, which can be imported into other Control Center instances.

 **Important:** The following items cannot be exported or imported:

- Individual components, such as subject, resource, and action components. Components can be exported and imported only if they are included in policies.
- Policy model components, such as resource types. These can be exported and imported only if they are included in policies.
- Delegation policies. Delegation policies cannot be imported or exported and must be created in each instance of Control Center separately as needed.

Exporting and importing objects using the Control Center interface

Policies and their related components can be exported and imported using the Control Center interface.

To avoid creating inconsistent policy sets, follow these guidelines:

- Perform exports or imports at a time when you are certain no one is using policy instances.
- Request that all policy analysts log out of Control Center when any policies are being imported.
- Publish the list of policies and components to be imported, and ask all policy analysts to refrain from editing any object in that list until the procedure is complete.

1. On the left navigation bar, click **Policies**.

2. On the Policy Management page, do any of the following:

- Select the export button in the upper right to export all policy objects.
- Select the import button in the upper right to import policy objects.
- Select the policy objects you want to export from the list, then click the export button above the list.

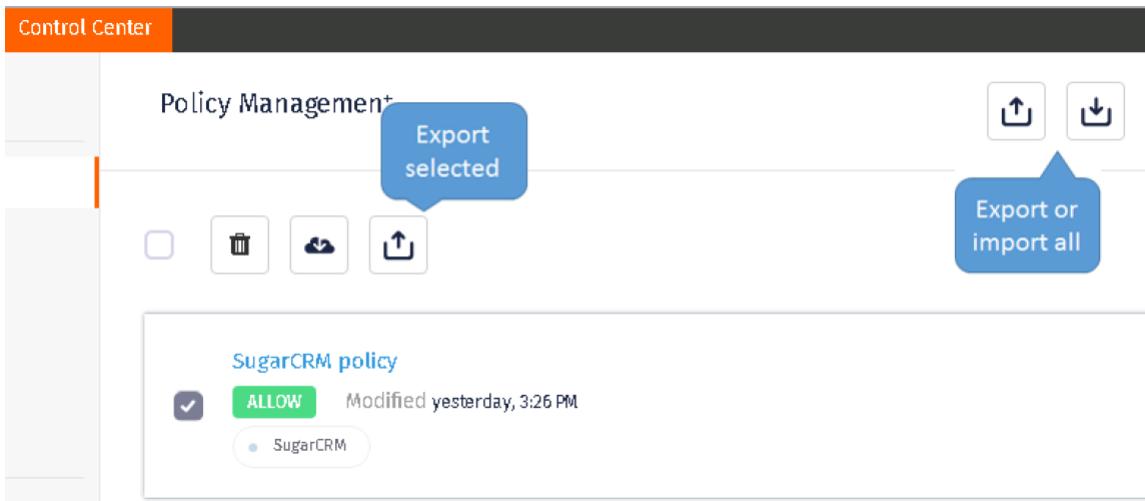


Figure 53: Export and import controls

When policies are exported, all subpolicies, components, and associated policy model components, such as resource types, are exported to a file with the extension .bin.

When importing policies:

- All subpolicies, components, and associated policy model components, such as resource types, are imported.
- With the exception of the User subject type, all policies or components whose component names or short names match those being imported are overwritten. For example, if the imported file contains a resource type named Support_Ticket and a resource type with the same short name already exists in the environment, the existing resource type is overwritten with the imported version.
- When the User subject type is imported, all new attributes are added, but the existing User subject type attributes are not overwritten or removed.

Exporting and importing policies securely

Control Center enables authorized users to export and import policies and their components that are signed and encrypted using the Advanced Encryption Standard (AES-256) specification.

The encryption enables authorized users to securely transfer policies and components from isolated test environments to live production environment. Authorized users can view or edit the policies only after they import the encrypted policy bundle into any Control Center instance using the same shared key.

If the administrator has selected the Signed and encrypted (with shared key) option during Control Center Installation, Control Center signs the data that you are exporting with its digital signature's private key, encrypts with a shared key, and exports to a file with the extension .ebin. Otherwise, administrators can manually enable encryption by configuring the cc-console-app.properties file.



Note: If you change the shared key, the system cannot decrypt or import previously exported .ebin file.

Related tasks

[Securely exporting and importing objects using the Control Center interface on page 203](#)

[Manually configuring properties for exporting or importing encrypted policy bundles on page 204](#)

You can manually configure the cc-console-app.properties file which enables you to export signed and encrypted policy bundles.

Securely exporting and importing objects using the Control Center interface

Policies and their related components can be securely exported and imported using the Control Center interface.

To avoid creating inconsistent policy sets, follow these guidelines:

- Perform exports or imports at a time when you are certain no one is using policy instances.
- Request that all policy analysts log out of Control Center when any policies are being imported.
- Publish the list of policies and components to be imported, and ask all policy analysts to refrain from editing any object in that list until the procedure is complete.

1. On the left navigation bar, click **Policies**.

2. On the Policy Management page, do any of the following:

- Select the export button in the upper right to export all policy objects.
- Select the import button in the upper right to import policy objects.
- Select the policy objects you want to export from the list, then click the export button above the list.

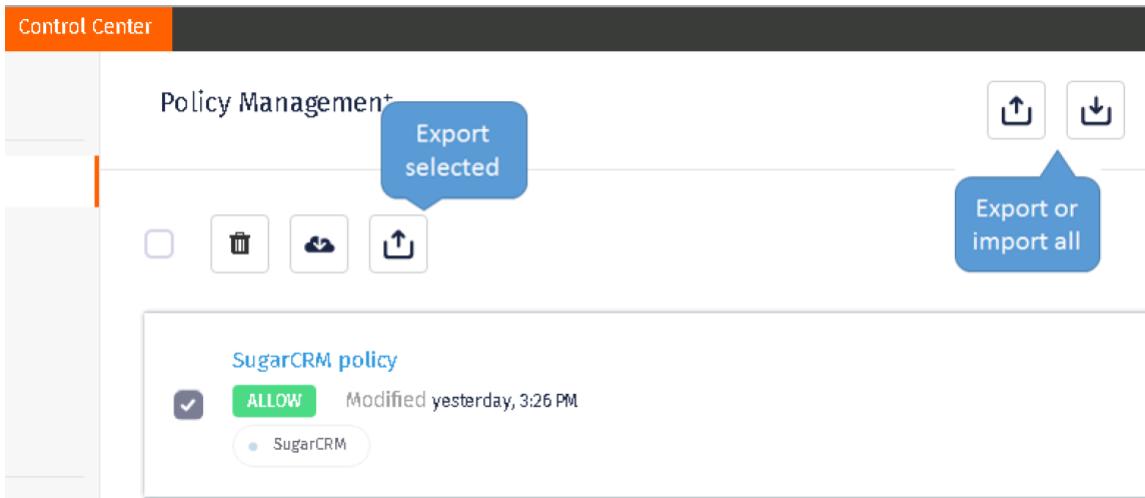


Figure 54: Export and import controls

The Export Format dialog appears if you have selected the option to allow export plain text data, during Control Center installation.

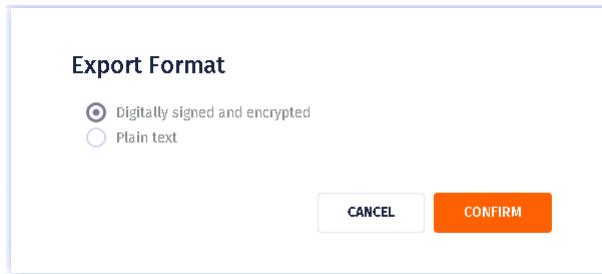


Figure 55: Export format

3. Select one of the following option.

- **Digitally signed and encrypted:** Exports policies and their components that are signed and encrypted using the Advanced Encryption Standard (AES-256) specification. All subpolicies, components, and associated policy model components, such as resource types, are exported to a file with the extension .ebin.

- **Plain text:** Exports policies and their components are exported in the plain text format. All subpolicies, components, and associated policy model components, such as resource types, are exported to a file with the extension .bin.

4. Click CONFIRM.

When importing policies:

- Ensure that you use the shared key and import the digital certificate (`digital-signature.cer`) into other Control Center instances where you are planning to import the `.ebin` file.
- All subpolicies, components, and associated policy model components, such as resource types, are imported.
- With the exception of the User subject type, all policies or components whose component names or short names match those being imported are overwritten. For example, if the imported file contains a resource type named `Support_Ticket` and a resource type with the same short name already exists in the environment, the existing resource type is overwritten with the imported version.
- If the User subject type is imported, all new attributes are added, but the existing User subject type attributes are not overwritten or removed.

Related concepts

[Exporting and importing policies securely](#) on page 202

Control Center enables authorized users to export and import policies and their components that are signed and encrypted using the Advanced Encryption Standard (AES-256) specification.

Related tasks

[Manually configuring properties for exporting or importing encrypted policy bundles](#) on page 204

You can manually configure the `cc-console-app.properties` file which enables you to export signed and encrypted policy bundles.

[Encrypting passwords](#) on page 144

Whenever you generate a new password for the configuration file, you should encrypt it.

Manually configuring properties for exporting or importing encrypted policy bundles

You can manually configure the `cc-console-app.properties` file which enables you to export signed and encrypted policy bundles.

Follow this procedure if you have not selected the option to export signed and encrypted policy bundles during installation.

Perform these steps on every Control Center instance where you want to export signed and encrypted policy bundles.

1. Stop the Control Center Policy Server service.

2. In the Control Center host server, navigate to:

`<installation folder>/PolicyServer/server/configuration`

3. Using any text editor, open the `cc-console-app.properties` file.

4. Search for the `Data transportation configurations` section and change the values as shown in the following example:

```
data.transportation.mode=SANE  
data.transportation.shared.key=<AES-256-encrypted-shared-key>  
data.transportation.allow.plain.text.import=true  
data.transportation.allow.plain.text.export=true
```



Note: Set the value of the parameters,

`data.transportation.allow.plain.text.import` and

`data.transportation.allow.plain.text.export` to `false` if you want to enforce only signed and encrypted format.

5. Save and close the `cc-console-app.properties` file.
6. Restart the Control Center Policy Server service.

Related concepts

[Exporting and importing policies securely](#) on page 202

Control Center enables authorized users to export and import policies and their components that are signed and encrypted using the Advanced Encryption Standard (AES-256) specification.

Related tasks

[Securely exporting and importing objects using the Control Center interface](#) on page 203

Importing the digital signature certificate for importing encrypted policy bundles

Consider the following example. You are importing the certificate from machine A (where policies are exported) to machine B (where policies are being imported). To import the policies on machine B, you must copy the `digital-signature.cer` from machine A and import that certificate into machine B `digital-signature-keystore.jks`.

1. Copy the `digital-signature.cer` from the Control Center instance (where policies are exported) to a folder in the Control Center instance (where policies are being imported).
2. To import the `digital-signature.cer` certificate from the Control Center instance you are importing the policy bundles, run the following command:

```
keytool -import -alias <hostname> -noprompt -file <digital-signature.cer> -keystore <digital-signature-keystore.jks> -storepass <storepass>
```

 **Important:** The `<hostname>` value must exactly match the hostname or FQDN of the Control Center instance (where policies are exported).

 **Note:** If you want to use the `keytool` utility and if there is no default JRE installed on the Control Center instance (where policies are being imported), you must navigate to `<installation_folder>/java/jre/bin` to run the `keytool` command.

3. To view your certificate information in a keystore, run the following command:

```
keytool -list -keystore <digital-signature-keystore.jks> -v
```

Using versions

The version mechanism is useful in keeping track of changing objects whose definitions evolve, so you can see the definitions of previously deployed policies.

This is most useful for auditing purposes, in cases when you need to demonstrate that a certain set of policy controls were in effect at a certain time.

Each newly created object, in Draft state, starts as Version 0. Once it is deployed, it becomes Version 1. Thereafter, every time the object moves out of Deployed state and is then redeployed, its version increments by one. The version number increments simply by virtue of changing back into Deployed state, regardless of whether any changes have been made to the definition of the object. A date stamp is permanently assigned to each incremental revision, which provides a useful tool for managing and keeping track of versions.

When working with versions, it is important to understand the distinction between an object's state and its status. Different versions of the same object can have different statuses, and in such cases, the same object is listed in more than one row in the Object Grid. In addition, if an object has undergone several changes, the different versions of it are listed in the grid, with the same status—for example, all obsolete versions would be listed on the Deactivated tab, showing Inactive status.

About object states

The following table lists the possible states of components, policies, and Policy Model objects.

Table 47: States and Statuses

Possible Statuses	Description
Inactive Draft	Open for editing. The object has been created and saved, but has not yet been submitted for deployment, or has been taken out of the deployed state for editing.
Deployed	The object has been deployed, and its definition has been sent to all relevant enforcers.
Deployed Draft	The object has been taken out of the deployed state for editing. An earlier version is deployed.
Deployment Pending	The object is scheduled for a specific time deployment.
Deployment Pending (An earlier version is deployed)	If an object is already deployed and a user selects to deploy an object at a specific time.
Inactive	The object has been deactivated, and the deactivated version has been deployed to enforcers.

Chapter

12

Managing Reports

Topics:

- Introducing Reporter
- About the Reporter console
- Using the Reporter Dashboard
- Working with reports
- Working with monitors and alerts

Introducing Reporter

Reporter is the web-based interface used to generate reports on information use and policy enforcement in your environment.

About Reporter

Reporter is a web-based interface that is installed as a component of the console.

You use Reporter to define and run custom queries about policy enforcement activities that are recorded in the Activity Journal. These queries are referred to as reports. Reports can be designed to answer a wide variety of questions, such as who has access to certain documents; who is using which resources, and when; what types of policy enforcement is taking place; what activity occurred within a given department; and so on.

In addition to reports, you can use Reporter to create monitors that trigger alerts when specified policy enforcement criteria are met. You can design monitors to cover a wide range of scenarios, such as sending an alert through email when access to a specified resource has been denied more than a specified number of times in a given time period; or when the classified documents that have been downloaded in a given time period exceed a specific file size. Together, monitors and alerts provide continuous coverage of critical policy enforcements in an enterprise, as well as a notification system that alerts administrators when action is required.

Reporter is intended for use by whoever is responsible for monitoring and reporting on compliance, gathering statistics about document usage, and investigating any suspected incidents of information mishandling. This may include administrators, IT staff, managers, executives, auditors, and any other authorized personnel.

Access to the Reporter console is determined by delegated administration policies.

Related concepts

[Manage access to the console](#) on page 20

Related tasks

[Adding and editing delegation policies](#) on page 21

Creating and editing delegation policies is similar to creating and editing the policies that govern access to resources.

Accessing the Reporter console

Access Reporter from the Control Center console.



Note: If you are using Firefox, you cannot access the Reporter console unless you enable TLS 1.0 (under Tools > Options > Advanced).

1. Open a web browser and log in to the console.
2. On the left navigation bar, click the **Reports** button. The Reporter console opens in a new browser window.

About the Reporter console

The Reporter console has three tabs: Dashboard, Reports, and Monitoring.

By default, the **Dashboard** tab appears when Reporter opens. Only users granted access to monitors can access the **Monitoring** tab.

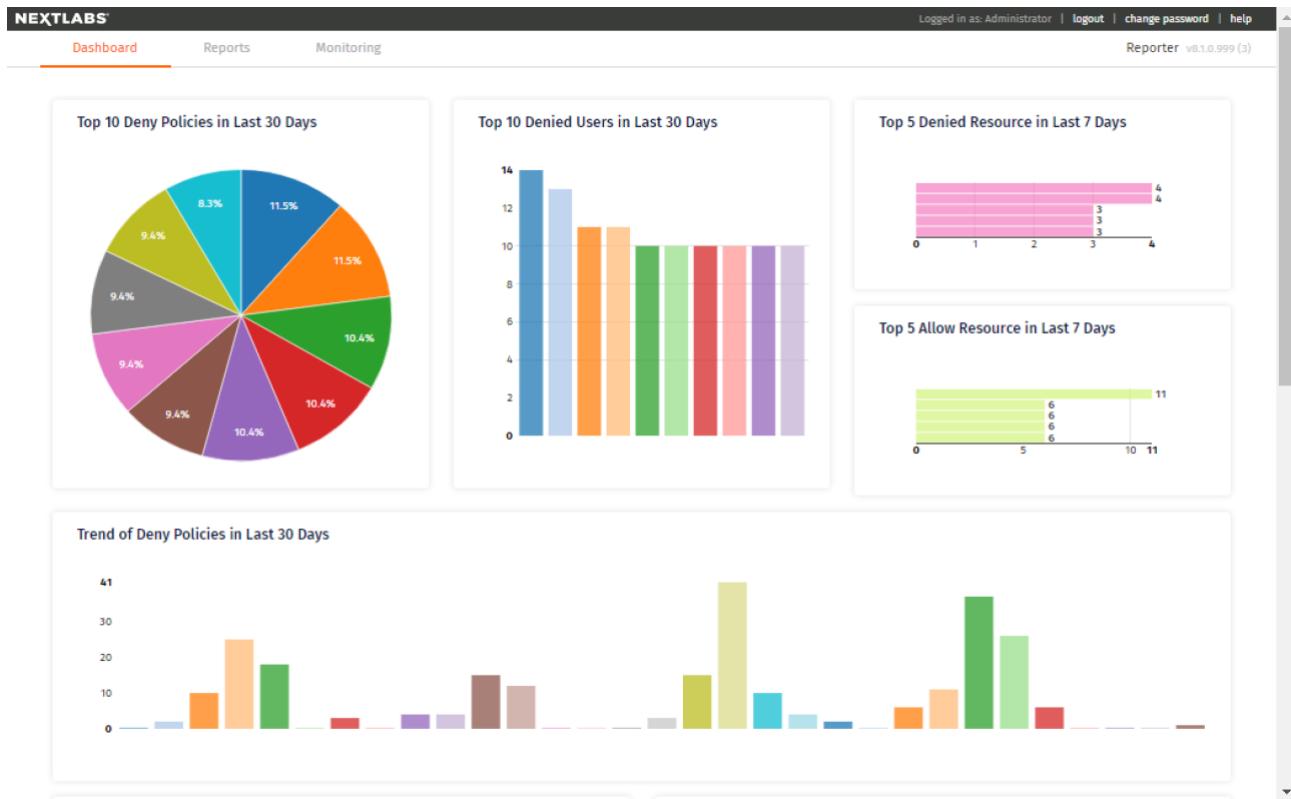


Figure 56: Dashboard tab

About the Reporter console

The Reporter console has three tabs: Dashboard, Reports, and Monitoring.

By default, the **Dashboard** tab appears when Reporter opens. Only users granted access to monitors can access the **Monitoring** tab.

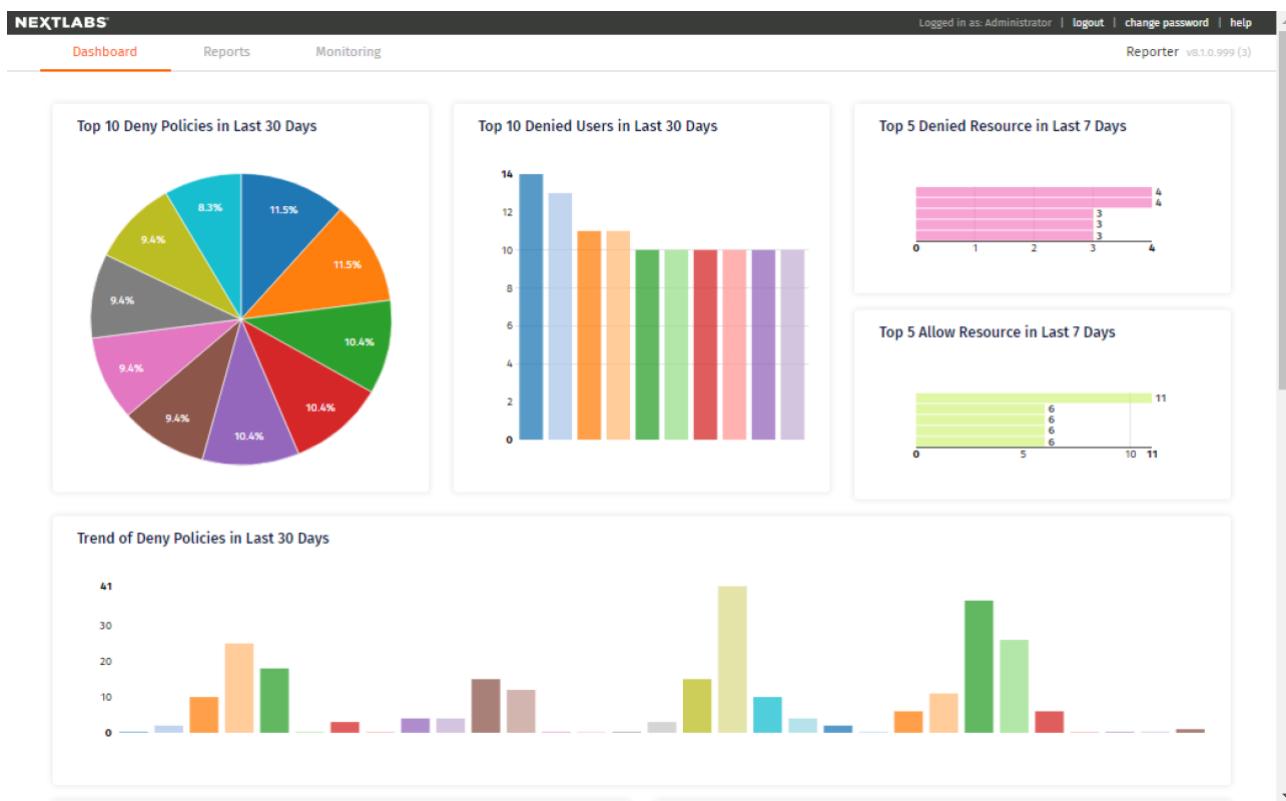


Figure 57: Dashboard tab

Dashboard tab

The **Dashboard** displays charts and tables representing a set of predefined queries that are updated automatically with fresh data from the Activity Journal.

Reports tab

The **Reports** tab is where you define and run reports, save reports for future use and, optionally, share reports with other users.

Related concepts

[Working with reports](#) on page 215

This section describes how to use the **Reports** tab to define, save, and run reports about data or resource access and use, and policy enforcement, throughout your enterprise.

Monitoring tab

The tab is where you define monitors to receive notifications (alerts) about specific policy enforcements, and view alerts triggered by the monitors. Only users granted access to monitors can access the **Monitoring** tab.

Related concepts

[Working with monitors and alerts](#) on page 236

This section describes how to use the **Monitoring** tab to create policy monitors to obtain notifications of policy enforcement activities that meet predefined criteria.

Banner

The banner at the top of the web page displays version information and links.

Logout

The **logout** link enables you to log out of the Reporter console.

You can also log out by closing your web browser, or closing the active tab in your web browser.

Help

The Help link provides information to help you use Reporter.

Using the Reporter Dashboard

This section describes the features of the Reporter Dashboard, the tab that displays when you access Reporter.

You access Reporter by clicking **Reports** on the left navigation bar of the Control Center console.

Introducing the Reporter Dashboard

The Reporter Dashboard is divided into panes, each displaying a predefined statistical view of data that provide a snapshot of policy enforcement trends.

In the default configuration of Reporter, these panes display data on:

- Most commonly Denied requests for access to data resources, by resource, user, and policy
- Most commonly Allowed request for access to resources, by resource
- 30-day trend for the enforcement of all Deny policies, for all users and resources
- Detailed information about the ten most recent instances of any policy either Allowing or Denying a user access to any resource
- Statistics on alerts in the current week
- Detailed information about alerts raised in the current day

What you see may be different if the application administrator changed the data that appears in the Dashboard or its layout.

The data displayed in all panes of the dashboard is refreshed from the Activity Journal each time you open the **Dashboard** tab. This means that data is updated on demand; for example, if a pane shows some statistic for the past week, that reflects not the last seven whole calendar days, but the last seven 24-hour periods starting from the top of the current hour.

The following figure shows a sample dashboard. The charts have interactive features. When you place the mouse pointer over a bar in a bar chart or over a slice in the pie chart, a tooltip displays information about that value series.



Figure 58: The Dashboard tab and summary information

About Reporter Dashboard data

The following table summarizes the data presented in each of the predefined queries represented on the Dashboard, and discusses the possible implications of and uses of the data displayed in each.

Table 48: Reporter Dashboard data

Graph	Description	May Indicate:
Top Deny Policies (30 days)	Pie chart representing the Deny policies that were most frequently enforced over the previous thirty days.	<ul style="list-style-type: none"> Misunderstanding of access level: users being blocked from a resource they believe they should use Incorrectly defined entitlements: users should have access, but policies are not updated or correctly designed

Graph	Description	May Indicate:
Top Denied Users (30 days)	Bar chart representing the users who have had the most instances of any Deny policy enforced against them.	<ul style="list-style-type: none"> Users who habitually snoop into resources they are not authorized to use Incorrectly defined entitlements: users or group should have access, but policies are not updated or are incorrectly designed
Top Denied Resources (7 days)	Bar chart representing the resources that any users have most frequently attempted to access and been blocked by an active policy, over the previous seven days.	<ul style="list-style-type: none"> Resources of broad interest to users who should not be using them Incorrectly designed resource or user component, blocking users who should have access
Top Allow Resources (7 days)	Bar chart representing the resources that users have most frequently attempted to access and been allowed by an active policy, over the previous seven days.	<ul style="list-style-type: none"> Improperly designed resource component or policies, which allow inappropriate users access to sensitive resources
Deny Policy Enforcement Trends (30 days)	Bar chart representing the trend, over the previous 30 days, of the daily total instances of any deny policy being enforced on any user, for any resource.	<ul style="list-style-type: none"> Progress (or lack of it) in educating users about access policies and individual/group entitlements, at a broad level Improperly designed policies that are blocking too many users who expect and are entitled to access or use
Last 10 Allow Enforcement in Last 7 Days	<p>List of details about the most recent instances of any allow policy being enforced against any user, for any resource. Details listed include:</p> <ul style="list-style-type: none"> time: Date and time of enforcement username: The name of the user who triggered the policy Resource Id: The resource the user was targeting policy_decision: The authorization decision action: The action that triggered the policy 	<ul style="list-style-type: none"> Instances where some urgent action is required, such as users being allowed access to some resource they should not be using, due to lack of policy coverage or an incorrectly defined policy

Graph	Description	May Indicate:
Last 10 Deny Enforcement in Last 7 Days	List of details about the most recent instances of any deny policy being enforced against any user, for any resource. Details listed include: <ul style="list-style-type: none"> • time: Date and time of enforcement • username: The name of the user who triggered the policy • Resource Id: The resource the user was targeting • policy_decision: The authorization decision • action: The action that triggered the policy 	<ul style="list-style-type: none"> • Instances where many users are attempting to get at data they are not authorized to use • Instances where some urgent correction is required to allow appropriate access, such as multiple authorized users being blocked from some resource they need by an incorrectly defined policy
Alerts this Week: Group by Tags	Treemap representing volume of alerts in the current week. Alerts are grouped by monitor tags.	<ul style="list-style-type: none"> • Policies being watched by monitors that are tagged, are being enforced at a rate that demands attention. Further review or action may be required.
Today's Alerts: Details	List of details about the alerts raised in the current day. Details include: <ul style="list-style-type: none"> • Level • Monitor name • Alert message • Date and time the alert was raised 	<ul style="list-style-type: none"> • Policies being monitored are being enforced at a rate that demands attention. Further review or action may be required.

Working with reports

This section describes how to use the **Reports** tab to define, save, and run reports about data or resource access and use, and policy enforcement, throughout your enterprise.

Related concepts

[Reports tab](#) on page 211

The **Reports** tab is where you define and run reports, save reports for future use and, optionally, share reports with other users.

Reporting functionality and permissions

You define and run reports in the Reports console. The functionality available to you depends on the rules established in the Delegated Administration section of the console. The **Reports** tab has two panes. The pane on the left displays the Saved Reports—the list of all saved reports available to you. This includes all reports you create and save (if you have permission to create reports), all reports saved by other users and shared with you, and the sample reports used to generate data that is displayed in the **Dashboard** tab.

When you click on any item in **Saved Reports**, the details of that report display in Report Details, on the right. This is also where you work when you create a report. In Report Details, you define the following:

- The time period of the policy activity data to cover in the report
- The criteria, or filters, that identify the policy activity data to include in the report
- The output format of the report

The default settings in Report Details appear when you click the **Reports** tab or when you click **New** in the Saved Reports pane. By default, the time period of the report is the current day, all policy activity data at the user level is included, and the data is presented in table format.

After defining a new report or editing an existing report, click **Run** at the bottom of the Report Details pane to view results as shown in the following figure.

Figure 59: The Reports tab

The following figure shows an example of a generated report. In the example, the output format is a bar chart. The chart appears at the bottom of the Report Details pane. Depending on the output format you select in Report Type and the browser you are using, the generated report appears either in the Report Details pane or in a separate browser tab.



Figure 60: Report Results (bar chart) at the bottom of Report Details

Related concepts

[About delegation policies](#) on page 21

Creating a report

A report is a formatted view of a particular set of information extracted from the Activity Journal, a set of database tables where policy activity data is logged.

You create a report by defining a query—criteria, or filters, for the policy activity data you want to include in the report—and specifying the display options that determine how the data is presented.

Just as policies can be designed to control information access and usage by types of users, actions, resources, applications, and computers, you can design report queries that filter policy activity data by attributes of those components. The greater the number of filters, the fewer the number of records returned. For example, compare the following queries:

- Retrieve data for deny enforcements of any policy (in other words, all policies), covering any action by any user on any resource, for one week.
- Retrieve data for deny enforcements by communication policies that cover only the email action on only documents classified as Confidential, for one week.

The first query specifies fewer conditions and retrieves more data than the second.

If you don't specify any filters or change any of the default settings, the report retrieves all policy activity data categorized as user-level events, for the current day.

1. Access the Reporter console: In the Control Center console, click **Reports** on the left navigation bar.
2. Click the **Reports** tab, if it is not already open.
3. If the query of an existing report is currently displayed in the Report Details pane, click **New** at the bottom of the Saved Reports pane.
4. Define the report query and display options. Many of the fields are optional. Required fields are populated with default values.

Table 49: Report query and display fields

Query field	Description
From and To	The start date and time, and end date and time, respectively, of the time period this report covers. Click in the field to choose a date and time from the calendar. When specifying a report period, you need to consider the time zone of the system, and the time period of data stored in the Activity Journal.
Event Level	<p>The level of event verbosity the report contains:</p> <ul style="list-style-type: none"> • User Events (default) • Application Events (application and user-level events) • All System Events (system, application, and user-level events) <p>As a rule, you should leave this setting at User Events. This setting significantly reduces the amount of system noise. Application- or system-level events are generally not useful in monitoring policy or user activities.</p>
Policy Decision	<p>The type of enforcement effect to include in this report:</p> <p>Allow: Instances when the policy permitted the user to perform the action covered by the policy. Report results are determined by the information that is logged. For example, if the policy contains no On Allow logging obligations, the report returns no On Allow data regardless of whether you select this option.</p> <p>Deny: Instances when the policy did not allow the user to perform the action. Deny decisions are always logged.</p> <p>Both: All instances when the policy was enforced, with either Allow or Deny effect.</p>
Resource Type	The resource type to include in the report. Resource types are the templates that include information about attributes, actions, and obligations available to components and policies.
Action	<p>The user action or actions to include in this report. The list shows all currently defined actions for the selected resource type. To select multiple actions, press Ctrl and click each action. If no actions are selected, all actions are included in the report.</p> <p>Policies involving Paste actions do not support logging obligations, therefore, instances of their enforcement are not included in reports.</p>
User Criteria	Additional user criteria by creating one or more conditions. Each condition consists of a user attribute, an operator, and a value. You must click the +button to add a condition to the query.

Query field	Description
Resource Criteria	Additional resource criteria specified using conditions. Each condition consists of a resource attribute, an operator, and a value. You must click the + button to add a condition to the query.
Policy Full Name	One or more policies on which to filter, or leave this field blank to include all policies. Use the Policy Lookup window to browse through and select the policies.
Policy Criteria	Additional policy criteria specified using conditions. Each condition consists of a policy attribute, an operator, and a value. You must click the + button to add a condition to the query.
Other Criteria	Additional criteria specified using conditions. Each condition consists of a general attribute (for example, hostname, host IP, and application name), an operator, and a value. You must click the + button to add a condition to the query.
Report Type	The output format in which to display the data: Table, Bar Chart, Horizontal Bar Chart, or Pie Chart. Use a table to display policy activity details in a row-and-column format. Use a chart to display a summary of policy activities.
Show	The display options for the report type (not available for Table type reports).
Sort By	<p>The field on which to sort the data, and select Asc to sort in ascending order or Desc to sort in descending order. If the report type is a table, you can sort the data by any attribute. If the report type is a chart, you can sort either by the grouping item (user, resource, policy, month, or day) or by Result Count (the number of enforcement events for each user, resource, policy, month, or day).</p> <p>For chart report types, select grouping options (grouping is not available to table type reports):</p> <ul style="list-style-type: none"> • Group by User: The chart shows the number of enforcement events for each user covered by the report. • Group by Resource: The chart shows the number of enforcement events for each resource covered by the report. • Group by Policy: The chart shows the number of enforcement events for each policy covered by the report. • Group by Month: The chart shows the number of enforcement events for each month covered by the report. Select this option only if the time period you specified spans more than one month. • Group by Day: The chart shows the number of enforcement events for each day covered by the report.
Max Results	The maximum number of results to display in the table or chart. For charts, this number represents the maximum number of bars in a bar chart, or slices in a pie chart. For readability reasons, charts should display a limited number of bars or slices. For a table, the number represents the maximum number of rows (each row represents an event). Tables that show a large number of rows present the data on multiple pages.

Query field	Description
Display Columns	<p>The columns to display in a table. This setting applies to a table only. USER_NAME, POLICY_FULLNAME, POLICY_DECISION, HOST_NAME, and APPLICATION_NAME are selected by default. To remove any of those columns or to add other columns, click  and use the arrow icons to move columns out of, or into, the selected pane.</p>

5. Click **Run** to generate the report.
6. If you want to run report at a future time, save it by clicking **Options > Save**.

Related concepts

[Specifying a report period](#) on page 220

When specifying the start date and time and end date and time of the time period to cover in a report, you need to take into account the time zone where the system is installed, and how long data is stored in the Activity Journal.

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Related tasks

[Filtering by criteria](#) on page 222

To filter by user, resource, or policy criteria, use conditions in the User Criteria, Resource Criteria, or Policy Criteria fields.

[Saving reports](#) on page 228

Specifying a report period

When specifying the start date and time and end date and time of the time period to cover in a report, you need to take into account the time zone where the system is installed, and how long data is stored in the Activity Journal.

Related tasks

[Creating a report](#) on page 217

A report is a formatted view of a particular set of information extracted from the Activity Journal, a set of database tables where policy activity data is logged.

Time zone considerations

The report period is determined using the time zone where the system is installed, not the time zone where you are running Reporter console or the time zone where the events you are querying on occurred.

Therefore, you might want to add hours to the beginning or end of your time period, depending on the time difference between the location of the Reporter console application and the location where the events of interest occurred.

To generate reports for specified periods, you might want to add hours to the beginning or end of your time period, depending on the time difference between the location of the Reporter console application and the location where the events of interest occurred.

For example, suppose your system is installed in New York, but you are running Reporter in California, three time zones to the west, and your query is on events in California. If you specify date-time values that start and end at midnight, your query returns three hours of data from the night before the first date in your time period (in California time), and does not include the last three hours of data on the final day of the time period.

Activity Journal archive considerations

By default, the Activity Journal stores data for 90 days before archiving it, but this setting can be changed by the database administrator.

Depending on the archive setting, data for the period you specify might be unavailable. If a query uses all default values, except for the report period, and it returns no data, try changing the From and To values to later dates.

To create a report that uses data that is already archived, ask your database administrator to restore the data to the Activity Journal.

Related concepts

[Database management operations](#) on page 125

As the Report Server gathers data on user activity and policy enforcement, it temporarily stores it in OLTP tables before writing it to the Activity Journal database, where it is available for generating reports.

[Running a report](#) on page 225

Filtering by user resource, and policy

When filtering policy activity data by user, resource, or policy, you can use either, or both, of the following methods.

- Filter by one or more usernames, resource paths, or policy names
- Filter by user, resource, or policy criteria

Related tasks

[Creating a monitor](#) on page 237

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

Filtering by name

To filter by one or more usernames, resource paths, or policy names, you specify values in the User, Resource Path, and Policy Name fields.

For User and Policy Name, use the Lookup tool to browse for and select the names. The following figure shows the User Lookup window. The Policy Lookup tool is similar.

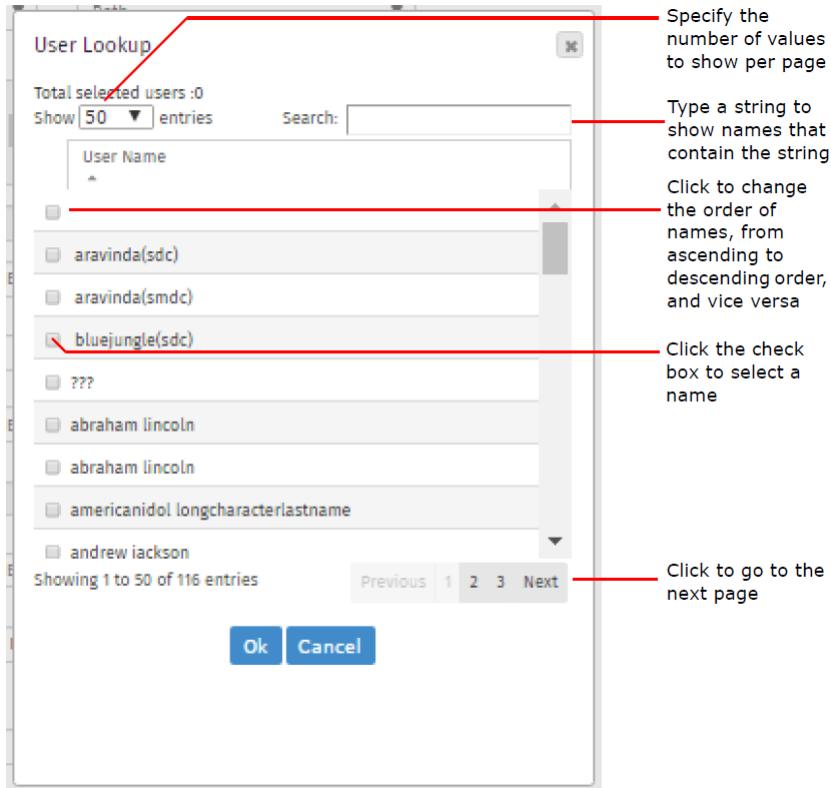


Figure 61: User Lookup window

There is no lookup tool for Resource Path. You must type the resource path value. Make sure you enter the correct information. If you specify a path that does not exist, the filter essentially has no effect. Observe the following guidelines when typing values:

- For multiple values, separate each value with a comma.
- Include the full folder path in each value.
- Use upper- or lowercase characters as needed. Queries are not case sensitive.
- Use an asterisk (*) as a wildcard character. For example, you can type the following string to specify all content—subfolders and files—in the Finance folder:

c:\Finance*

Filtering by criteria

To filter by user, resource, or policy criteria, use conditions in the User Criteria, Resource Criteria, or Policy Criteria fields.

Each condition consists of an attribute, an operator, and a value. You can create multiple conditions for each filter type. For example, you can filter resources by the Modified By and Keywords attributes to find documents modified by a particular author and that contain a specified keyword. Both conditions must be true for the filter to return any results.

You can create a condition for User Criteria, Resource Criteria, or Policy Criteria.

1. Access the Reporter console: In the Control Center console, click **Reports** on the left navigation bar.
2. Click the **Reports** tab, if it is not already open.
3. On the Report Query form, select an attribute from the Resource Name drop-down list. The list displays the attributes that have been definedThe policy attributes list also includes the tags, if any, defined for policies.

The following figure shows an example of a list of resource attributes.

The screenshot shows a search interface for resource attributes. On the left, there are several dropdown menus and input fields:

- Resource Criteria:** A dropdown menu showing "FROM_RESOURCE_NAME" selected. To its right is a dropdown for "Comparison Operator" set to "Equals", a text input field, and a "+" button.
- Policy Name:** A dropdown menu showing "FROM_RESOURCE_NAME" selected.
- Policy Criteria:** A dropdown menu showing "Revision" selected.
- Other Criteria:** A dropdown menu showing "Resource Signature" selected.
- Report Type:** A dropdown menu showing "Table" selected.
- Sort By:** A dropdown menu showing "user_name" selected.

Figure 62: Resource attributes list

4. Select a comparison operator from the second drop-down list.
 - Equals: Attribute value matches a specified value.
 - Not Equals: Attribute value does not match a specified value. Use this operator to exclude the specified value from the result set.
 - Like: Attribute value is similar to the specified characters. This operator is similar to the SQL LIKE keyword. Use it to match a specific pattern, or when you do not know the entire value.
 - In: Attribute value matches any one of a specified set of values. This operator is similar to the SQL IN keyword. Use it to create an OR condition, that is, when you want to match multiple values.
- For all the operators, searches are case-insensitive. For example, Patrick Marleau is the same as patrick marleau.
5. Type a value in the third field.
 - If you selected the Equals operator, the specified value must be an exact match (aside from case); otherwise, the filter does not find any matches and the report does not return any results.
 - If you selected the Not Equals operator, the specified value you do not want to match must also be an exact match to an attribute value (aside from case); otherwise, the filter does not exclude any values, and you get more results than you expect.
 - If you selected the Like operator, you can use the following wildcard characters:
 - %: matches one or more characters.
 - _: matches any single character. You can use more than one underscore consecutively; for example, to match any two consecutive characters, use two underscores.
 - If you selected the In operator to specify a set of values, separate each value with a comma.
6. Click the + button to add the condition to the report query.
7. Repeat the previous steps to add conditions as needed. The following figure shows an example of a query where two conditions have been created for Resource Criteria. To remove a condition, click the [-] button.

The screenshot shows the "Resource Criteria" section with two conditions listed:

- 1. "Modified By" followed by a dropdown menu showing "Modified By" selected, an "Equals" dropdown menu, a text input field containing "Max 255 characters", and a "+" button.
- 2. "Keywords like confidential" followed by a dropdown menu showing "Keywords like confidential" selected, a "-" button, and another dropdown menu showing "Modified By == John Daly" selected, with its own "-" button.

Figure 63: Two conditions in Resource Criteria

Related concepts

[Filter examples](#) on page 224

Related tasks

[Creating a report](#) on page 217

A report is a formatted view of a particular set of information extracted from the Activity Journal, a set of database tables where policy activity data is logged.

Filter examples

These condition searches for resources that contain the keyword Confidential (the search is not case sensitive).

Keywords Equals Confidential

The following condition searches for users who are not US citizens:

ISO Country Code Not Equals US

The following condition searches for users whose email addresses end with acme.com:

Email Address Like %acme.com

The following condition searches for machine names that start with NL, followed by any two characters (represented by two underscores), the letters CC, and other characters, for example, NL03CC-Win64:

HOST_NAME Like NL__CC%

The following condition searches for users who are in the Accounting or Finance departments:

Department In Accounting, Finance

Related tasks

[Filtering by criteria](#) on page 222

To filter by user, resource, or policy criteria, use conditions in the User Criteria, Resource Criteria, or Policy Criteria fields.

Filtering by action

You can filter the policy activity data by user actions, such as create, run, or rename.

The Action list displays all the actions that have been defined. You can select more than one action from the list, for example, for a report on all instances of users moving, renaming, or deleting certain files and those actions triggering policy enforcement.

The following table describes the user actions on which you can filter. The actions in the list differ, depending on the policy enforcement software installed in your environment.

Table 50: Action Filters

Action	Instances included in the report
Ask Question	Users submitting a question during a Web meeting, via Live Meeting
Attach Device	Users attaching removable devices to a computer
Attach to Item	Users attaching a file to any kind of portal content, such as a list or a library; or uploading a file to a library or list
Change Attributes	Users changing attributes of a file
Change File Permissions	Users changing any of the security properties of any file
Copy/Embed File	Users copying any file, or contents of any file or inserting one file into another

Action	Instances included in the report
Create/Edit	Users either editing an existing file, or creating a new file in any location covered by the policy
Delete	Users deleting any file
Download	User downloading a file from a portal
E-Mail	Users sending e-mail to any recipient, with or without an attachment (as dictated by the policy)
Export	Users export a spreadsheet or datasheet from a collaboration portal.
FTP	Users FTP files (FTP, SFTP, or FTPS) from one host to another
Host Connect	User connecting to a specified port, as designated in policy; not specific to the resource or document being accessed
Instant Message	Users sending instant message to any recipient, with or without an attachment (as dictated by the policy)
Join	Users joining a established Web meeting via Live Meeting
Meet	Users setting up a meeting via Office Communicator with any user
Move	Users moving any file from its current location to any other location
Network Access	User connecting to a specified port, as designated in policy; specific to a resource or document that is also defined in the policy
Open	Users opening any file or portal content item
Print	Users printing any file or portal content item
Record	Users using Live Meeting's Record feature to record a Web meeting
Rename	Users changing the name of any file or portal content item
Run	Users running an application covered by the policy
Share	Users sharing their desktop, via Live Meeting, during a Web meeting
Screen Capture	Users taking screen captures of content on their desktop
Upload	Users uploading content to a portal
Video	Users initiating a video call via Live Meeting
Voice	Users initiating a voice call via Live Meeting
Voice Call / Video Call	Users initiating a voice or video call via Office Communicator, with any user covered by the policy

Related tasks

[Creating a monitor](#) on page 237

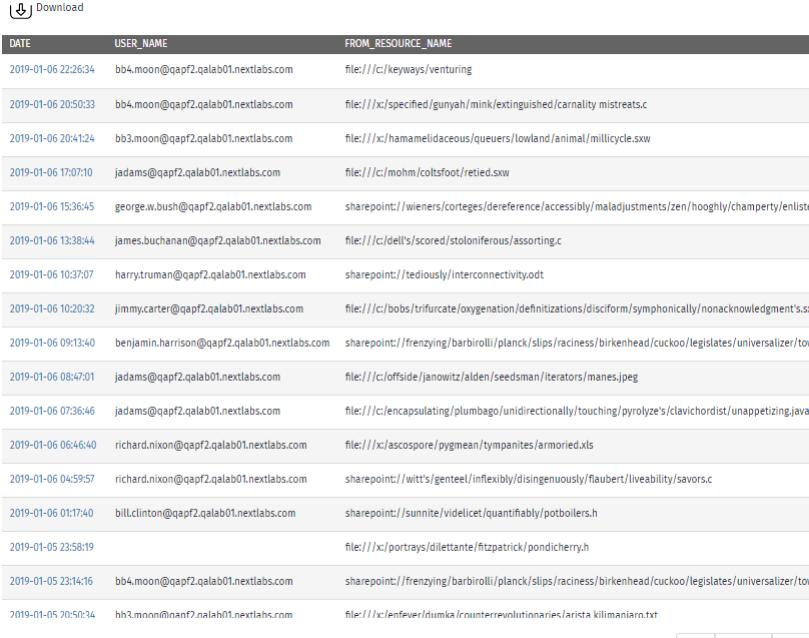
Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

Running a report

You can run any report selected from Saved Reports, or a new report you define in Report Details. Simply click the Run button at the bottom of Report Details.

When you run a report, Reporter compares the information currently in the Activity Journal to the report criteria, selects the matching information (if any), calculates the totals, and displays the results in the format you selected.

If you selected a chart as the output format, the chart appears at the bottom of the Report Details pane. Scroll down to view the chart. If the output format is a table, it might appear in a separate tab in the browser window, depending on the browser you are using. The following figure shows an example of policy activity data displayed in a table.



The screenshot shows a table with columns: DATE, USER_NAME, FROM_RESOURCE_NAME, POLICY_FULLNAME, and POLICY. The table contains 20 rows of data. At the bottom, there are navigation buttons: First, Previous, and Next.

DATE	USER_NAME	FROM_RESOURCE_NAME	POLICY_FULLNAME	POLICY
2019-01-06 22:26:34	bb4.moon@qapf2.qalab01.nextlabs.com	file:///c:/keyways/venturing	policy indispensably 4	Allowed
2019-01-06 20:50:33	bb4.moon@qapf2.qalab01.nextlabs.com	file:///c:/specified/gunyah/mink/extinguished/carnality mistreats.c	policy pyrope 1	Allowed
2019-01-06 20:41:24	bb3.moon@qapf1.qalab01.nextlabs.com	file:///c:/hamanetidaceous/queuers/lowland/animal/millicycle.sxw	policy pinkroot 5	Allowed
2019-01-06 17:07:10	jadams@qapf2.qalab01.nextlabs.com	file:///c:/mohm/coltsfoot/reited.sxw	pf-subject-01	Denied
2019-01-06 15:36:45	george.w.bush@qapf2.qalab01.nextlabs.com	sharepoint://wiener/corteges/dereference/accessible/maladjustments/zen/hooghly/champerty/enlister/sodalist	policy counterfeiter boardbound 9	Allowed
2019-01-06 13:38:44	james.buchanan@qapf2.qalab01.nextlabs.com	file:///c:/dell/s/scored/stoloniferous/assorting.c	policy edema 30	Denied
2019-01-06 10:37:07	harry.truman@qapf2.qalab01.nextlabs.com	sharepoint://tediously/interconnectivity.odt	policy gaspee's blaspheme 3	Allowed
2019-01-06 10:20:32	jimmy.carter@qapf2.qalab01.nextlabs.com	file:///c:/bobs/trifurcate/oxygenation/definitizations/disciform/symphonically/nonacknowledgment.sxw	policy spats 19	Allowed
2019-01-06 09:13:40	benjamin.harrison@qapf2.qalab01.nextlabs.com	sharepoint://frenzying/barbirolli/plancn/slips/raciness/birkenhead/cuckoo/legislates/universalizer/towage/clamming.java	policy engulf 26	Allowed
2019-01-06 08:47:01	jadams@qapf2.qalab01.nextlabs.com	file:///c:/offside/janowitz/alden/seedsman/iterators/manes.jpeg	policy greater mermaids 11	Allowed
2019-01-06 07:36:46	jadams@qapf2.qalab01.nextlabs.com	file:///c:/encapsulating/plumbago/unidirectionally/touching/pyrolyze's/clavichordist/unappetizing.java	policy aftershave polysemous 16	Allowed
2019-01-06 06:46:40	richard.nixon@qapf2.qalab01.nextlabs.com	file:///c:/ascopore/pygmean/tympanites/armored.xls	policy engulf 26	Denied
2019-01-06 04:59:57	richard.nixon@qapf2.qalab01.nextlabs.com	sharepoint://witt's/gentle/inflexibly/disingenuously/flauber/liveability/savors.c	multivalue policy	Allowed
2019-01-06 01:17:40	bill.clinton@qapf2.qalab01.nextlabs.com	sharepoint://sunnite/videlicet/quantifiably/potboilers.h	policy slinkly spiranolactone 18	Allowed
2019-01-05 23:58:19		file:///c:/portrays/dilettante/ftzpatrick/pondicherry.h	policy stenographer 8	Allowed
2019-01-05 23:14:16	bb4.moon@qapf2.qalab01.nextlabs.com	sharepoint://frenzying/barbirolli/plancn/slips/raciness/birkenhead/cuckoo/legislates/universalizer/towage/clamming.java	policy enuresis 7	Allowed
2019-01-05 20:50:34	hh2.moon@nanf2.qalab01.nextlabs.com	file:///c:/enfeaver/dumka/counterrevolutionaries/arista/kilimanjaro.txt	policy aftershave noxiousmous 16	Denied

Figure 64: Report results in table format

If no data matches the specified criteria, a message appears, as shown in the following figure.



Figure 65: No data message

Return to the report query to check the specified criteria, and modify the query as needed. Check the time period. As discussed in the following figure, if the specified time period is too far in the past, the data is likely to have been archived. Remember, too—the greater the number of filters, the narrower the scope of data returned. If there are conditions specified in multiple fields, such as User Criteria, Policy Criteria, and Resource Criteria, all the conditions must be true for the data to match.

Conversely, if the report returns more data than you expect, check the filters in the query. For example, verify that the filters have been added to the query. The following figure shows an example of a resource filter that has been defined (values are entered in the Resource Criteria fields), but that has not been added to the query. You must click the + button to add a filter. When a filter has been added, it appears below the field, as indicated by the policy filter that appears below Policy Criteria.

Resource Criteria:	Keywords	Equals	ITAR	+	Click to add the filter
Policy Full Name:	<input type="text"/>				
Policy Criteria:	POLICY_NAME	Like	Max 255 characters	+	
POLICY_NAME like SAP [-] , Filter added					

Figure 66: Filter examples, one added and one not

Related concepts

[Activity Journal archive considerations](#) on page 220

By default, the Activity Journal stores data for 90 days before archiving it, but this setting can be changed by the database administrator.

Viewing report output

When viewing the report output, you can drill down for additional information.

A table shows all the policy enforcement events that meet the criteria specified in the report query. You can click the date value in any row to get details for an event. The details appear in a Log Details Report on a separate page, as shown in the following figure.

Report Detail		Export report to:	POLICY_DECISION
DATE	USER_NAME		
2019-01-03 15:20:37	chris.webber@hdesk.com		
2019-01-03 15:20:32	chris.webber@hdesk.com		
2019-01-03 15:20:28	chris.webber@hdesk.com		
2019-01-03 14:22:00	chris.webber@hdesk.com		
2019-01-02 13:44:41	S-1-5-21-4674712		
2019-01-02 13:44:41	S-1-5-21-4674712		
2019-01-02 12:22:39	S-1-5-21-4674712		
2019-01-02 12:22:39	S-1-5-21-4674712		
2019-01-02 11:49:16	S-1-5-21-4674712		
2019-01-02 11:49:16	S-1-5-21-4674712		
2019-01-02 11:49:15	S-1-5-21-4674712		
2019-01-02 11:49:15	S-1-5-21-4674712		
2019-01-02 11:47:41	S-1-5-21-4674712		
2019-01-02 11:47:41	S-1-5-21-4674712		
Event Detail			
Date:	2019-01-03 15:20:37		
Policy:	Deny access to Security Vulnerabilities if not Created By or Assigned To the User		
User:	chris.webber@hdesk.com		
Application:	Unknown		
From Resource:	ticket:2206		
To Resource:			
Decision:	Deny		
Action:	View		
Host:	w2k12pc1		
Host IP:	10.23.57.207		
Event Level:	Event Level 3		
		<input type="button" value="Close"/>	

Figure 67: Selecting a row to obtain event details

Unlike a table, a chart does not show each policy enforcement event. Rather, a chart aggregates policy enforcement events by the grouping option you specify. For example, a chart can group events by user. In this case, each slice of a pie chart or bar in a bar chart represents a user and the total number of events triggered by the user. You can click a chart slice or bar to see all the events triggered by a particular user.

The following figure shows the list of events that appear when a bar in a bar chart is clicked. The number, 5.00, on top of the bar indicates the number of events for a particular user, and,

correspondingly, the table lists the five events. Drill down for details about each event by clicking a date value in a row in the table.

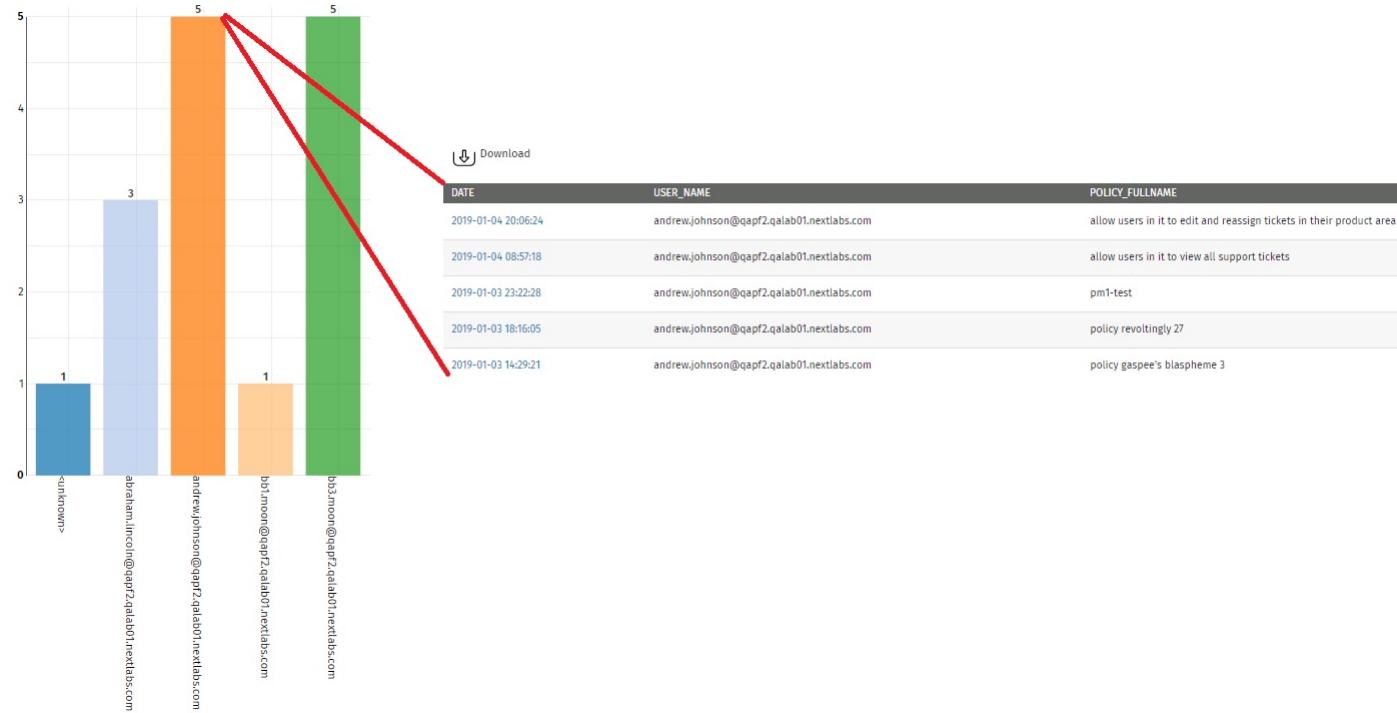


Figure 68: Clicking a bar in a chart to obtain a list of events

Non-Latin characters

If you are using Microsoft SQL and your database includes columns with content in languages that use character sets other than standard Latin, reports based on them might not display properly. For example, they might show question marks instead of the proper characters.

If you encounter this problem in reports, you can solve it by having your database administrator change the collation value for the columns in question, to match the language of the content of that column. Note that only non case-specific collations should be used, whatever the character set. Once you do this, the reports should generate properly.

This should be done on a column-by-column basis, and applies only to Microsoft SQL databases.

Saving reports

When you create a report, it is not saved automatically. Similarly, if you change any settings in a report you selected from Saved Reports, the changes are not saved until you explicitly save the report. This behavior is convenient for creating and running reports on an ad hoc basis. However, to run a report regularly—for example, for a weekly status report on policy activity related to certain export-controlled documents stored on specific servers—you can save the report.

Another reason to save a report is to share it with other users. The actions that users can take with a shared report depend on their permissions. Administrators can run the report with different criteria, save a new version of the report and share it with others. Non-administrative users can only run and view the report. They cannot modify the report criteria, but they can select a different report format, for example, a different chart type.

You can save a new and modified reports as needed.

1. At the bottom of the Report Details pane, click the **Options** button, then select **Save**. If you are saving changes made to a previously-saved report, you can select **Save As** to save the report as a

new report. The **Save As** command is your only option if you are saving a report that another user created.

2. In the **Save As** dialog, specify the following information:

- In **Name**, type a name for the report.
- In **Description**, provide a clear description of the report, particularly if you plan to share the report with other users.
- In **Date mode**, select one of the following values:
 - **Fixed Dates**: The report uses the dates currently specified in the **From** and **To** fields.
 - **Relative Dates**: The report uses dates relative to the date the report is run. This is the typical option for running a report on a regular schedule, for example, weekly, monthly, or quarterly.

3. If you selected **Relative Dates**, select one of the following values in **Date Selection**:

Value	Description
Today	The report searches for data for the current day.
Current Week	The report searches for data in the week in which the report is run. For example, if the report is run on 2014-09-15, it searches for data from 2014-09-14 (Sunday) to 2014-09-20 (Saturday).
Current Month	The report searches for data in the month in which the report is run. For example, if the report is run on 2014-09-15, it searches for data from 2014-09-01 to 2014-09-30.
Current Quarter	The report searches for data in the quarter in which the report is run. For example, if the report is run on 2014-09-15, it searches for data from 2014-07-01 to 2014-09-30.
Yesterday	The report searches for data from the previous day.
Last Calendar Week	The report searches for data in the previous week. For example, if the report is run on 2014-09-15, it searches for data from 2014-09-07 to 2014-09-13.
Last Calendar Month	The report searches for data in the previous month. For example, if the report is run on 2014-09-15, it searches for data from 2014-08-01 to 2014-08-31.
Last Calendar Quarter	The report searches for data in the previous quarter. For example, if the report is run on 2014-09-15, it searches for data from 2014-04-01 to 2014-06-30.
Last 7 Days	The report searches for data from the last 7 days.
Last 30 days	The report searches for data from the last 30 days.

4. In **Share with**, select one of the following options:

- **Only me**: The report is private. No one else can see the report.
- **Public**: The report is available to all users who have access to Reporter.
- **Users/Groups**: The report is available to selected users or user groups.

5. If you selected **Users/Groups**, specify each user or group of users with whom to share the report:

- a. In the text field that appears, type the first letter of the user's first name or the group name. A list displays all the user and group names that begin with the letter you typed as shown in the following figure.
- b. Select a name.
- c. Repeat the previous steps to add names as needed.

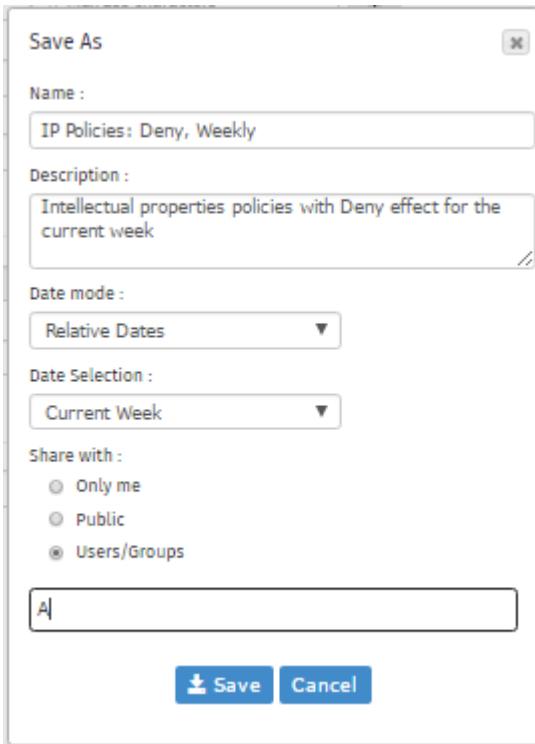


Figure 69: Selecting users with whom to share a report

6. Select Save. The report appears in the Saved Reports pane.

Related tasks

[Creating a report](#) on page 217

A report is a formatted view of a particular set of information extracted from the Activity Journal, a set of database tables where policy activity data is logged.

Sample reports

This section presents examples of the available report formats, all representing a simple set of event data returned by the same report query.

The query requests all events by all users on all resources. By comparing the examples carefully, you can get a better understanding of the way the different formats can be used to highlight different aspects of the same data.

Table

The following figure shows a small set of policy activity data displayed in a table.

[Download](#)

DATE	USER_NAME	FROM_RESOURCE_NAME	POLICY_FULLNAME	POLICY_DECISION
2019-01-03 15:20:37	chris.webber@hdesk.com	ticket:2206	deny access to security vulnerabilities if not created by or assigned to the user	Denied
2019-01-03 15:20:32	chris.webber@hdesk.com	ticket:1103	allow users in it to edit and reassign tickets in their product area	Allowed
2019-01-03 15:20:28	chris.webber@hdesk.com	ticket:1103	allow users in it to view all support tickets	Allowed
2019-01-03 14:22:00	chris.webber@hdesk.com	ticket:1103	allow users in it to view all support tickets	Allowed

Figure 70: Data presented in a table

This report reflects the following:

- Four policy enforcement events were logged—that is why there are four rows in the table.
- As the Date column shows, all four events occurred on the same day, August 7, 2014.

- Each event was triggered by a different user, as shown by the four different values in the User column.
- Each event represents a different policy, as shown by the four different values in the Policy column.
- Only three resources were involved. The first two rows show enforcement of two different policies by two different users, but both were connected to the same RTF document. This means that document is covered by (at least) two policies concurrently—not an unusual situation.
- The enforcements were triggered by three different user actions, as shown by the values in the Action column. The third and last instances were caused by different users doing the same thing—trying to change a document’s attributes.
- In all four cases, the enforcement decision was Allow, as shown in the last column.

Keep these details in mind as you analyze the data in the following charts.

Related concepts

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Group by policy chart

When the report is re-run with Bar Chart selected in the Report Type field, and Group By Policy selected in the Show field, Reporter generates a bar chart with four bars, one per policy.

As discussed, four policies are reflected in the table above. Each bar is the same height, indicating one enforcement event per policy, as shown in the following figure.



Figure 71: Group By Policy chart

Grouping events by policy is useful for identifying policies that are being triggered with unexpected frequency, which may be an indication that they are improperly designed and cover users, resources or actions that they should not. It can also be an indication of concentrated efforts at unauthorized data access. To examine the latter possibility, it is often helpful to switch to the **Group by User** option, to focus on who is performing the activity.

Related concepts

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Group by user chart

When the same data is grouped by user, and the bar chart is selected, the chart is generated.

See the following figure. As noted previously, the four policies were each triggered by a different user, and so the graph shows four bars—each representing one user. Each is labeled with a username. In this example, the bars are the same height, since each of the four users triggered a policy once.

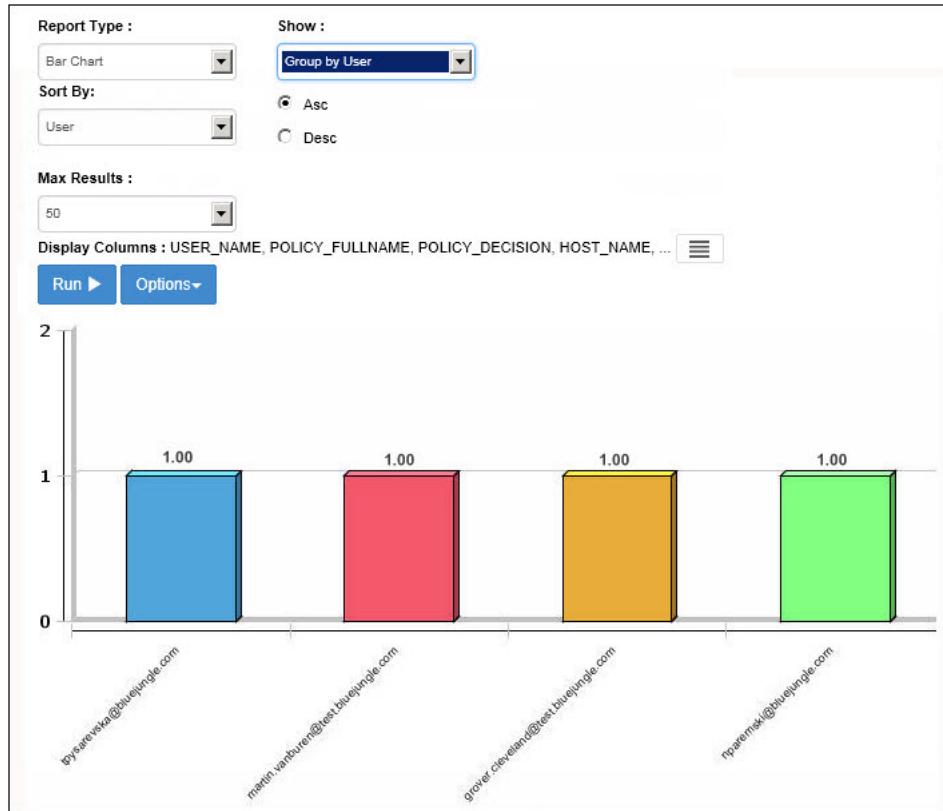


Figure 72: Group By User chart

In a typical live environment, there is little difference among users' rates of triggering policies. If a user shows a significantly disproportionate rate of triggering policies, especially Deny policies, the situation merits closer investigation. The following figure shows an example of how such a case might look in a chart where data is grouped by user.

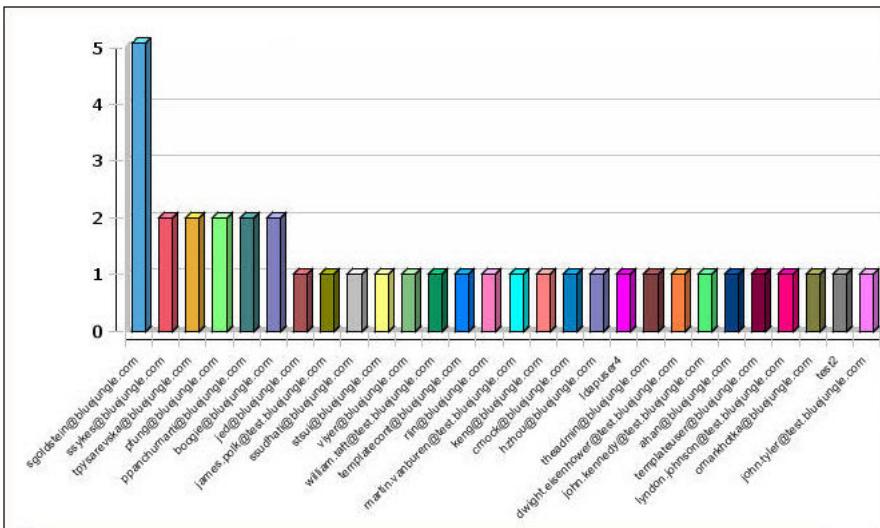


Figure 73: Group by user chart—red flag

Related concepts

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Group by resource chart

When the same data is grouped by resource, the chart shows the extent of specified events—in this case, policies being triggered—per each individual resource covered by the report. In our case, of the four events, two involved the same RTF file (shown in blue) and the other two, one file each (red and orange). This means three resources total, as shown in the following figure.

Because policies often cover large numbers of individual documents or other resources, grouping by resource is only helpful when the number of events has already been narrowed down to a smaller set by various report filters, such as policies or users. A pie charts is ideal here, because in the context of resource use, the relative access activity regarding some single file or other resource as compared to all others, is generally of more interest than any absolute number of instances of access.

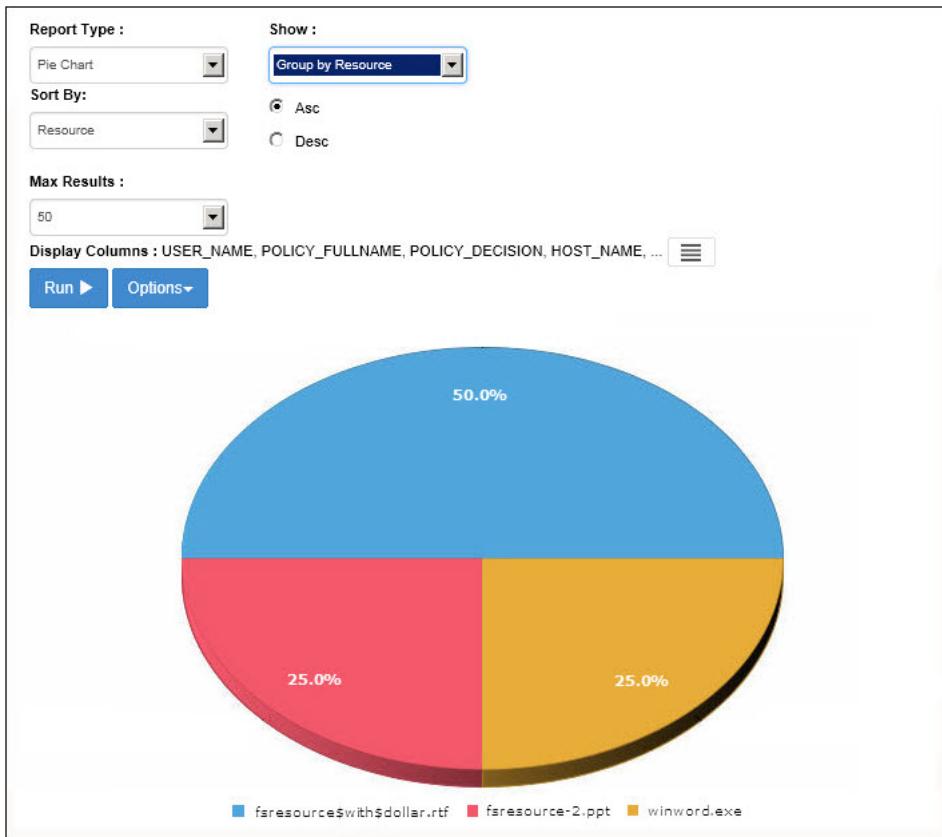


Figure 74: Group by resource chart

Related concepts

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Group by time chart

The **Group by Month** and **Group by Day** options group all events by month and day, respectively.

In our example, since all four events occurred on the same day, the chart shows only one bar, labeled 07 Aug 2014, as shown in the following figure. The height of the bar indicates that four incidents occurred on that date. Because all the events occurred on the same day for a reporting period of one week, the chart appears the same whether the **Group by Month** or **Group by Day** option was selected. It makes sense to select the **Group by Month** option only if the report period you specify spans more than one month.

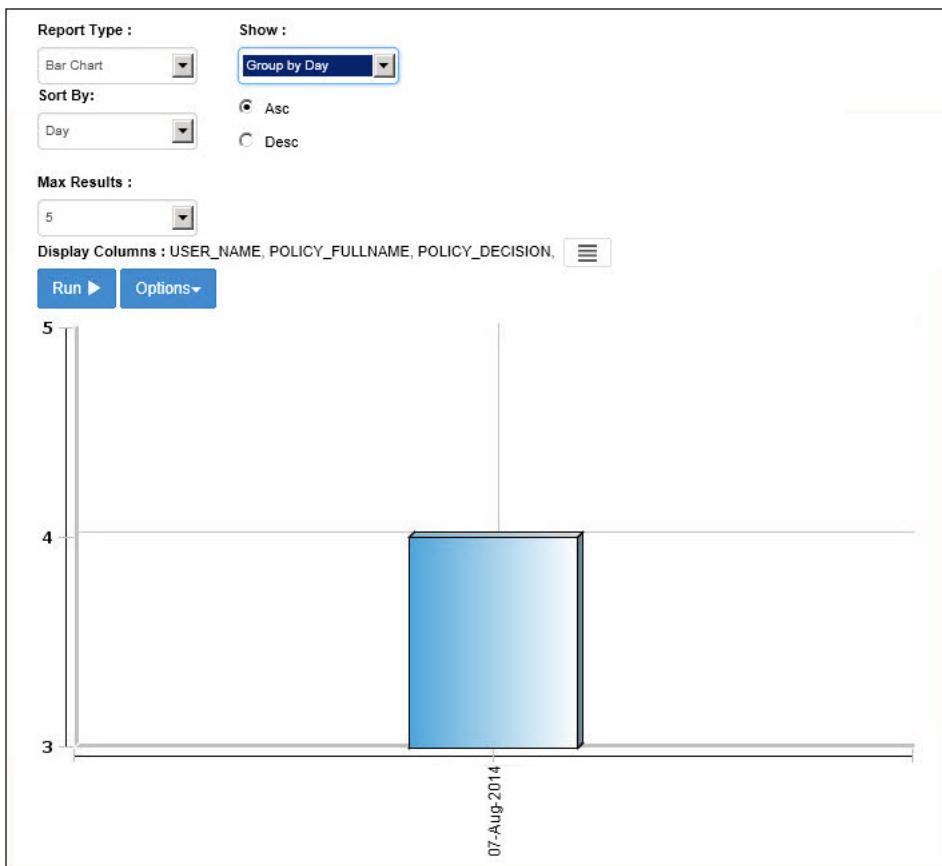


Figure 75: Group By Day chart

In a live environment, this format is particularly useful in identifying time-related enforcement spikes, which are similar to the user-related red flags. Spikes in policy enforcement at specific times, such as weekends or the end of a quarter (especially with regard to Deny policies) can indicate intentional efforts at unauthorized access to sensitive resources or to financial documents prepared at the end of each quarter.

Related concepts

[Sample reports](#) on page 249

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

Report anomalies

Reports can show unexpected results at times, which may seem to reflect incorrect behavior of policies, but in fact are a result of some of the ways Windows works internally, or of inherent statistical issues.

Implied actions

When Windows Explorer handles file operations, it sometimes performs several actions sequentially in a way that is not visible to users.

For example, when you run Copy on a file, Explorer first opens the file, then reads the content, then writes a copy to the new location. Similarly, when you print a file, Explorer must first open the file, then send it to the print server. In fact, most of the actions you deal with in policies also involve Open in this way, as what might be called an implied action.

Because Open is a very common implied action, you may see some reporting anomalies whenever you deploy several policies, one of which involves a Deny Open or Allow Only Open effect. For example, suppose you have a Deny Open policy deployed, then deploy a second policy with a Deny Print effect, test it for a while, and run a report on the test activity. The test behavior might seem correct—printing

is blocked, as intended—but the report does not show any instances where Print actions were denied; instead it shows many cases of Open actions being blocked. This can seem like an error in the policy or in the report, but it actually reflects the fact that, because the actions implicitly triggered the Deny Open policy, they were blocked by that rather than by the Deny Print policy.

Note that if you encounter this kind of seeming anomaly regarding actions in your reports, it is likely that some kind of implied action (involving Open, specifically) is the cause.

Preview activity

When you preview document-type components, the system performs a Read action on the underlying documents to display information about them in Preview pane.

This may lead to unexpected reporting results if, for example, you have deployed policies that prohibit users from reading these document resources. Even though the policy is correctly blocking this activity by users, your report might show instances of Read actions—those performed in the course of Preview. This can be confusing at times, since there is no way Reporter can distinguish between actions performed internally by the system and actions performed by users.

Working with monitors and alerts

This section describes how to use the Monitoring tab to create policy monitors to obtain notifications of policy enforcement activities that meet predefined criteria.

Together, monitors and alerts can provide continuous coverage of key policy enforcements in an enterprise, as well as a notification system that warns administrators when action is required.

Related concepts

[Monitoring tab](#) on page 211

Monitoring functionality and permissions

In Reporter, you use the Monitoring tab to create policy monitors and to view alerts triggered by those monitors.

The functionality available to you depends on the rules established in the Delegated Administration section the console.

When you click the Monitoring tab, the Alerts Overview page appears. It displays a list of alerts, if any, in the order they were raised, with the latest on top.

If there are alerts, you can take the following actions:

- Filter alerts by specified criteria, including time period, monitor, and tag name and value
- View alerts aggregated by tag, monitor, or time, and presented in chart format
- Dismiss any alert from the list
- Delete any alert from the list

Two subtabs appear below Monitoring: Alerts and Monitors. Click **Monitors** to view the list of defined monitors, if any, and to create a monitor.

Figure 76: Monitor without any data

You can perform the following tasks in the Monitors tab, if you have administrator privileges:

- Create monitors
- Edit monitors
- Deactivate and activate monitors
- Delete monitors

Creating a monitor

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

You define the following properties for a monitor:

- The criteria, or filters, that specify what policy activities to watch. As with a report, a monitor can filter activity data by user, resource, user action, policy, application, and host. For example, you can create a monitor to watch all policy activities where downloads of documents classified as Confidential are denied.
- The conditions that determine when an alert is raised. For example, you can create a monitor to raise an alert when the activity specified above occurs more than 10 times in one day, or when more than 10MB of confidential documents have been downloaded in one week.
- The type of alert—an email notification, or an alert displayed in the Alerts Overview page, or both.
- A monitor tag. Although tags are optional, they are useful for organizing monitors and the alerts they generate. In the Alerts tab, you can filter and analyze alerts by tag.

To create a monitor, perform the following steps. Note that steps or properties are optional unless described as required.

1. In the Monitors tab, click the **Add Monitor** button. The Create Monitor dialog appears, with the General tab open.
2. In the General tab, define the general properties of the monitor.
 - In **Name**, type a name for the monitor. This property is required.
 - In **Description**, provide a description of the monitor.
 - In **Duration**, select a value from the drop-down list. Duration is the time period that the monitor considers when evaluating filters and other conditions that you specify for the monitor. This property is required, and the default is Today.
 - In **Group By**, select a grouping option from the drop-down list. This property specifies whether the condition specified in the next property, Aggregators, applies at the selected group level, or to all the policy activity records covered by the monitor (the default).
 - In **Aggregators**, specify a file size or number of records threshold that the monitor must meet or exceed before raising an alert. This property is required.

- In **Alert**, select how you want to receive alerts raised by the monitor. By default, all alerts are listed in the **Alerts Overview Tab**. In addition, you can elect to receive an email alert. If you select the email option, enter an email address.
- In **Level**, select a value from the drop-down list. You can use level to indicate the priority or seriousness of the alert generated by the monitor. In the **Alerts Overview tab**, you can sort alerts by level, so it is useful to assign a meaningful level to each monitor. By default, all alerts are set to L1.
- In **Message**, type a message that you want the alert to display. For example:

Access to ITAR resources was denied more than 10 times today

Depending on the option or options you selected for the Alert property, this message appears in the **Alerts Overview tab**, or in an email, or both.

3. Click the **Filters** tab to define the criteria that determine the policy activities to monitor. The filter options are the same as the ones you can specify in a report query. You can filter on the following: Action, Decision, User, Resource, Policy, and Other attributes.
 4. Click the **Tags** tab to create one or more tags for the monitor.
 - In **Name**, type a name for the tag.
 - In **Value**, type a value for the tag.
 - Click the + button to add the tag to the monitor.
 5. Click the **Access Control** tab to grant other users access to the monitor. Other users can only view the definition of the monitor; they cannot modify it.
- In the Share with section, select one of the following options:
- **Only me**: The monitor is private. No one else can see this monitor or the alerts it generates.
 - **Public**: The monitor, and any alerts it generates, can be viewed by all users who have access to Reporter.
 - **Users/Groups**: The monitor, and any alerts it generates, can be viewed by selected users or user groups. If you select this option, specify each user or group of users with whom to grant access to the monitor. In the text field that appears, type the first letter of the user's first name or the group name. Select a name from the list that appears. Repeat the previous steps to add names as needed.
6. Click **Save**. The monitor is activated. It appears in the list of monitors in the **Monitors** tab, where you can edit, deactivate, or delete it.

Related concepts

[Configuring Control Center components](#) on page 140

A set of sections in configuration.xml control the various components of the Control Center.

[Specifying the file size or records criteria](#) on page 240

The **Aggregator** property requires at least one condition—either file size or number of records—to apply to the activity data that match the specified duration and filters. A monitor can include both conditions.

[Filtering by user resource, and policy](#) on page 221

When filtering policy activity data by user, resource, or policy, you can use either, or both, of the following methods.

[Filtering by action](#) on page 224

You can filter the policy activity data by user actions, such as create, run, or rename.

[Defining a tag](#) on page 240

Assigning a tag is optional, but useful for organizing alerts raised by the monitor, as well as for gathering statistics, by tag, on those alerts.

[Filtering by tag name and tag value](#) on page 242

You can create one or more tags for each monitor.

Specifying the duration

The **Duration** property defines the time period that the monitor covers when evaluating filters and other conditions that you specify for the monitor. For example, to create a monitor that raises an alert when Policy01 denies access to documents classified as ITAR (International Traffic in Arms Regulations) more than 20 times in a week, you would select **Current Week** as the duration.

As long as the monitor remains active, it watches for policy activities that meet the filter conditions each week, and raises an alert if the number of records (policy activities) threshold of 20 is exceeded for that week.

While it is typical to use monitors to track current policy activities and receive alerts when certain conditions occur, monitors can also be used to evaluate past activities. For example, a comparison of historical data with current data can reveal trends, such as whether an organization's information controls are more or less effective from one period to the next. A monitor can evaluate past data up to 90 days prior, which is, by default, the amount of time data is stored in the Activity Journal.

By default, a monitor can only evaluate data that is logged after the monitor's creation date and time. For example, if you create a monitor and specify **Last Week** as the duration, the monitor cannot access data from the previous week until a week later when a week's data has accumulated. If you want to evaluate past activities for a specific time period, plan ahead and create the monitor before that time. Alternatively, administrators can configure monitors to use past data that is stored in the Activity Journal. This configuration is set through the `use.past.data.for.monitoring` property in `configuration.xml`.

The following table describes the time periods you can select for Duration.

Table 51: Duration values

Value	Description
Today	The monitor evaluates data for the current day. Use Today for critical policies whose activities you want to monitor daily for unusual or unacceptable levels of enforcement.
Current Week	The monitor evaluates data for the current week. A week is from Sunday to Saturday.
Current Month	The monitor evaluates data for the current month.
Current Quarter	The monitor evaluates data for the current quarter.
Yesterday	The monitor evaluates data from the previous day.
Last Calendar Week	The monitor evaluates data in the previous week (Sunday to Saturday).
Last Calendar Month	The monitor evaluates data in the previous month (first day to last day in the month).
Last Calendar Quarter	The monitor evaluates data in the previous quarter (first day to last day in the quarter).
Last 7 Days	The monitor evaluates data from the last 7 days. Each day the monitor is active, the window of 7 days moves up by one day.
Last 30 days	The monitor evaluates data from the last 30 days. Each day the monitor is active, the window of 30 days moves up by one day.

Related tasks

[Filtering by relative dates](#) on page 242

Relative dates are the dates that are relative to the current date.

Specifying the file size or records criteria

The **Aggregator** property requires at least one condition—either file size or number of records—to apply to the activity data that match the specified duration and filters. A monitor can include both conditions.

Number of Records is the number of policy activities that meet the duration and filters defined for the monitor.

The other condition you can specify involves file sizes. In the previous monitor examples, the aggregator conditions are applied across all the activity records that match the specified duration and filters. If, however, the monitor specifies a grouping option in the **Group By** property, an aggregator condition applies at the selected group level. Suppose you want to change the previous monitor to do the following: Raise an alert when Policy02 has allowed the downloading of ITAR documents totaling more than 10MB, by any one user, in a week. In **Group By**, you would select **Group by User**. The File Size condition would then apply to user groups, rather than to all activity records.

Related tasks

[Creating a monitor](#) on page 237

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

Defining a tag

Assigning a tag is optional, but useful for organizing alerts raised by the monitor, as well as for gathering statistics, by tag, on those alerts.

In the Alerts page, you can view alerts that are grouped and aggregated by monitor tags.

When deciding if a monitor should be tagged, and what tag or tags to create, think about the ways you want to organize or categorize alerts. For example, you can create a tag that reflects the purpose of the monitor, that is, the type of policies, the application system, the resources, or the groups of users, being monitored. You can also create a tag that indicates the type of risk or risk level associated with the policy activities being monitored.

In the example above, two tags are defined for the monitor, which enables the alerts triggered by the monitor to be organized and analyzed in two ways.

- The SAP tag is set to the value DMS. This indicates that the monitor watches for policies that enforce information controls on the DMS application module of an SAP system. There are probably other monitors that watch policies associated with other modules of an SAP system, such as PLM or EasyDMS. By tagging a monitor as SAP, you can obtain statistics on the number of alerts triggered for SAP in general, as well as for each SAP application module.
- The ITAR Policies tag is set to the value Access Control. In addition to watching for policy enforcement on an SAP DMS system, the monitor also watches for policies that control access to ITAR data. Presumably there are other similar monitors that watch policies that control the storage, use, or distribution of ITAR data. By assigning an ITAR Policies tag, you can obtain statistics on the number of alerts triggered for ITAR policies in general, and for each type of control governing ITAR policies.

Related tasks

[Creating a monitor](#) on page 237

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

Deactivating and deleting monitors

When you create and save a monitor, it is activated automatically, and stays in effect until you deactivate or delete it.

Deactivating or deleting monitors are common tasks during the testing phase. You may want to deactivate a monitor to temporarily suspend it. You can reactivate a monitor at any time. If a monitor no longer serves a purpose, delete it. When you delete a monitor, all alerts it triggered are also deleted.

To deactivate or delete a monitor, click **Monitors** to open the Monitors tab. Select the monitor, and in the Action column, select **Deactivate** or **Delete**. If your system contains a large number of monitors, you can use **Search** to find a monitor. You can search for any value in any of the columns (Name, Description, Created At, and so on) by typing a full or partial value.

Viewing alerts

The Alerts Overview tab provides a summary list of alerts in a table.

By default, the table displays all alerts raised by all monitors in the current day. The alerts are sorted by the time they were raised, starting with the latest.

By default, ten alerts are shown on each page. You can increase the number of alerts to display per page by selecting a number in Show 10 entries. Use the navigation links on the bottom right to browse through all the alerts.

Sorting alerts

By default, alerts in the Alerts Overview table are sorted by date and time values in the Raised At column.

You can select a different column—Level, Monitor Name, or Alert Message—on which to sort by clicking the column name in the table header. By also clicking the column header, you can change the sorting order of a column from descending to ascending order, and vice versa.

Filtering alerts

You can filter the list of alerts to view only the ones of interest.

You can filter by dates (specific or relative), monitor name, tag name, or tag value as shown in the following figure.

Alerts Overview				
Level	Monitor Name	Alert Message	Raised At	Action
No data available in table				
Showing 0 to 0 of 0 entries				

Figure 77: Filtering alerts

Filtering by specific dates or times

A specific time period can be any span of time.

For example, you can define a filter that selects data for a specific week or day. You can narrow the data to a range of hours, or even minutes, in a specific day. To display alerts triggered in a specific time period, perform the following steps.

Alerts Overview

Date mode: Date Selection:

Monitor: Tag Name: Tag Value: Show Dismissed Go

Figure 78: Filtering by specific dates or times

1. In Date mode, select **Fixed Dates**.
2. In **From** and **To**, specify the start date/time and end date/time, respectively, of the time period in which the alerts you want to view, occurred. Click in the field to choose a date and/or time from the calendar.
3. Click **Go**.

Related concepts

[Alerts by monitor tag](#) on page 243

When you select the option to view alerts by monitor tag, Reporter groups alerts by tag name, and presents the data in a treemap.

[Alerts by monitors](#) on page 244

When you select the option to view alerts by monitor, Reporter groups alerts by monitor name, and presents the data in a bar chart.

[Alerts by time](#) on page 245

When you select the option to view alerts by time, Reporter groups alerts by the time period that you specify, and presents the data in a bar chart; if you don't change the default settings, the time period is the current day.

Filtering by relative dates

Relative dates are the dates that are relative to the current date.

For example, you may want to view alerts that were triggered yesterday or this week. You can display alerts by relative dates.

1. In Date mode, use the default option **Relative**.
2. In **Date Selection**, select a time period from the drop-down list.
3. Click **Go**.

Related concepts

[Specifying the duration](#) on page 239

[Alerts by monitor tag](#) on page 243

When you select the option to view alerts by monitor tag, Reporter groups alerts by tag name, and presents the data in a treemap.

[Alerts by monitors](#) on page 244

When you select the option to view alerts by monitor, Reporter groups alerts by monitor name, and presents the data in a bar chart.

[Alerts by time](#) on page 245

When you select the option to view alerts by time, Reporter groups alerts by the time period that you specify, and presents the data in a bar chart; if you don't change the default settings, the time period is the current day.

Filtering by monitor

You have the option of viewing alerts triggered by all monitors (the default) or by a specific monitor.

In **Monitor**, select All, or one of the monitors in the drop-down list.

Filtering by tag name and tag value

You can create one or more tags for each monitor.

Each tag is a name-value pair. You can filter alerts by either name or value, or both.

Related tasks

[Creating a monitor](#) on page 237

Creating a monitor is similar to creating a report. Both access policy activity data from the Activity Journal.

Displaying alert details

You can view all the policy enforcement events associated with the alert.

To display this information, select **Details** in the Action column.

Deleting and dismissing alerts

The system retains all alerts until you delete them.

If there are alerts you no longer need to see or include in analyses, you should delete them. For each alert to delete, select **Delete** in the Action column.

If you want to keep an alert in the system for statistical reasons, but want to exclude it from the table in Alerts Overview, select **Dismiss** in the Action column. To re-display alerts that you have dismissed, select **Show Dismissed** above the table.

Analyzing alerts

In addition to displaying the complete list of alerts, Reporter provides the option of viewing and analyzing alerts.

Alerts are grouped by tag, monitor, or time, aggregated, and presented in charts. To view alerts in one of these formats, in the Alerts Overview tab, select an option from the menu on the left, as shown in the following figure.

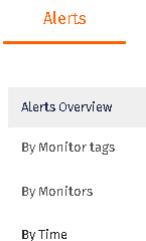


Figure 79: Alerts menu

Alerts by monitor tag

When you select the option to view alerts by monitor tag, Reporter groups alerts by tag name, and presents the data in a treemap.

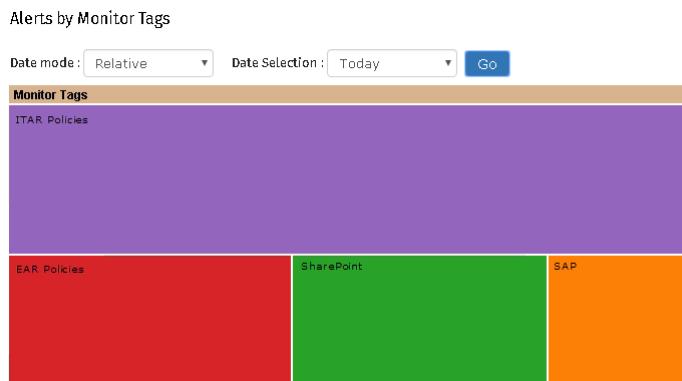


Figure 80: Alerts by Monitor Tag treemap

Each rectangle in a treemap represents a tag, and the rectangle sizes let you compare at a glance the number of alerts associated with each tag.

To see the distribution of alerts by tag value for a particular tag, click the rectangle representing the tag. The following figure shows the distribution of alerts, by tag value, for the ITAR Policies tag.

To go back to the top-level treemap, click the **Monitor Tags. ITAR Policies** title bar. To display the list of alerts associated with a tag value, click the corresponding rectangle.

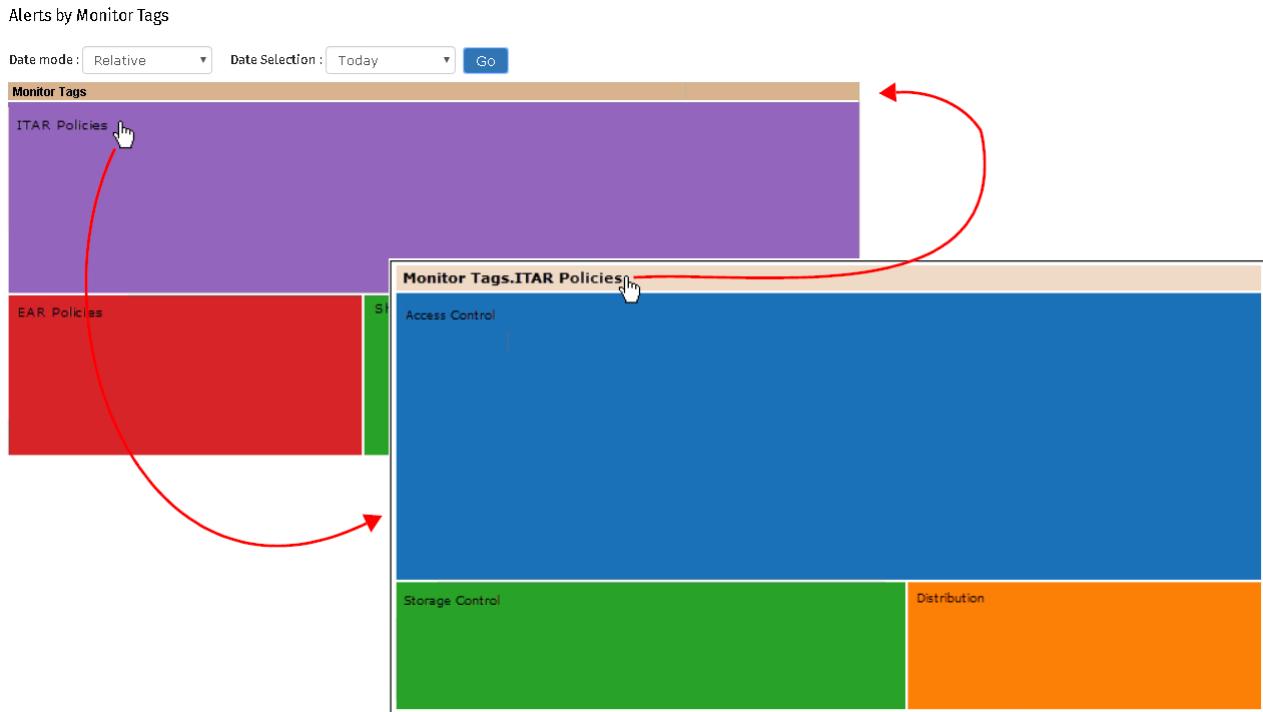


Figure 81: Clicking a tag to view the distribution of alerts by tag value

By default, the treemap shows alerts for the current day. To change the time period, in **Date mode**, select the value you want.

Related tasks

[Filtering by specific dates or times](#) on page 241

A specific time period can be any span of time.

[Filtering by relative dates](#) on page 242

Relative dates are the dates that are relative to the current date.

Alerts by monitors

When you select the option to view alerts by monitor, Reporter groups alerts by monitor name, and presents the data in a bar chart.

As shown in the following figure, the labels in the x-axis show the monitor names.

By default, the chart shows the number of alerts for each monitor, for the current day. To change the time period, in **Date mode**, select the value you want.

To display the list of alerts associated with a monitor, click the corresponding bar.

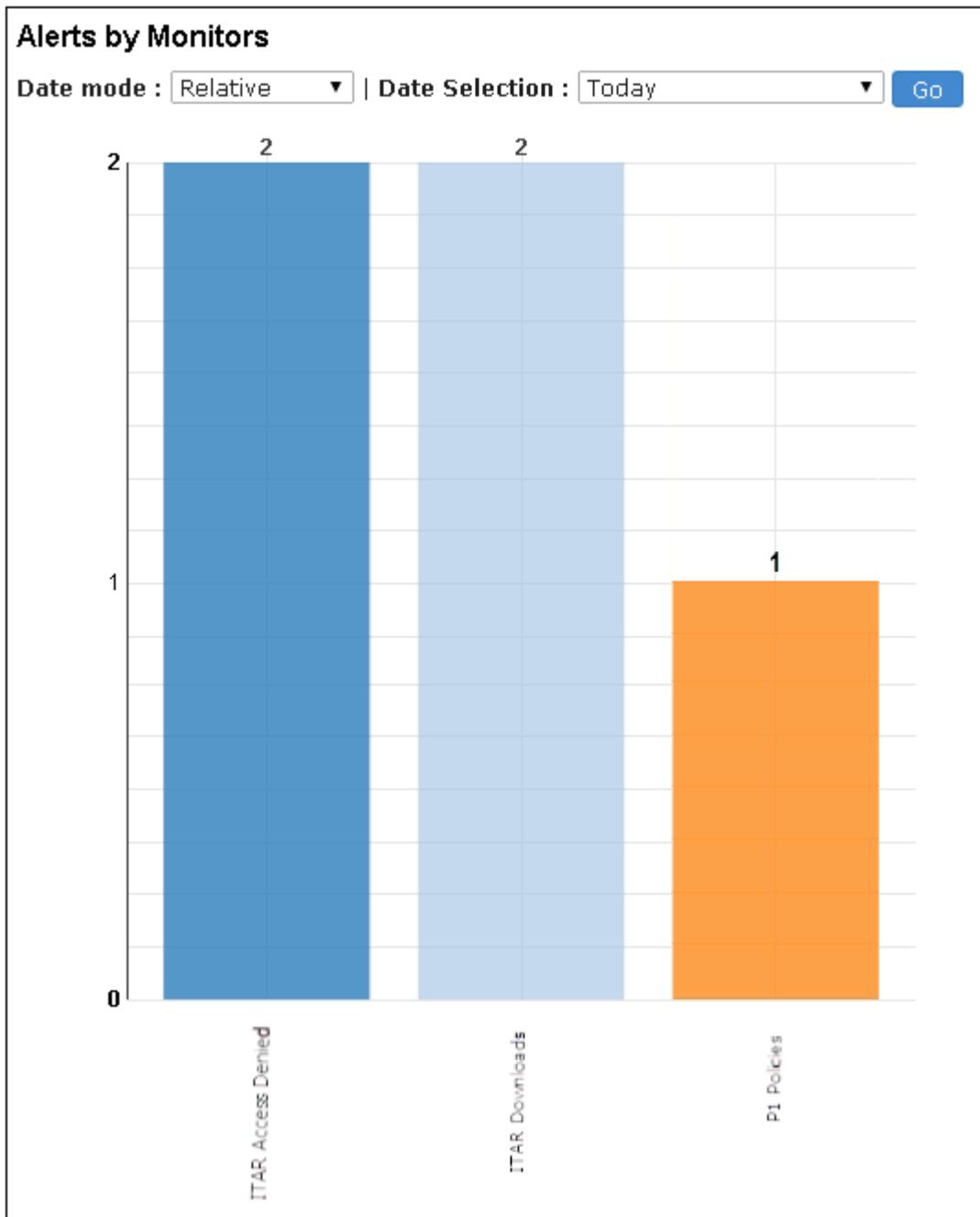


Figure 82: Alerts by Monitor chart

Related tasks

[Filtering by specific dates or times](#) on page 241

A specific time period can be any span of time.

[Filtering by relative dates](#) on page 242

Relative dates are the dates that are relative to the current date.

Alerts by time

When you select the option to view alerts by time, Reporter groups alerts by the time period that you specify, and presents the data in a bar chart; if you don't change the default settings, the time period is the current day.

To change the time period, in Date mode, select either Fixed Dates or Relative.

The following figure shows two chart examples. The chart on the left shows the number of alerts for the current day (the default), and the chart on the right shows the number of alerts per day for the current week. In the second chart, Date mode is set to Relative, and Date Selection is set to Current Week. The labels in the x-axis show the dates. Click a bar in the chart to view the alerts for that day.



Figure 83: Two charts, one displaying alerts for the current day, the second for the current week

In the examples above, alerts are grouped by day. If Date Selection is set to Current Quarter or Last Quarter, alerts are grouped by month.

Related tasks

[Filtering by specific dates or times](#) on page 241

A specific time period can be any span of time.

[Filtering by relative dates](#) on page 242

Relative dates are the dates that are relative to the current date.

Using Audit Logs

This section describes how to use the Audit Logs tab to view activity logs.

About Audit Logs

In Reporter, you can use the Audit Logs tab to view audit logs.

The audit logs include the creation, modification, and deletion of the following entities:

- Action
- Application User
- Delegate Admin Rule
- Policy
- Policy Model

- Report
- Resource
- Subject

Audit logging is an automated function, and does not require any configuration.

Viewing Audit Logs

To view audit logs, log in to Control Center with an account that has permission to view reports, such as the superuser Administrator account.

1. In the Control Center console, navigate to **Reports > Audit Logs**.
2. In the Audit Logs pane, define the query criteria.

Table 52: Audit logs query criteria

Field	Description
From and To	The start date and time, and end date and time, respectively, of the time period this report covers. Click in the field to choose a date and time from the calendar.
Action	The actions you want to show the log for. You can select one of the following: <ul style="list-style-type: none"> • All • Create • Update • Delete • Deploy • Undeploy
Entity Type	The type of entity you want to show the log for. You can select one of the following entities: <ul style="list-style-type: none"> • All - Selects all the entities • Action • Application User • Delegate Admin Rule • Policy • Policy Model • Report • Resource • Subject
User	If you define a user, or several users, the log only shows the activities from that user(s). You can type the first letter of the username, and select from the drop-down list.
Entity ID	The value of the ENTITY_ID Column in ENTITY_AUDIT_LOG table. All entities such as policy model resource, subject/resource/action components, policies, users, and delegation policies have entity IDs.
Sort By	You can sort the result of your query by date, entity type, or action.
Asc or Desc	You can sort the result of your query by the order it appears in. You can select ascending or descending from the drop-down list.

3. Click Run.

4. The query results appear in a table at the bottom of the page.

The screenshot shows the NextLabs interface with the 'Audit Logs' tab selected. The 'Query Criteria' section includes fields for 'From' (2018-04-01 00:00:00), 'To' (2018-05-04 23:59:59), 'Action' (All), 'Entity Type' (All), 'User' (empty), 'Entity ID' (empty), and 'Sort By' (DATE, Asc). Below the table, there is a 'Run ▶' button and a 'Show 10 ▾ entries' dropdown.

Timestamp	User Name	Action	Entity Type
2018-12-27 17:43:41	Administrator	DEPLOY	Policy
2018-12-27 17:43:41	Administrator	DEPLOY	Subject
2018-12-27 17:43:40	Administrator	DEPLOY	Subject
2018-12-27 17:43:40	Administrator	DEPLOY	Action
2018-12-27 17:43:39	Administrator	DEPLOY	Resource
2018-12-24 12:08:11	Administrator	CREATE	Resource
2018-12-21 13:30:12	Administrator	CREATE	Policy Model
2018-12-21 13:29:10	Administrator	CREATE	Policy Model
2018-12-21 11:52:46	Administrator	DELETE	Policy Model
2018-12-21 10:57:41	Administrator	CREATE	Policy Model

Figure 84: Audit logs

You can show the number of results you want to see, and search within the results.

- To view entity details, click **Details**. For example, entity details for CREATE and DELETE operations are displayed as shown in and entity details for UPDATE operations are displayed as shown in the following figure.

Entity Details

```
{
  "Component Type": "RESOURCE",
  "Policy Model": {
    "Policy Model ID": 12,
    "Name": "Classified Documents",
    "Version": 0
  },
  "Display Name": "Import and Export classified documents",
  "Description": "",
  "Tags": "classified",
  "Conditions": [
    {
      "Attribute Short Name": "dept",
      "Operator": "!=",
      "Value": "customer service"
    }
  ],
  "Include Sub-Components": null,
  "Status": "DRAFT",
  "Version": 0
}
```

Figure 85: Entity details for CREATE and DELETE operations

Entity Values

Old Value	New Value
<pre>{ "Component ID": 62, "Name": "MOVE", "Version": 3 }, { "Obligations on Allow": null, "Obligations on Deny": [{ "Policy Model ID": 0, "Name": "log" }, { "Policy Model ID": 14, "Name": "Display user Alert", "Parameters": { "akert": "the user john.tyler is blocked" } }], { "Schedule": null, "Manual Deploy": false, "Deployment Time": 1546483725107, "Deployed": true, "Version": 4 }</pre>	<pre>{ "Component ID": 62, "Name": "MOVE", "Version": 3 }, { "Obligations on Allow": null, "Obligations on Deny": [{ "Policy Model ID": 0, "Name": "log" }, { "Policy Model ID": 14, "Name": "Display user Alert", "Parameters": { "akert": "the user john.tyler is blocked" } }], { "Schedule": null, "Manual Deploy": false, "Deployment Time": 1546483725107, "Deployed": false, "Version": 5 }</pre>

Figure 86: Entity details for UPDATE operations

Sample reports

This section presents examples of different report formats, all representing a small set of event data returned by the same report query.

The query requests all events by all users on all resources for one week (August 3, 2014 to August 9, 2014). By comparing the examples, you can get a better understanding of the way the different formats can be used to highlight different aspects of the same data.

Related concepts

[Table](#) on page 230

[Group by policy chart](#) on page 231

When the report is re-run with Bar Chart selected in the Report Type field, and Group By Policy selected in the Show field, Reporter generates a bar chart with four bars, one per policy.

[Group by user chart](#) on page 232

When the same data is grouped by user, and the bar chart is selected, the chart is generated.

[Group by resource chart](#) on page 233

[Group by time chart](#) on page 234

The **Group by Month** and **Group by Day** options group all events by month and day, respectively.

Related tasks

[Creating a report](#) on page 217

A report is a formatted view of a particular set of information extracted from the Activity Journal, a set of database tables where policy activity data is logged.

Event Details reports

The Event Details report reflects on a set of policy activity data.

The following figure shows the Event Details report on a small set of policy activity data.

Showing page 1 of 1						
Date	User	Policy	Resource	Action	Enforcement	
Aug 7, 2014 1:54 PM	grover.cleveland@test.bluejungle.com	APolicy13	fresource\$with\$ollar.rtf	User Logout	Allow	
Aug 7, 2014 2:47 PM	martin.vanburen@test.bluejungle.com	CPolicy32	fresource\$with\$ollar.rtf	Voice Call / Video Call	Allow	
Aug 7, 2014 6:03 PM	nparemski@bluejungle.com	CPolicy3	winword.exe	Change Attribute	Allow	
Aug 7, 2014 5:50 PM	tpysarevska@bluejungle.com	CPolicy37	fresource-2.ppt	Change Attribute	Allow	

Figure 87: Event Details Report

The Event Details report reflects the following:

- Four instances of policy enforcement were logged—that is why there are four rows in the table.
- As the Date column shows, all four instances occurred on the same day, June 9.
- Each was triggered by a different user, as shown by the four different values in the User column.
- Each represents a different policy, as shown by the four different values in the Policy column.
- Only three resources were involved. The first two rows show enforcement to two different policies by two different users, but both were connected to the same RTF document. This means that document is covered by (at least) two policies concurrently—not an unusual situation.
- The enforcements were triggered by three different user actions, as shown by the values in the Action column. The third and last instances were caused by different users doing the same thing—trying to change a document’s attributes.
- In all four cases, the enforcement result was Allow, as shown in the last column.

Bear these details in mind as you analyze the Group by Details reports below.

Group By Policy reports

When you run a report with Show Group By Policy selected, the result a bar chart with four bars, one per policy. For each, the height is the same, indicating one enforcement instance apiece.

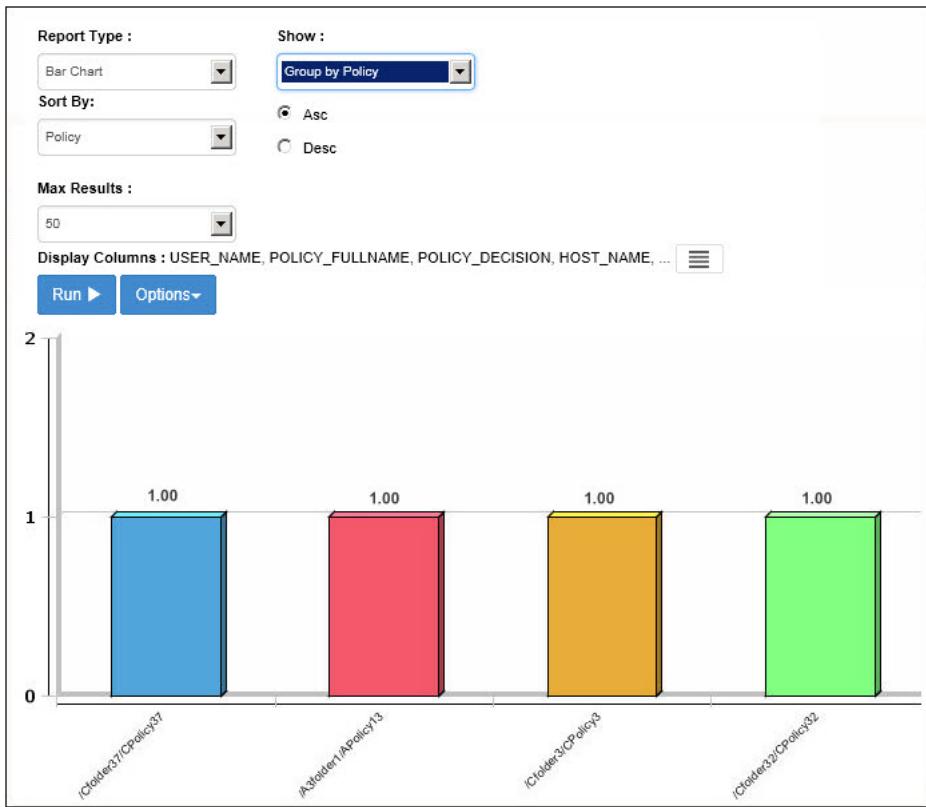


Figure 88: Group By Policy Report

This format is useful for identifying policies that are being triggered with unexpected frequency, which may be an indication that they are improperly designed and cover users, resources or actions that they should not. It can also be an indication of concentrated efforts at unauthorized data access. To look more closely into the latter possibility, it is often helpful to switch to Group by User format, to focus on who is performing the activity.

Group By User reports

Data can be grouped by user.

As shown in the following figure, the four policies are each triggered by different users, and so the graph shows four bars—each representing one user. Each is labeled with a username. In this example the bars are the same height, since each of the four users triggered a policy one time.

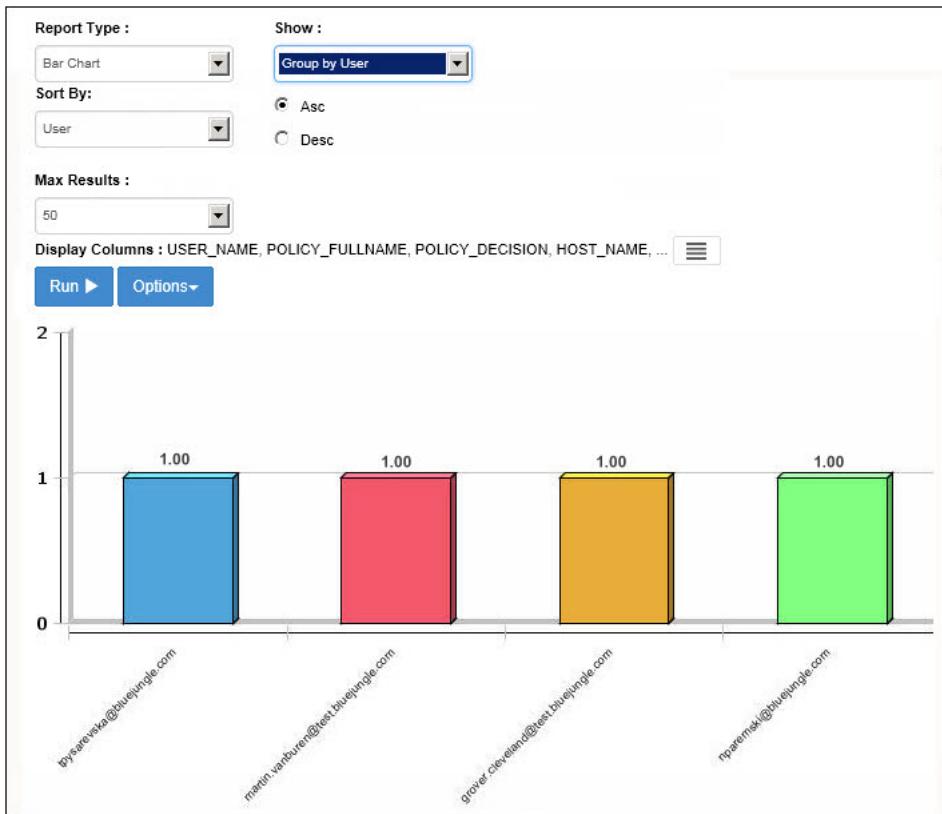


Figure 89: Group By User Report

All things being equal, in a live environment there will generally be little difference among users' rates of triggering policies. In cases where a big difference does arise—when one user shows a significantly disproportionate rate of triggering policies (especially Deny policies)—then the situation will likely merit closer investigation. The following figure shows how such a case might look in a Grouped by User Report.

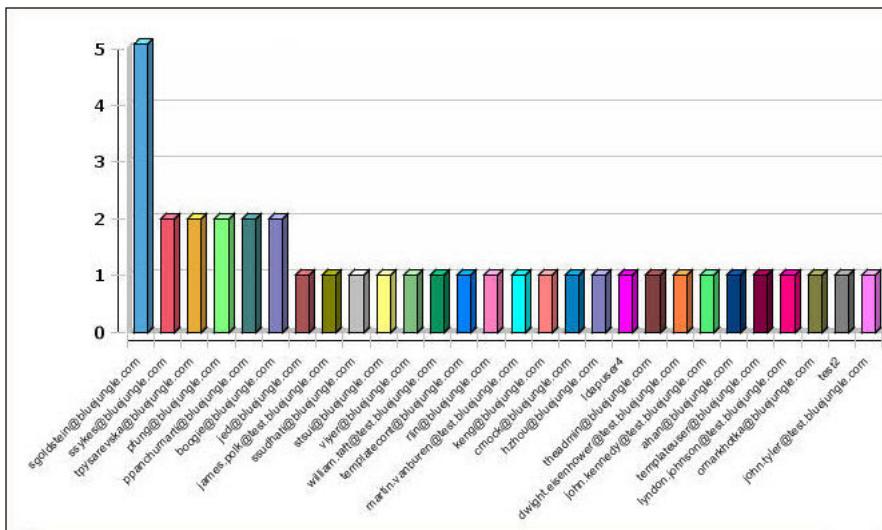


Figure 90: Group By User Report—Red Flag

Group By Resource reports

Group by Resource, shows the extent of specified events—in this case, policies being triggered—per each individual data resource covered by the report.

In our case, we know that of the four instances, two involved the same RTF file (shown in blue) and the other two, one file each (red and orange). This means three resources total, as reflected in the pie graph below.

Because policies often cover very large numbers of individual documents or other resources, this format will only be helpful when the number of incidents has already been narrowed down to a sample by various report filters, such as policies or users. This is why a pie chart format is manageable in this context. Pie charts are preferable here since in the context of resource use, the relative access activity regarding some single file or other resource as compared to all others, will generally be of more interest than any absolute number of instances of access.

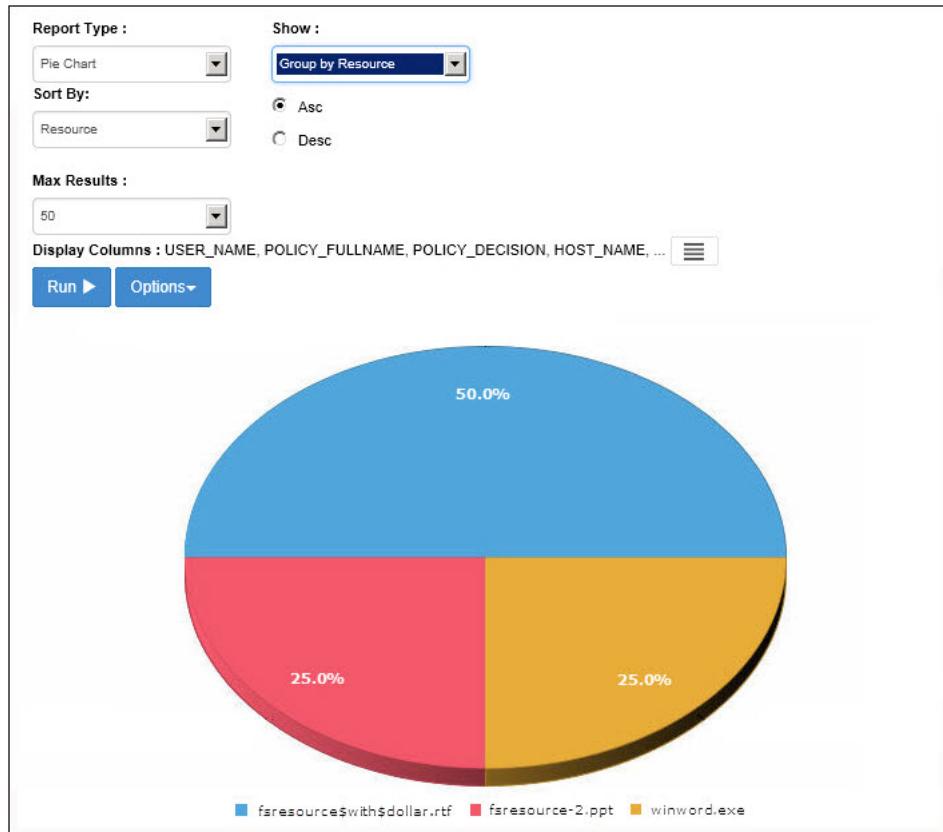


Figure 91: Group By Resource Report

Group By Time reports

When you select Show Group By Time and re-run the report all reported events are grouped by time.

Since we know that all four events in the report occurred on the same day, this chart shows only one bar, labeled June 9 as we would expect. The height of the bar indicates that four incidents occurred on that date—also as expected.

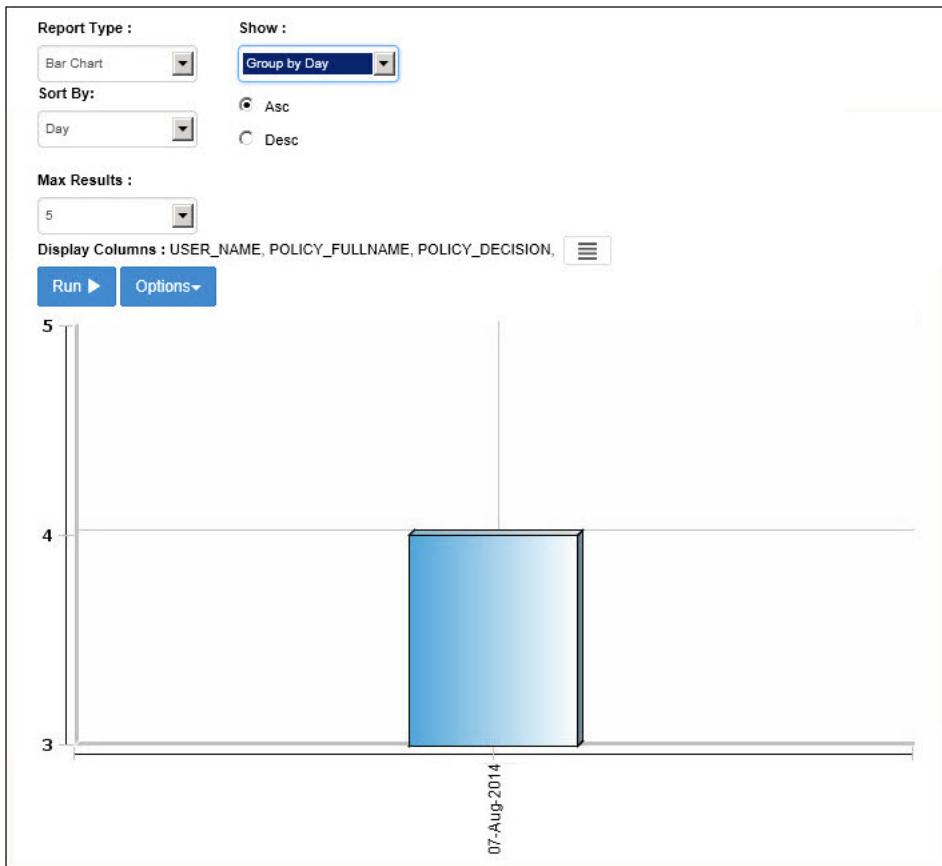


Figure 92: Group By Time Report

In a live environment, this format is particularly useful in identifying time-related enforcement spikes, which are similar to the user-related red flags. Spikes in policy enforcement at specific times such as evenings or weekends (especially with regard to Deny policies) can indicate intentional efforts at unauthorized access to sensitive resources.

Chapter

13

Overview of PEP development

Topics:

- [About PEPs and interactions with PDPs](#)
- [Applications to be enforced and available SDKs](#)
- [PEP development process](#)
- [Policy Controller components](#)

This section provides an overview of PDPs (policy decision points) and PEPs (policy enforcement points). It also describes how to develop them using SDKs and APIs.

Related concepts

[About policy enforcers](#) on page 107

Policy enforcers are Control Center clients that monitor document access and use, and enforce policies at the point of use.

About PEPs and interactions with PDPs

PEPs (policy enforcement points) are software components that intercept user events and enforce authorization decisions provided by PDPs (policy decision points).

Developers can create PEPs to enforce policies for applications written in languages such as Java, .NET, JavaScript, and PHP.

 **Note:** PDPs are also known as Policy Controllers (PCs).

Types of PDPs

NextLabs supports remote and embedded PDPs.

Related concepts

[PEP client SDK \(OpenAz API\)](#) on page 262

[About the OpenAz PEP client SDK](#) on page 263

OpenAz is an open source initiative that provides a standards-based interface.

Remote PDPs

Remote PDPs are deployed on servers that are separate from PEPs.

Remote PDPs include:

- Windows Server PDP (CE-PC): Typically used with NextLabs enforcers for Microsoft products such as SharePoint, Exchange Server, and Skype for Business.
- RestPDP: Typically used with applications that can invoke REST APIs as shown in the following figure. Examples include J2EE applications that run in Apache Tomcat or JBoss, .NET applications running in IIS. NextLabs enforcer for SAP can also use the Java PDP.

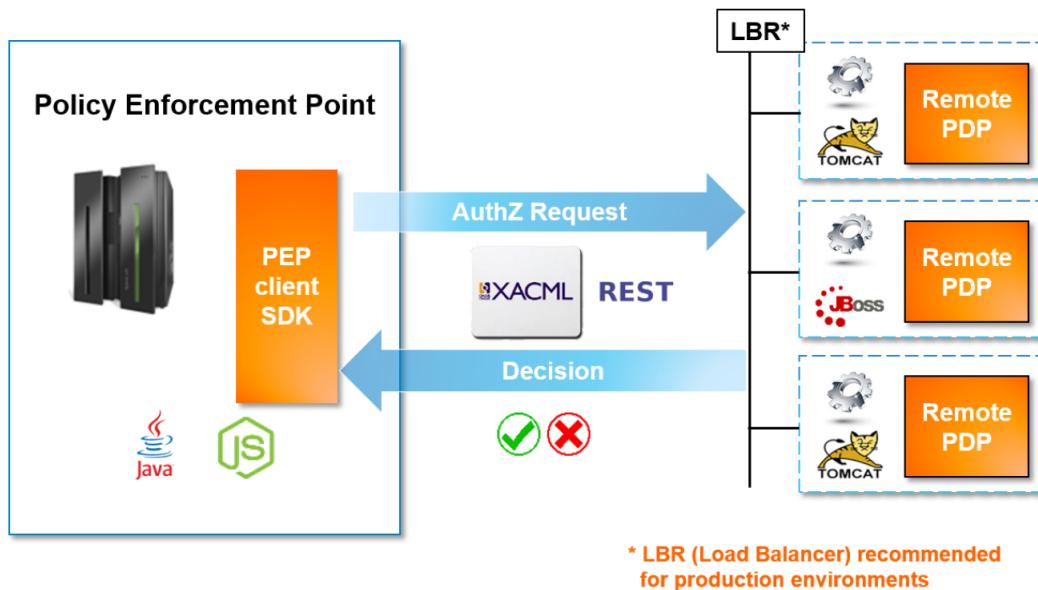


Figure 93: Remote PDP - PEP interaction using PEP client SDK

Embedded PDPs

Embedded PDPs are deployed with PEPs and they can be running as the same application context as PEPs.

Embedded PDPs include:

- Endpoint PDPs: These are shipped with NextLabs endpoint products such as Outlook Enforcer and Windows Desktop Enforcer.

- Java embedded PDPs: These are used with J2EE applications where performance is paramount and PEPs do not want to incur network latency. Examples include NextLabs Rights Management Server and other custom applications that run in Apache Tomcat or Red Hat JBoss. The following figure illustrates interaction between PEPs and Java embedded PDPs.

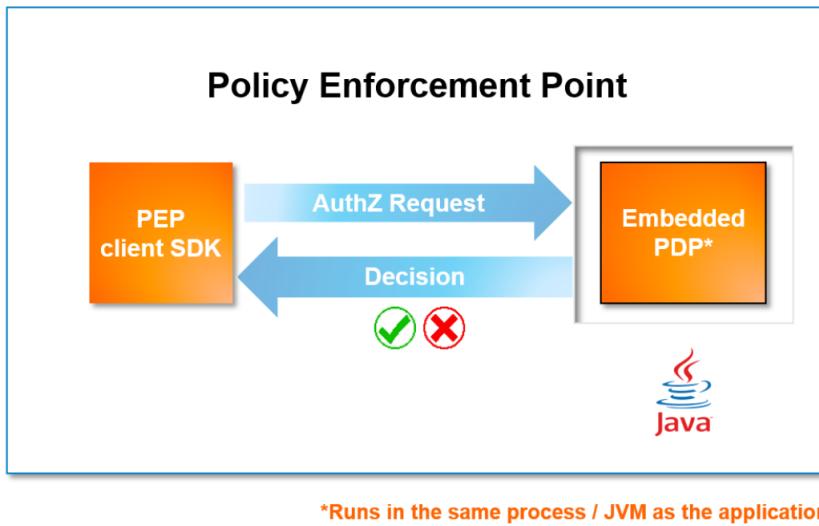


Figure 94: Embedded PDP - PEP interaction using PEP client SDK

Using embedded PDPs

An alternative to installing standalone PDPs for Java is to embed the PDP in a web application.

Embedding the PDP for Java integrates it with the application, simplifying the installation process.

Related tasks

[Configuring the embedded Policy Controller for Java on Tomcat servers](#) on page 271

The embedded Policy Controller for Java is supported on Tomcat servers. This section explains how to set up the web application.

Applications to be enforced and available SDKs

All the SDKs provide the same core functionality—the means of creating a PEP to send policy evaluation requests to the PDP and to receive policy decisions (deny or allow) from the PDP.

Each request passed to the PDP contains event details, such as the user, action, resource, and application involved in the event. The following table provides an overview of the applications to be enforced and the SDKs used to create PEPs.

Table 53: PEP technologies and available SDKs

PEP technology (the language of the application)	Suggested PEP client SDK	Description	Detailed information
Java web application integration with in-process PDP	OpenAZ PEP API with embedded PDP	Single PEP with single PDP for better performance	Using the Java client library on page 266
JavaScript applications	OpenAZ PEP JavaScript client SDK	Multiple PEPs built using JavaScript with central PDP to improve scaling	Using the JavaScript client library on page 280
C and C++ applications with on-premise deployment of NextLabs CE-PC	C & C++ SDK	.NET applications that use central server PDP or CE-PC	C and C++ API reference on page 328
Other cloud applications with external PDP	REST API	PEPs built in other technologies such as PHP that use remote or external PDPs	REST API for external PDPs on page 286

PEP development process

Building PEPs involves these steps.

1. Set up a Control Center development environment: This includes installing Control Center, a PDP, and one or more desktops where you can test the policy enforcement points you are developing on the application you want to control. Setting up a development environment enables you to deploy policies and conduct tests without affecting your production system.
2. Prepare the environment for the targeted application. Prepare a software development environment for the targeted application. This should include your compiler of choice, the application or system you are developing to, debug libraries, and everything you need to control the application.
3. Learn the application. Learn how to intercept all of the application actions your policies might want to control, and how to read the resources and properties that the application manipulates. You must also understand how to stop an application action and modify it. After that, all that is left to do is link the policy aspect into the application controls they have established; this is the function of the SDK code.
4. Add the SDK to the development environment. Install the SDK in the development environment.
5. Develop interceptors. Use the SDK to build a set of interceptors to call into the PDP at runtime and obtain policy decisions for various actions involving the application.
6. Test the PEP. Create and deploy a policy to the test host. You can test PEP behavior using your own debugger or using audit logs to track progress. When PEP behavior is fine-tuned and debugged, you have a running PEP for the chosen application.
7. Deploy the PEP to the production environment. When your development is finished and fully tested, deploy your PEP to each host where policies need to be enforced in your production environment.

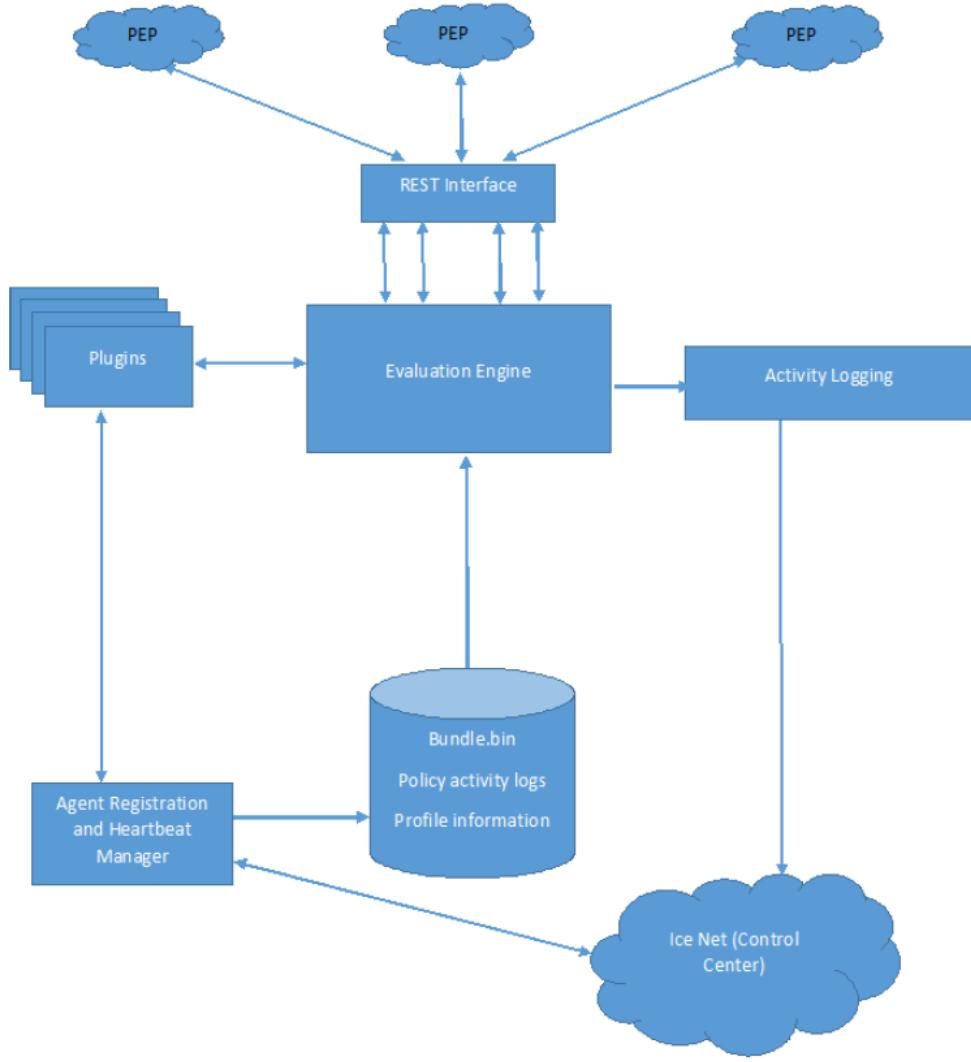


Figure 95: System overview

Related concepts

[Configure Control Center](#) on page 11

To use NextLabs Control Center to enforce policies, the system must be configured for your environment, and policies must be designed and deployed.

[Installing the Policy Adapter SDK](#) on page 321

This section provides instructions for installing the Policy Adapter SDK.

[Configuring a development environment](#) on page 263

To use this API, you must configure a development environment as described in the PEP development process section.

[How to set up the REST API](#) on page 286

To use the REST API, the Java Policy Controller must be installed and configured. For more information, see the NextLabs Control Center Installation and Upgrade Guide.

Policy Controller components

The functional components of the Policy Controller (PDP) include the following.

- Context Manager: Keeps constant track of the environmental context of all events, and provides this information to the Policy Engine and PEP. Context includes user identity, computer hostname, network connection type, and date and time.
- Policy Evaluation Engine: Evaluates policies using multiple criteria, and returns decisions as to whether policies apply to events. This is the PDP of each adapter.
- Obligation Manager: For any policy that evaluates to True, initiates an obligation by sending a request to a Policy Adapter's obligation services or executing built-in obligations. Contains three sub-components:
 - Application Extender: Launches an application or custom function for post-decision processing.
 - Notification Obligation: Sends messages to recipients or targets listed in policies.
 - Policy Logger: Collects and logs activity details and policy decision results.
- Controller Manager: Records non-policy activities, updates the configuration, and secures the controller. Components include:
 - Activity Recorder: Records activities tracked by the Policy Adapter in real time.
 - Configuration Manager: Applies profile and system configuration changes, in real time.
 - Policy Authentication: Authenticates the policy set from a Policy Server and encrypts it on the local file system.
 - Tamper Resistance: Prevents tampering of Policy Controller service, installed files, registry settings, and automatic start-up.
- ICENet Client: Provides the interface for all communication with the Policy Server, via the ICENet Server. This includes deploying new or changed policies, periodically sending activity logs from each control point, and providing controller health status.

Chapter

14

PEP client SDK (OpenAz API)

Topics:

- [About the OpenAz PEP client SDK](#)
- [Configuring a development environment](#)
- [Configuring one-way and two-way SSL authentication between PDP and PEP](#)
- [Using the Java client library](#)
- [Using the JavaScript client library](#)
- [REST API for external PDPs](#)
- [Overview of plug-ins](#)

This section describes how to use the OpenAz API with external and embedded PDPs (policy decision points).

External PDPs are separate from PEPs (Policy Enforcement Points), whereas embedded PDPs are embedded in the process of the PEP.

Related concepts

[Types of PDPs](#) on page 257

NextLabs supports remote and embedded PDPs.

About the OpenAz PEP client SDK

OpenAz is an open source initiative that provides a standards-based interface.

This interface enables developers to interact with an Attribute Based Access Control (ABAC) system. The OpenAz PEP client SDKs provide an abstraction layer that PEP (Policy Enforcement Point) developers can use to enforce authorization. In addition, if you are using the Java client SDK, this abstraction layer enables you to switch seamlessly among the PDPs (Policy Decision Points) of various ABAC providers. NextLabs provides Java and JavaScript client SDKs.

- ! **Important:** OpenAz PEP client SDKs can be used to make authorization requests from Java embedded PDPs, REST PDPs, and CloudPDPs.

Related concepts

[Types of PDPs](#) on page 257

NextLabs supports remote and embedded PDPs.

Configuring a development environment

To use this API, you must configure a development environment as described in the PEP development process section.

An alternative to installing a standalone version of Policy Controller for Java is to embed it in a web application. If your PEP is part of a web application, embedding the Policy Controller for Java integrates it with the application, simplifying the installation of your application.

Related concepts

[PEP development process](#) on page 259

Building PEPs involves these steps.

Configuring one-way and two-way SSL authentication between PDP and PEP

For the REST API, you can configure the communication between a PDP server and a PEP client using one-way or two-way SSL authentication.

In one-way SSL, only the PEP client validates the PDP server to ensure that it receives data from the PDP. In the following illustration, CA certificate B exists in the TrustStore on the PEP client and also in the KeyStore on the PDP server.

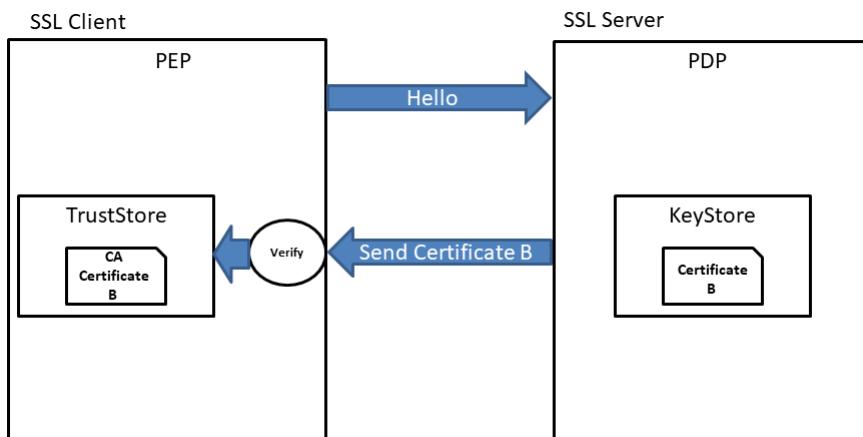


Figure 96: One-way SSL

In two-way SSL, both the PEP client and the PDP server authenticate each other to ensure that both parties involved in the communication are trusted. In the following illustration, there is a CA certificate A in the TrustStore and a CA certificate B in the KeyStore on both the PEP client and the PDP server.

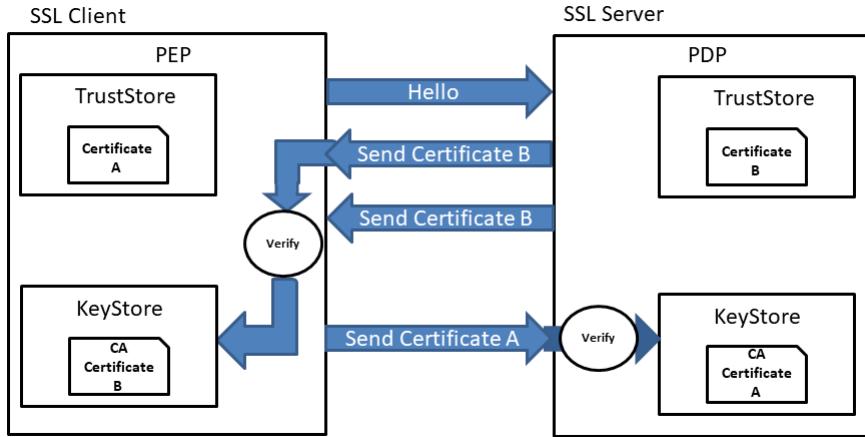


Figure 97: Two-way SSL

Configuring one-way SSL for Java Policy Controller

1. Create a KeyStore for the Java Policy Controller using a self-signed certificate or a certificate issued by a trusted authority. In this document, it is called `pdp-keystore.jks`.
2. Export the certificate from the newly-created KeyStore.
3. Import the exported certificate to the PEP TrustStore.
4. Copy the `pdp-keystore.jks` file to the PDP server.
5. Configure the PDP server to use SSL using the `pdp-keystore.jks` file.
 - On Tomcat, edit the `server.xml` file to add a connector under the `<Service name='CE-Core'>` section. That is, specify the values for `keystoreFile` and the encrypted `keystorePass` password. For example:

```

<Connector port="8443"
protocol="com.bluejungle.destiny.server.security.
secureConnector.SecurePasswordHttp11Protocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile = "C:\pdpcert\pdp-keystore.jks" keystoreType = "JKS"
keystorePass = "salf78f49e437288039751654ece96ede"/>

```

- On JBoss, edit the `standalone.xml` file to add a connector under the `"urn:jboss:domain:web:"` subsystem. That is, specify the values for: `certificate-key-file`, `password (raw)`, and `key-alias`. For example:

```

<subsystem xmlns='urn:jboss:domain:web:2.1' default-virtual-
server='default-host' native='false'>
<connector name='http' protocol='HTTP/1.1' scheme='http' socket-
binding='http'/>
<connector name="https" protocol="HTTP/1.1" scheme="https" socket-
binding="https" secure="true">
<ssl name="pdpssl" password="123next!" protocol="TLSv1" key-
alias="pdpssl" certificate-key-file="..../standalone/configuration/pdp-
keystore.jks" />
</connector>
...
</subsystem>

```

Configuring two-way SSL for Java Policy Controller

1. Create a KeyStore for the Java Policy Controller using a self-signed certificate or a certificate issued by a trusted authority. For example, pdp-keystore.jks.
2. Export the certificate from the newly-created KeyStore.
3. Import the exported certificate to the PEP TrustStore.
4. Copy the pdp-keystore.jks file to the PDP server.
5. Configure the PDP server to use SSL using the pdp-keystore.jks file.
 - On Tomcat, edit the `server.xml` file to add a connector under the `<Service name='CE-Core'>` section. That is, specify values for `keystoreFile` and `keystorePass` (the password value must be encrypted). For example:

```
<Connector port="8443"
protocol="com.bluejungle.destiny.server.security.
secureConnector.SecurePasswordHttp11Protocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile = "C:\pdpcert\pdp-keystore.jks" keystoreType = "JKS"
keystorePass = "sa1f78f49e437288039751654ece96ede"/>
```

- On JBoss, edit the `standalone.xml` file to add a connector under the `urn:jboss:domain:web: subsystem`. That is, specify the values for: `certificate-key-file`, `password` (password must be raw), and `key-alias`. For example:

```
<subsystem xmlns='urn:jboss:domain:web:2.1' default-virtual-
server='default-host' native='false'>
<connector name='http' protocol='HTTP/1.1' scheme='http' socket-
binding='http' />
<connector name="https" protocol="HTTP/1.1" scheme="https" socket-
binding="https" secure="true">
<ssl name="pdpssl" password="123next!" protocol="TLSv1" key-
alias="pdpssl" certificate-key-file="..../standalone/configuration/pdp-
keystore.jks" />
</connector>
...
</subsystem>
```

6. Either export or create a certificate for the PEP. For example, `client.cer`.
7. Create a TrustStore for the PDP and import the `client.cer` file into the PDP TrustStore. For example, `pdp-truststore.jks`.
8. Copy the `pdp-truststore.jks` file to the PDP server.
9. To configure the PDP server to use SSL using the `pdp-truststore.jks` file, perform one of the following steps:

- On Tomcat, edit the `server.xml` file to modify the connector under the `<Service name='CE-Core'>` section. That is, specify the values for `truststoreFile` and `truststorePass` (the password value must be encrypted). Change the value of `clientAuth` to `true`. For example:

```
<Connector port="8443"
protocol="com.bluejungle.destiny.server.security.
secureConnector.SecurePasswordHttp11Protocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="true" sslProtocol="TLS"
keystoreFile = "C:\pdpcert\pdp-keystore.jks" keystoreType = "JKS"
keystorePass = "sa1f78f49e437288039751654ece96ede" truststoreFile = "C:
\pdpcert\pdp-truststore.jks"
truststoreType = "JKS" truststorePass =
"sa1f78f49e437288039751654ece96ede" />
```

- On JBoss, edit the `standalone.xml` file to modify the connector under the `urn:jboss:domain:web: subsystem`. That is, specify the values for `ca-certificate-file` and `ca-password` (password must be raw). Change the value of `verify-client` to true. For example:

```
<subsystem xmlns='urn:jboss:domain:web:2.1'
default-virtual-server='default-host' native='false'>
<connector name='http' protocol='HTTP/1.1'
scheme='http' socket-binding='http'/>
<connector name="https" protocol="HTTP/1.1"
scheme="https" socket-binding="https" secure="true">
<ssl name="pdpssl" password="123next!" protocol="TLSv1"
key-alias="pdpssl" certificate-key-file="../standalone/configuration/pdp-
keystore.jks"
ca-certificate-file="C:/jboss-eap-6.4/standalone/configuration/pdp-
truststore.jks"
ca-certificate-password="123next!" verify-client = "true" />
</connector>
...
</subsystem>
```

Using the Java client library

This section explains how to use NextLabs OpenAz PEP Java client with remote PDPs (REST PDPs and CloudPDPs) and embedded PDPs.

Setting up the Java SDK

If you are developing Java PEPs, add the required JAR libraries to your Java project to set up the Java SDK.

- Set up the Java SDK with JDK 8 or higher. For instructions, go to https://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html.
- Obtain the following package from NextLabs: `Nextlabs-OpenAZ-PEP.zip`.
- Extract the files from the package.

 **Note:** Files are extracted with the Read-only attribute set in their properties. To view or change file properties in Windows, right-click the file, then select **Properties**.

- Add `nextlabs-openaz-pep.jar` and the required jar libraries, which are included in the `lib` folder, to your Java project.

Related concepts

[Invoking the PDP](#) on page 266

To invoke the PDP, you must configure the Java client SDK as described in this section.

Related tasks

[Setting up user authentication for the REST API](#) on page 154

To use the REST API with Java Policy Controllers, you need to set up user authentication.

[Configuring user authentication for the REST API](#) on page 283

The CloudAz REST API uses OAuth 2.0 for authentication. To help you get started quickly, the CloudAz instance comes preconfigured with `client_id` and `client_secret` credentials.

Invoking the PDP

To invoke the PDP, you must configure the Java client SDK as described in this section.

Related tasks

[Setting up the Java SDK](#) on page 266

If you are developing Java PEPs, add the required JAR libraries to your Java project to set up the Java SDK.

Configuring the Java client SDK to invoke remote or CloudAz PDPs

For remote or CloudAz PDPs, authorization requests are sent through a `PEPAgent`, which is created and configured using a `PEPAgentFactory`.

The `PEPAgentFactory` is configured with properties that can come from either a file or a `Properties` object. To use a file, create the file with the appropriate configuration for your system, then reference that file in the Java code. To use a `Properties` object, include the properties in the Java code directly.

- ! **Important:** NextLabs strongly recommends that you use HTTPS by specifying `https=true` for the REST Service.

1. Specify the `PDP Engine` name to initialize the PDP. NextLabs supports two types of `PDP Engines`: `EmbeddedPDPEngine` and `RestPDPEngine`. To use `EmbeddedPDPEngine`, specify only the `Embedded PDP class` in the properties file. Do not specify both the `EmbeddedPDPEngine` and `RestPDPEngine` class in the file. For example:

```
# PDP Engine configurations
# - Embedded PDP: com.nextlabs.openaz.pdp.EmbeddedPDPEngine
# - REST/ CloudAz PDP: com.nextlabs.openaz.pdp.RestPDPEngine
#-----
# PDPEngine class
# When using embedded PDP, set to
# "com.nextlabs.openaz.pdp.EmbeddedPDPEngine"
nextlabs.pdp.engine.name=com.nextlabs.openaz.pdp.EmbeddedPDPEngine
#
# When using embedded PDP configure the properties below
# Path to dpc folder: Embeded PDP requires the dpc folder path which
# includes all
# the resources required for embedded pdp
#-----
nextlabs.dpc.root=<path_to_embeddedPDP_dpc_folder>
```

2. To use the `RestPDPEngine`, configure the following properties:

```
# When using REST/ CloudAz PDP configure the properties below
# The host of CloudAz server e.g.: saas-jpc.cloudaz.com
#-----
nextlabs.cloudaz.host=<CloudAz REST API host>
# The port on which the CloudAz service is listening on the server
#-----
nextlabs.cloudaz.port=443
# Whether the CloudAz service is over https (true or false)
#-----
nextlabs.cloudaz.https=false
# The authentication settings to connect with the REST/CloudAz service
# Two authentication type are available to use
# - No authentication required: NONE
# - Use with OAuth2 provided authentication service: OAuth2
#-----
nextlabs.cloudaz.auth_type=OAUTH2
#
# OAuth2 Related configurations
# only if nextlabs.cloudaz.auth_type is OAUTH2
# The OAuth2 Authorization Grant Type
# available grant type is
# - client_credentials (default)
#-----
nextlabs.cloudaz.oauth2.grant_type=client_credentials
# The OAuth2 server configurations
```

```

# Default OAuth2 service is provided by Control Center server, in that
# case
# configure Control Center server host and port
#-----
nextlabs.cloudaz.oauth2.server=<control center host>
nextlabs.cloudaz.oauth2.port=<control center port>
nextlabs.cloudaz.oauth2.https=true
# Client ID to identify the client connecting using OAuth2
#-----
nextlabs.cloudaz.oauth2.client_id=<CLIENT_ID>
# Client secret for OAuth2 client credentials grant
#-----
nextlabs.cloudaz.oauth2.client_secret=<CLIENT_SECRET>
# OAuth endpoint to get the token for the client credential grant
# - CloudAz endpoint: /oauth/token
# - REST endpoint : /cas/token
#-----
nextlabs.cloudaz.oauth2.token_endpoint_path=/cas/token
# Ignore HTTPS self-signed certificates error if using self-signed
# certificates
#-----
nextlabs.cloudaz.ignore_https_certificate=true

```

3. To use a Properties object, include the properties in the Java PEP client code directly as follows:

 **Note:** As a good practice, use a Credential Store Framework to securely store and retrieve the `client_secret` key.

```

Properties xacmlProperties = new Properties();
// The host of CloudAz/ REST PDP server host
xacmlProperties.setProperty(Constants.PDP_REST_HOST, "<CloudAz REST API
host>");
// The OAuth2 server configurations
xacmlProperties.setProperty(Constants.PDP_REST_PORT, "443");
xacmlProperties.setProperty(Constants.PDP_REST_HTTPS, "true");
xacmlProperties.setProperty(Constants.PDP_REST_AUTH_TYPE, "OAUTH2");
xacmlProperties.setProperty(Constants.PDP_REST_OAUTH2_TOKEN_GRANT_TYPE,
"client_credentials");
// Client ID to identify the client connecting using OAuth2
xacmlProperties.setProperty(Constants.PDP_REST_OAUTH2_CLIENT_ID,
"<CLIENT_ID>");
// Client secret for OAuth2 client credentials grant
xacmlProperties.setProperty(Constants.PDP_REST_OAUTH2_CLIENT_SECRET,
"<CLIENT_SECRET>");
// OpenAz API PDP engine factory implementation
xacmlProperties.setProperty(XACMLProperties.PROP_PDPENGINEFACTORY,
"com.nextlabs.openaz.pdp.PDPEngineFactoryImpl");
//OpenAz API configuration
xacmlProperties.setProperty(Constants.ENGINE_NAME,
"com.nextlabs.openaz.pdp.RestPDPEngine");
// Mapper classes used internally to map requests
xacmlProperties.setProperty("pep.mapper.classes",
"com.nextlabs.openaz.pepapi.RecipientMapper,com.nextlabs.openaz.pepapi.
DiscretionaryPoliciesMapper,com.nextlabs.openaz.pepapi.HostMapper,com.
nextlabs.openaz.pepapi.ApplicationMapper");

```

4. To use a properties file, create the file with the appropriate configuration for your system, then reference that file in the Java code as follows:

```

import org.apache.openaz.pepapi.std.StdPepAgentFactory;
import org.apache.openaz.pepapi.PepAgent;
PepAgentFactory pepAgentFactory = new StdPepAgentFactory("/path/to/
properties.file");

```

```
PepAgent pepAgent = pepAgentFactory.getPepAgent();
```

Configuring the Java client to invoke embedded PDPs

You can configure Java client to invoke embedded PDPs as described in this section.

Obtain the following files from NextLabs Technical Support: Nextlabs-OpenAZ-PEP-<VersionNumber>.zip and PolicyControllerJava-<VersionNumber>.zip.

1. Extract the Nextlabs-OpenAZ-PEP-<VersionNumber>.zip archive.
2. Extract PolicyControllerJava-<VersionNumber>.zip archive and navigate to the embeddedpdp directory.
3. Configure the embeddedpdp directory:
 - a. In the extracted PolicyControllerJava-<VersionNumber>.zip archive, navigate to the embeddedpdp directory and copy the embeddedpdp folder to a convenient location.
 - b. In the embeddedpdp/config directory, modify the DABSLocation value in commprofile.xml to point to the ICENet server (host and port).
4. Configure the embeddedpdp parameters in Nextlabs-OpenAZ-PEP-<VersionNumber>:
 - a. Open the following file in a text editor: sample_code/config/openaz-pep.properties.
 - b. Replace the <path_to_embeddedPDP_dpc_folder> for your system and save the file. The embedded PDP requires the dpc folder path, which includes all the resources required by the embedded PDP:

```
#PDPEngine class, when using Embedded PDP to connect to server, set to
#"com.nextlabs.openaz.pdp.EmbeddedPDPEngine"
nextlabs.pdp.engine.name=com.nextlabs.openaz.pdp.EmbeddedPDPEngine
nextlabs.dpc.root=<path_to_embeddedPDP_dpc_folder>
```

5. Send the sample authorization request:

- a. Open a Windows Command prompt or Linux terminal, and navigate to the sample_code folder.
- b. Run one of the following scripts to send the request:
 - Windows: cloudaz_request.bat
 - Linux: cloudaz_request.sh

The response appears in the console.

Example properties file

```
-----#
# NEXTLABS OpenAZ PEP Properties  #
-----
# PDP Engine configurations
# - Embedded PDP: com.nextlabs.openaz.pdp.EmbeddedPDPEngine
# - REST/ CloudAz PDP: com.nextlabs.openaz.pdp.RestPDPEngine
#
# PDPEngine class, when using embedded PDP, set to #
"com.nextlabs.openaz.pdp.EmbeddedPDPEngine"
nextlabs.pdp.engine.name=com.nextlabs.openaz.pdp.EmbeddedPDPEngine
#
# When using Embedded PDP configure the properties below
# path to dpc folder: Embedded PDP requires dpc folder path which
# includes all
# the resources required for embedded PDP
#
nextlabs.dpc.root=<path_to_embeddedPDP_dpc_folder>
#
# When using REST/ CloudAz PDP configure the properties below
# The host of CloudAz server e.g.: saas-jpc.cloudaz.com
#-----
```

```

# nextlabs.cloudaz.host=<CloudAz REST API host>
# The port on which the Tomcat/JBoss service is listening on the
server
#-----
# nextlabs.cloudaz.port=443
# Whether the Tomcat/JBoss service is over https (true or false)
#-----
# nextlabs.cloudaz.https=false
# The authentication settings to connect with the REST/CloudAz
service
# Two authentication type are available to use
# - No authentication required: NONE
# - Use with OAuth2 provided authentication service: OAUTH2
#-----
# nextlabs.cloudaz.auth_type=OAUTH2
#
# OAuth2 Related configurations
# only if nextlabs.cloudaz.auth_type is OAUTH2
# The OAuth2 Authorization Grant Type
# available grant type is
# - client_credentials (default)
#-----
# nextlabs.cloudaz.oauth2.grant_type=client_credentials
# The OAuth2 server configurations
# Default OAuth2 service is provided by control center server, in
that case
# configure control center server host and port
#-----
# nextlabs.cloudaz.oauth2.server=<control center host>

# nextlabs.cloudaz.oauth2.port=<control center port>
# nextlabs.cloudaz.oauth2.https=true
# Client Id to identify the client connecting using OAuth2
#-----
# nextlabs.cloudaz.oauth2.client_id=<CLIENT_ID>
# Client secret for OAuth2 client credentials grant
#-----
# nextlabs.cloudaz.oauth2.client_secret=<CLIENT_SECRET>
# Oauth endpoint to get the token for the client credential grant
# - CloudAz endpoint: /oauth/token
# - REST endpoint : /cas/token
#-----
# nextlabs.cloudaz.oauth2.token_endpoint_path=/cas/token
# Ignore HTTPS self-signed certificates error if using self-signed
certificates

#-----
# nextlabs.cloudaz.ignore_https_certificate=true
# OpenAz API configuration
# no need to change this unless required
#-----
```

xacml.pdpEngineFactory=com.nextlabs.openaz.pdp.PDP EngineFactoryImpl\
Mapper classes used internally to map requests
#-----
pep.mapper.classes=com.nextlabs.openaz.pepapi.
RecipientMapper,com.nextlabs.openaz.pepapi.
DiscretionaryPoliciesMapper,com.nextlabs.
openaz.pepapi.HostMapper,com.nextlabs.
openaz.pepapi.ApplicationMapper

Configuring the embedded Policy Controller for Java on Tomcat servers

The embedded Policy Controller for Java is supported on Tomcat servers. This section explains how to set up the web application.

Obtain the following file from NextLabs Technical Support:

PolicyControllerJava-<VersionNumber>.zip

1. Extract the PolicyControllerJava-<VersionNumber>.zip and navigate to the embeddedpdp directory.
2. Place embeddedpdp.jar on the server where your custom enforcer is deployed.
3. Add the embeddedpdp.jar to your application classpath.
4. On the Tomcat server, navigate to tomcat/bin and modify the Catalina.bat file to point to the location of embeddedpdp.jar:
 - a. Find the following lines:

```
:noJuliManager  
set "JAVA_OPTS=%JAVA_OPTS% %LOGGING_MANAGER%"
```

- b. Modify the JAVA_OPTS line to add the path to embeddedpdp.jar. For example:

```
set "JAVA_OPTS=%JAVA_OPTS% %LOGGING_MANAGER% -Done  
jar.jar.path=C:\nextlabs\dependencies\embeddedpdp.jar"
```

5. Configure the embedded PDP directory:

- a. In the extracted PolicyControllerJava-<VersionNumber>.zip archive, navigate to nextlabs/embeddedpdp, and copy the embeddedpdp folder to a convenient location.
- b. In the embeddedpdp/config directory, modify the DABSLocation value in commprofile.xml to point to the ICENet server (host and port).

Related concepts

[Using embedded PDPs](#) on page 258

An alternative to installing standalone PDPs for Java is to embed the PDP in a web application.

Related tasks

[Configuring attribute plug-ins](#) on page 327

Attribute plug-ins can be configured for resource and subject attributes as needed.

Configuring the embedded Policy Controller for Java on JBoss servers

The embedded Policy Controller for Java is supported on JBoss servers. This section explains how to set up the web application.

Obtain the following file from NextLabs Technical Support:

PolicyControllerJava-<VersionNumber>.zip.

1. Extract the PolicyControllerJava-<VersionNumber>.zip and navigate to the embeddedpdp directory.
2. Place embeddedpdp.jar on the machine where the custom enforcer is deployed.
3. Modify the standalone.conf file to point to the location of embeddedpdp.jar:
 - a. Add the following line:

```
JAVA_OPTS="$JAVA_OPTS -Done-jar.jar.path=<path to embeddedPDP.jar>"
```

4. Configure the embedded PDP directory:

- a. In the extracted PolicyControllerJava-<VersionNumber>.zip archive, navigate to nextlabs/embeddedpdp, and copy the embeddedpdp folder to a convenient location.

- b. In the `embeddedpdp/config` directory, modify the `DABSLocation` value in `commprofile.xml` to point to the ICENet server (host and port).

Related tasks

[Configuring attribute plug-ins](#) on page 327

Attribute plug-ins can be configured for resource and subject attributes as needed.

Making authorization requests using Java

PEPs (policy enforcement points) sent authorization requests to PDPs (policy decision points) and receive authorization decisions in response. To make authorization requests using Java PEPs, use the NextLabs PEPAgent.

1. Initialize the PEPAgent and establish the PEPAgent secured connection with the PDP:

```
// Using a properties file
PepAgentFactory pepAgentFactory = new StdPepAgentFactory("/path/to/
properties.file");
PepAgent pepAgent = pepAgentFactory.getPepAgent();
// Using a properties object
PepAgentFactory pepAgentFactory = new StdPepAgentFactory(xacmlProperties);
PepAgent pepAgent = pepAgentFactory.getPepAgent();
```

2. Build the request:

- a. (Required) Build the Subject object with a unique ID. Typically this is a SID (Windows security identifier or UNIX ID) or an email address. Additional attributes can be populated using `addAttribute`.

```
Subject user = Subject.newInstance("chris.webber@hdesk.com");
user.addAttribute("user_id", "chris.webber");
```

- b. (Required) Build the Action object using the Action short name. The short name must match that of the Action resource type.

```
Action action = Action.newInstance("VIEW_TKTS");
```

- c. (Required) Build a Resource object with a resource ID, which is typically the name of the resource, and populate all the available resource attributes. `ID_RESOURCE_RESOURCE_TYPE` is a required attribute. The value of this attribute must match the short name of the resource type:

```
Resource resource = Resource.newInstance("Ticket:1103");
resource.addAttribute(Constants.ID_RESOURCE_RESOURCE_TYPE.stringValue(),
"support_tickets");
```

- d. (Required) Build an Application object:

```
Application application = Application.newInstance("Outlook");
```

- e. (Optional) Build an Environment object with environment attributes:

```
Environment environment = Environment.newInstance();
environment.addAttribute("authentication_type", "multifactor");
```

- f. (Optional) If you are enforcing policies on an application that sends email, such as Microsoft Outlook, build the Recipient object. The Recipient object can include a single email address or a list of email addresses. However, attributes can be associated with a single recipient only. If your policies depend on attributes, and you need to specify multiple recipients, each recipient must be placed in a separate request:

```
Recipient recipient = Recipient.newInstance
("frank.underwood@whitehouse.gov",
```

```

"claire.underwood@whitehouse.gov", "doug.stamper@whitehouse.gov");
Recipient recipient = Recipient.newInstance
    ("frank.underwood@whitehouse.gov");
recipient.addAttribute("title", "President");

```

3. Send the authorization request from the PEPAgent using the decide() method:

 **Note:** The order of arguments (user, action, resource, environment, and recipient) does not matter. Arguments can be specified in any order. Also, only subject, action, and resource are required.

```

// Submit the authorization request
PepResponse pepResponse = pepAgent.decide(subject, action, resource,
    application, environment, recipient);

```

Exception handling

Based on OpenAz specification, `result.allowed()` supports only two responses: Allow and Deny.

If the PDP evaluation does not match the request to any of the system policies, it raises a `PepException`. The suggested behavior for handling this exception is for the PEP to Deny the request. This ensures the application is secured by default if there is no policy explicitly allowing access.

Table 54: Exception handling

Language	Exception example
Java	<pre> catch(error) { effectValue = "Deny"; } </pre>
JavaScript	<pre> /* * pepResponse.allowed() returns a boolean value. * If the * decision is allowed it returns true; otherwise * it returns * false for deny * * The above method call throws a PepException for * all other * indeterminate states */ String effectValue = ""; try { effectValue = pepResponse.allowed() ? "Allow" : "Deny"; } catch (PepException ex) { effectValue = "Indeterminate"; } </pre>

SDK quick reference

This section provides a quick reference for the OpenAz PEP SDK.

Methods

Use the following methods when making authorization requests.

Table 55: Methods used in authorization requests

Method	Description and example
newInstance	Used to create an instance of a category class, such as Subject.
Java example	<pre>Subject user = Subject.newInstance("chris.webber@hdesk.com");</pre>
addAttribute	Used to add attributes to subject, resource, environment, and recipient category objects.
Java example	<pre>Subject user = Subject.newInstance("chris.webber@hdesk.com"); user.addAttribute("user_id", "chris.webber")</pre>
JavaScript example	<pre>var user = new Subject("chris.webber@hdesk.com"); user.addAttribute("department", "IT", "Development");</pre>
simpleDecide	Used to send requests to the server for testing and development. This method takes a user ID, an action, and a resource ID. However, you cannot pass user or resource attributes, discretionary policies, environment details, and so on using this method.
Java example	<pre>PepResponse pepResponse = pepAgent.simpleDecide("Frank Underwood", "OPEN", "Document.docx");</pre>
JavaScript example	<pre>pepAgent.simpleDecide("Frank Underwood", "OPEN", "Document.docx").then(function(pepResponse) { assert(pepResponse instanceof PEPResponse); });</pre>
decide	Used to send a single authorization request to the server, which expects a dynamic list of categories, and returns a promise that resolves to a single PEPResponse object. This method is more complex than simpleDecide() because it requires the construction of user and resource and action objects, rather than IDs alone. With that complexity, however, comes flexibility and power. This method enables you to send information such as user or resource attributes, environment information, destination resource information, host information, and discretionary policies with the request.
Java example	<pre>PepResponse pepResponse = pepAgent.decide(user, action, resource, environment);</pre>
JavaScript example	<pre>var promise = agent.decide(subject, resource, action, application); promise.then(function(result) { // result is an object of PEPResponse // TODO process the response });</pre>

Method	Description and example
bulkDecide()	<p>Used to send multiple authorization requests to the server in a single call or payload. For example, a web page might have several restricted items. When a user visits the page, the PEP can evaluate the authorization request for all of those items in a single call. This might or might not be more efficient than <code>decide()</code>, depending on the number of queries, the speed of the network, the cost of evaluating policies, and so on, but it is conceptually cleaner. For single authorization requests, however, use <code>decide()</code>.</p> <p>The server expects the first argument to be an array of objects of the multiple category (called associations), followed by the rest of the category objects. Returns a promise that resolves to an array of <code>PEPResponse</code> objects following the order of the associations.</p>
Java example	<pre data-bbox="657 656 1432 857"> List<Resource> resources = new ArrayList<Resource>(); resources.add(resource1); resources.add(resource2); List<PepResponse> responses = pepAgent.bulkDecide(resources, subject, action, host, application, recipient, environment); </pre>
JavaScript example	<pre data-bbox="657 910 1432 1174"> var promise = agent.bulkDecide([resource, resource1], subject, action, application); promise.then(function(result) { result.forEach(function(r, index) { // r is an object of PEPResponse // TODO process response }); }); </pre>

Request category classes

Request category classes represent components in an authorization request, such as the subject, action, and resource.

Instances of such classes may be passed to the `decide` and `bulkDecide` methods of the `PEPAgent`. The `simpleDecide` method cannot be used to pass subject or resource attributes.

Request category classes include:

- Subclasses of the `org.apache.openaz.pepapi.CategoryContainer`
 - `org.apache.openaz.pepapi.Subject`
 - `org.apache.openaz.pepapi.Action`
 - `org.apache.openaz.pepapi.Resource`
 - `org.apache.openaz.pepapi.Environment`
- NextLabs specific category classes include:
 - `com.nextlabs.openaz.pepapi.Application`
 - `com.nextlabs.openaz.pepapi.Host`
 - `com.nextlabs.openaz.pepapi.Recipient`

Table 56: Request category classes

Category class & Description code example		Data type/ valid input	Required/optional
Subject	The individual performing the action. Typically, an a SID (Windows security identifier or UNIX ID) or an email address.	String	Required
Java example	<pre>Subject user = Subject.newInstance("chris.webber@hdesk.com"); user.addAttribute("user_id", "chris.webber");</pre>		
JavaScript example	<pre>var Subject = nextlabsopenaz.Subject; var user = new Subject("chris.webber@hdesk.com"); user.addAttribute("user_id", "chris.webber");</pre>		
Action	The short name of the action, such as edit, view, or copy, for which access is requested. The short name must match that of the Action resource type.	String	Required
Java example	<pre>Action action = Action.newInstance("VIEW_TKTS");</pre>		
JavaScript example	<pre>var Action = nextlabsopenaz.Action; var action = new Action("VIEW_TKTS");</pre>		
Resource	The document or other resource on which the action is being performed. The following attribute is required: ID_RESOURCE_RESOURCE_TYPE. The value of this attribute must match the short name of the resource type.	String	Required;
Java example	<pre>Resource resource = Resource.newInstance("Ticket:1103"); resource.addAttribute(Constants. ID_RESOURCE_RESOURCE_TYPE.stringValue(), "support_tickets");</pre>		
JavaScript example	<pre>var Resource = nextlabsopenaz.Resource; var resource = new Resource("Ticket:1103"); resource.addAttribute(NextLabsXACML. ID_RESOURCE_RESOURCE_TYPE, "support_tickets") resource.addAttribute("ticket_id", "1103");</pre>		
Application	The application used to access the resource and perform the actions.	String	Required for JavaScript only; not used in Java
JavaScript example	<pre>var Application = nextlabsopenaz.Application; var application = new Application("Payroll App");</pre>		

Category class & Description code example	Data type/ Required/optional valid input
Environment The conditions, such as location or computer operating system, under which the action is performed.	String Optional
Java example <pre>Environment environment = Environment.newInstance(); environment.addAttribute("authentication_type", "multifactor");</pre>	
JavaScript example <pre>var Environment = nextlabsopenaz.Environment; var environment = new Environment(); environment.addAttribute("authentication_type", "multifactor");</pre>	
Recipient A single email address with attributes or a list of email addresses; Used with email enforcers such as NextLabs Outlook Enforcer. Attributes can be associated with a single recipient only. If your policies depend on attributes, and you need to specify multiple recipients, each recipient must be placed in a separate request.	String; email address Optional
Java example <pre>Recipient recipient = Recipient.newInstance ("frank.underwood@whitehouse.gov", "claire.underwood@whitehouse.gov", "doug.stamper@whitehouse.gov"); Recipient recip = Recipient.newInstance ("frank.underwood@whitehouse.gov"); recip.addAttribute("title", "President");</pre>	
JavaScript example <pre>var Recipient = nextlabsopenaz.Recipient; var recipient = new Recipient("frank.underwood@whitehouse.gov"); recipient.addAttribute("title", "President");</pre>	
Host The hostname or IP address of the machine on which the action is being performed.	String or Integer Optional
Java example <pre>Host host = Host.newInstance("mymachine.mycompany.com");</pre>	
JavaScript example <pre>// default localhost var host = new Host() // using host name var host1 = new Host("some_host") // using inet address var host2 = new Host(0x7f000001);</pre>	

PEPResponse

The Cloud PDP responds to authorization requests by returning the `PEPResponse` object.

This object includes the evaluation decision and any obligations that the PEP should attempt to perform when it enforces the decision.

Table 57: PEPResponse

Response object	Description
allowed	Returns a value representing the authorization decision: <ul style="list-style-type: none">• true: Returns the decision, Allow.• false: Returns the decision, Deny.• indeterminate or invalid: Returns the decision, Not Applicable or <code>PEPEException</code>.
getObligations	Returns the obligations (instructions for performing additional tasks) required by the policy being enforced.
Java example	Returns the set of <code>org.apache.openaz.pepapi.Obligations</code> associated with the current result indexed by <code>ObligationId</code> .
JavaScript example	Returns a map containing the obligations returned by the Cloud PDP: <pre>var promise = agent.decide(subject, resource, action, application); promise.then(function(result) { var obligations = result.getObligations(); obligations.forEach(function(obl, id) { // obl is an object of Obligation // TODO process obligation }); })</pre>
getWrappedResult	Returns the raw object representing the response. This exposes the helper methods to access the evaluation response details.
Java example	<pre>Decision decision = pepResponse.getWrappedResult().getDecision();</pre>
JavaScript example	<pre>var decision = result.getWrappedResult().Decision;</pre>
Obligation	Instructions for performing additional required tasks, which are sent by the PDP (Policy Decision Point) to the PEP (Policy Enforcement Point) along with the authorization decision.
getId	Returns the obligation ID.
Java example	<pre>String obligationId = obligation.getId();</pre>
JavaScript example	<pre>var obligationId = obligation.getId();</pre>

Response object	Description
getAttributeMap	Returns a map of the obligation attribute name, object-value-array pairs, indexed by name, where name is the Attributeld and the value is an array of one or more object values of the attribute. An array with a length greater than one indicates a multi-value attribute.
Java example	<pre> /* * The following section prints all the obligations * and their * attributes if any obligations are * available for the * authorization request decision */ if (!pepResponse.getObligations().isEmpty()) { System.out.println(" Obligations: "); Iterator<Entry<String, Obligation>> obligationEntryIter = pepResponse.getObligations().entrySet() .iterator(); while (obligationEntryIter.hasNext()) { Entry<String, Obligation> obligationEntry = obligationEntryIter.next(); System.out.println(" { " + obligationEntry.getKey() + " : ["); /* * Print each obligation's attributes as key values */ Iterator<Entry<String, Object[]>> attrMapIter = obligationEntry.getValue().getAttributeMap(). entrySet().iterator(); while (attrMapIter.hasNext()) { Entry<String, Object[]> attrMapEntry = attrMapIter.next(); System.out.print(" Key: " + attrMapEntry.getKey() + ", values: "); for (int j = 0; j < attrMapEntry.getValue().length; j++) { System.out.print("\\" + attrMapEntry.getValue()[j] + "\\""); if (j < attrMapEntry.getValue().length - 1) { System.out.print(", "); } } if (attrMapIter.hasNext()) { System.out.println(","); } if (obligationEntryIter.hasNext()) { System.out.println(","); } } } } } </pre>
JavaScript example	<pre> var obligations = result.getObligations(); obligations.forEach(function(obligation, key) { var attributes = obligation.getAttributeMap(); /* * Print each obligation's attributes as key values */ attributes.forEach(function(values, name) { console.log(" " + name + ": " + values); }); var ticketId = attributes.get('ticket_id')[0]; var assignedTo = attributes.get('assigned_to')[0]; var processedMsg = String(attributes.get('message') [0]).replace("\${ticket_id}", ticketId). replace("\${assigned_to}", assignedTo); console.log(" updated message: " + "</pre>

Using the JavaScript client library

This section explains how to use NextLabs OpenAz PEP JavaScript client to construct and send an authorization request to NextLabs REST API and how to process its response.

Setting up the JavaScript SDK

To use the JavaScript client library, set up the JavaScript SDK.

1. Set up the JavaScript SDK. For instructions, go to <https://nodejs.org/en/download/>.
2. Verify that the JavaScript node version is 4.5 or higher, and the npm version is 3.10.6 or higher. To check the versions, use these commands:

```
node --version  
npm --version
```

3. Obtain the following package from NextLabs: `Nextlabs-OpenAZ-PEP.zip`.
4. Extract the files from the package.
5. Open the extracted folder and go to the `js` folder, which contains the required files and samples for the NodeJS API.
 -  **Note:** Files are extracted with the Read-only attribute set in their properties. To view or change file properties in Windows, right-click the file, then select **Properties**.
6. After cloning the project, specify NextLabs OpenAz as a dependency in `package.json` using local path:

```
"dependencies": {  
  "nextlabs-openaz": "path/to/project",  
  ...other dependencies...  
}
```

7. To install the module:

```
npm install
```

8. To use the module:

```
var nextlabsopenaz = require("nextlabs-openaz");
```

Load required properties.

Related concepts

[Required properties on page 281](#)

Properties can be loaded through `openAz-pep.json` or as a JavaScript object. The following properties are required to connect to the REST API.

Related tasks

[Setting up user authentication for the REST API on page 154](#)

To use the REST API with Java Policy Controllers, you need to set up user authentication.

[Making authorization requests using JavaScript on page 281](#)

PEPs (policy enforcement points) send authorization requests to PDPs (policy decision points) and receive authorization decisions in response.

[Configuring user authentication for the REST API on page 283](#)

The CloudAz REST API uses OAuth 2.0 for authentication. To help you get started quickly, the CloudAz instance comes preconfigured with `client_id` and `client_secret` credentials.

Required properties

Properties can be loaded through `openAz-pep.json` or as a JavaScript object. The following properties are required to connect to the REST API.

- `nextlabs.cloudaz.host`: The host name of the server where the NextLabs Policy Decision Point is located. For example, `jpc.localdomain`.
- `nextlabs.cloudaz.port`: The port used by Tomcat or JBoss for REST API connections. For example, `58080`.
- `nextlabs.cloudaz.https`: The protocol used for REST API connections. For HTTPS use `false`.
- `nextlabs.cloudaz.auth_type`: The authorization method for connecting to the REST API. For example, `OAUTH2`.
- `nextlabs.cloudaz.oauth2.grant_type`: The authentication grant type for REST API connections. For example, `client_credentials`.
- `nextlabs.cloudaz.oauth2.client_id`: The `client_id` credential required for authentication with the application. This is the `username` of the account created for the connection.
- `nextlabs.cloudaz.oauth2.client_secret`: The `client_secret` credential required for authentication with the application. This is the `password` of the account created for the connection.
- `nextlabs.cloudaz.ignore_https_certificate`: Whether to ignore the HTTPS certificate warnings (`true`) or not (`false`). If you are using a self-signed SSL certificate, use the value `true`.
- `nextlabs.cloudaz.oauth2.server`: The hostname of the server where the Control Center is installed. For example, `<Control_Center_host>`.
- `nextlabs.cloudaz.oauth2.port`: The port to use for REST API connections. For HTTPS, use `443`.
- `nextlabs.cloudaz.oauth2.https`: The HTTPS connection for the REST API. For HTTPS, use `true`.
- `nextlabs.cloudaz.oauth2.token_endpoint_path`: The Control Center host name or IP address and the authentication token. For example, `<Control_Center_host>/cas/token`.

Example

```
{  
  "nextlabs.cloudaz.host": "<Java PC host> e.g.: jpc.localdomain",  
  "nextlabs.cloudaz.port": "58080",  
  "nextlabs.cloudaz.https": false,  
  "nextlabs.cloudaz.auth_type": "OAUTH2",  
  "nextlabs.cloudaz.oauth2.grant_type": "client_credentials",  
  "nextlabs.cloudaz.oauth2.client_id": "<CLIENT_ID>",  
  "nextlabs.cloudaz.oauth2.client_secret": "<CLIENT_SECRET>",  
  "nextlabs.cloudaz.ignore_https_certificate": true,  
  "nextlabs.cloudaz.oauth2.server": "<Control_Center_Host>",  
  "nextlabs.cloudaz.oauth2.port": "443",  
  "nextlabs.cloudaz.oauth2.https":  
    true, "nextlabs.cloudaz.oauth2.token_endpoint_path": "/cas/token"  
};
```

Related tasks

[Setting up the JavaScript SDK](#) on page 280

To use the JavaScript client library, set up the JavaScript SDK.

[Configuring user authentication for the REST API](#) on page 283

The CloudAz REST API uses OAuth 2.0 for authentication. To help you get started quickly, the CloudAz instance comes preconfigured with `client_id` and `client_secret` credentials.

[Making authorization requests using JavaScript](#) on page 281

PEPs (policy enforcement points) send authorization requests to PDPs (policy decision points) and receive authorization decisions in response.

Making authorization requests using JavaScript

PEPs (policy enforcement points) send authorization requests to PDPs (policy decision points) and receive authorization decisions in response.

To make an authorization request using JavaScript, you need to load properties, build the request, and send the request to the PDP from the NextLabs PEPAgent, an instance of which is created with the new operator.

1. Load the required properties using a JavaScript object or a JSON file:

To load properties using a JavaScript object:

```
var pepAgent = new NextLabsPEPAgent(openazProperties);
```

To load properties using a JSON file and establish a secure connection:

```
var openazProperties = require("./config/openaz-pep.json");
```

2. Initialize the PEPAgent and establish the PEPAgent secured connection with the PDP:

```
var nextlabsopenaz = require("nextlabs-openaz");
var NextLabsPEPAgent = nextlabsopenaz.NextLabsPEPAgent;
var openazProperties = require("./config/openaz-pep.json");
var pepAgent = new NextLabsPEPAgent(openazProperties);
```

3. Build the request:

- a. (Required) Build the Subject object with a unique ID. Typically this is a SID (Windows security identifier or UNIX ID) or an email address. Additional attributes can be populated using addAttribute:

```
var Subject = nextlabsopenaz.Subject;
var user = new Subject("chris.webber@hdesk.com");
user.addAttribute("user_id", "chris.webber");
```

- b. (Required) Build the Action object using the Action short name. The short name must match that of the Action resource type:

```
var Action = nextlabsopenaz.Action;
var action = new Action("VIEW_TKTS");
```

- c. (Required) Build a Resource object with a resource ID, which is typically the name of the resource, and populate all the available resource attributes. ID_RESOURCE_RESOURCE_TYPE is a required attribute. The value of this attribute must match the short name of the resource type:

```
var Resource = nextlabsopenaz.Resource;
var resource = new Resource("Ticket:1103");
resource.addAttribute(NextLabsXACML.ID_RESOURCE_RESOURCE_TYPE,
"support_tickets")
resource.addAttribute("ticket_id", "1103");
```

- d. (Required) Build the Application object and its attributes:

```
var Application = nextlabsopenaz.Application;
var application = new Application("HelpDeskApp");
application.addAttribute("version", "1.0");
```

- e. (Optional) Build an Environment object and its attributes:

```
var Environment = nextlabsopenaz.Environment;
var environment = new Environment();
environment.addAttribute("authentication_type", "multifactor");
```

- f. (Optional) Specify the host or the machine that processes authorization requests. To create an instance of Host, pass an integer as an IP address or a string as a hostname, whichever is most convenient. No additional attributes are supported:

```
// default localhost
var host = new Host()
// using host name
var host1 = new Host("some_host");
// using inet address
var host2 = new Host(0x7f000001);
```

- g. (Optional) If you are enforcing policies on an application that sends email, such as Microsoft Outlook, build the Recipient object. The Recipient object can include a single email address or a list of email addresses. However, attributes can be associated with a single recipient only. If your policies depend on attributes, and you need to specify multiple recipients, each recipient must be placed in a separate request:

```
// single recipient with id and email
var recipient = new Recipient("abc@gmail.com", "abc")
// multiple recipients with email addresses
var recipient2 = new Recipient("abc@gmail.com", "def@gmail.com");
```

4. Send the authorization request from the PEPAgent using the decide() method:

```
// Include the details required for the decide method: subject, action,
// resource, application
// Include optional details as needed: environment, host, recipient
// pepAgent.decide(String subject, String action, String resource,
// String environment, String application, String recipient)
pepAgent.decide(subject, action, resource, application, environment, host,
    recipient).then(function(pepResponse) {
    assert(pepResponse instanceof PEPResponse);
});
```

Related concepts

[Required properties](#) on page 281

Properties can be loaded through `openAz-pep.json` or as a JavaScript object. The following properties are required to connect to the REST API.

Related tasks

[Setting up the JavaScript SDK](#) on page 280

To use the JavaScript client library, set up the JavaScript SDK.

Configuring user authentication for the REST API

The CloudAz REST API uses OAuth 2.0 for authentication. To help you get started quickly, the CloudAz instance comes preconfigured with `client_id` and `client_secret` credentials.

You can add additional credentials to your instance and change authentication settings as described in this section.

1. Set authentication properties:

- For the Java PEP SDK, specify the following properties in the `openaz-pep.properties` file:

```
#-----
#                               NEXTLABS OpenAZ PEP Properties
#
#-----
```

```

# PDP Engine configurations
#   - Embedded PDP:      com.nextlabs.openaz.pdp.EmbeddedPDP Engine
#   - REST/ CloudAZ PDP: com.nextlabs.openaz.pdp.RestPDP Engine
#-----
# PDP Engine class, when using embedded PDP, set to
# "com.nextlabs.openaz.pdp.EmbeddedPDP Engine"
nextlabs.pdp.engine.name=com.nextlabs.openaz.pdp.EmbeddedPDP Engine

#
# When using Embedded PDP configure the properties below

# path to dpc folder: Embedded PDP requires dpc folder path which
# includes all
# the resources required for embedded pdp
#-----
# E.g. For Windows Environment: nextlabs.dpc.root=C:/
nextlabs_embedded_pdp/dpc
#                               or nextlabs.dpc.root=C:\nextlabs_embedded_pdp\dpc
# For Linux Environment:       nextlabs.dpc.root=/home/
nextlabs_embedded_pdp/dpc
# nextlabs.dpc.root=<path_to_embeddedPDP_dpc_folder>

# Payload type. json or xml
#-----
#nextlabs.cloudaz.payload_content_type=application/xml
nextlabs.cloudaz.payload_content_type=application/json

#
# When using REST/ CloudAz PDP configure the properties below

# The host of CloudAz server eg: saas-jpc.cloudaz.com
#-----
nextlabs.cloudaz.host=<CloudAz REST API host>

# The port on which the CloudAz service is listening on the server
#-----
nextlabs.cloudaz.port=443

# Whether the CloudAz service is over https (true or false)
#-----
nextlabs.cloudaz.https=true

# The authentication settings to connect with the REST/CloudAz service
# Two authentication type are available to use
#   - No authentication required: NONE
#   - Use with OAuth2 provided authentication service: OAUTH2
#-----
nextlabs.cloudaz.auth_type=OAUTH2

#
# OAuth2 Related configurations
#   only if nextlabs.cloudaz.auth_type is OAUTH2

# The OAuth2 Authorization Grant Type
#   available grant type is
#     - client_credentials (default)
#-----
nextlabs.cloudaz.oauth2.grant_type=client_credentials

# The OAuth2 server configurations
#   Default oauth2 service is provided by control center server, in that
#   case

```

```

# configure control center server host and port
#-----
nextlabs.cloudaz.oauth2.server=<control center host>
nextlabs.cloudaz.oauth2.port=<control center port>
nextlabs.cloudaz.oauth2.https=true

# Client Id to identify the client connecting using Oauth2
#-----
nextlabs.cloudaz.oauth2.client_id=<CLIENT_ID>

# Client secret for Oauth2 client credentials grant
#-----
nextlabs.cloudaz.oauth2.client_secret=<CLIENT_SECRET>

# Oauth endpoint to get the token for the client credential grant
#   - CloudAZ endpoint: /oauth/token
#   - REST endpoint : /cas/token
#-----
nextlabs.cloudaz.oauth2.token_endpoint_path=/cas/token

# Ignore HTTPS self signed certificates error, if using self signed
# certificates
#-----
nextlabs.cloudaz.ignore_https_certificate=true

# OpenAZ api configuration
#   no need to change this unless required
#-----
xacml.pdpEngineFactory=com.nextlabs.openaz.pdp.PDP EngineFactoryImpl

# Mapper classes used internally to map requests
#-----
pep.mapper.classes=com.nextlabs.openaz.pepapi.RecipientMapper,com
.nextlabs.openaz.pepapi.DiscretionaryPoliciesMapper,com.
nextlabs.openaz.pepapi.HostMapper,com.
nextlabs.openaz.pepapi.ApplicationMapper

```

- For the JavaScript PEP SDK, specify the following properties in the openaz-pep.json file:

```
{
"nextlabs.cloudaz.host": "<Java PC host> eg: jpc.localdomain",
"nextlabs.cloudaz.port": "58080",
"nextlabs.cloudaz.https": false,
"nextlabs.cloudaz.auth_type": "OAUTH2",
"nextlabs.cloudaz.oauth2.grant_type": "client_credentials",
"nextlabs.cloudaz.oauth2.client_id": "<CLIENT_ID>",
"nextlabs.cloudaz.oauth2.client_secret": "<CLIENT_SECRET>",
"nextlabs.cloudaz.ignore_https_certificate": true,
"nextlabs.cloudaz.oauth2.server": "<Control Center Host>",
"nextlabs.cloudaz.oauth2.port": "443",
"nextlabs.cloudaz.oauth2.https": true,
"nextlabs.cloudaz.oauth2.token_endpoint_path": "/cas/token"
}
```

- In the CloudAz console, create a user account for the PEP client with the following settings:

- Username = The `client_id` specified in the properties file.
- Password = The `client_secret` specified in the properties file.
- Add the user attribute `jwt_passphrase`. The value of this attribute can be anything except for the password (`client_secret`).

- After the account has been created, log in to the CloudAz console using the account and supply the appropriate `client_secret` password. This is the password specified in the properties file. For

security, all users must set their passwords when they log in to their CloudAz console accounts for the first time.

Related concepts

[Required properties](#) on page 281

Properties can be loaded through `openAz-pep.json` or as a JavaScript object. The following properties are required to connect to the REST API.

Related tasks

[Setting up the Java SDK](#) on page 266

If you are developing Java PEPs, add the required JAR libraries to your Java project to set up the Java SDK.

[Setting up the JavaScript SDK](#) on page 280

To use the JavaScript client library, set up the JavaScript SDK.

REST API for external PDPs

This section describes how to use the REST API to send policy evaluation requests to external PDPs (Policy Decision Points).

About the REST API

The REST API is a service that enables a PEP (Policy Enforcement Point) to send policy evaluation requests in XACML to external PDPs (Policy Decision Points), and to receive policy decisions.

Requests and responses are sent using HTTPS with mutual authentication. The following figure illustrates the flow of REST API requests and responses.

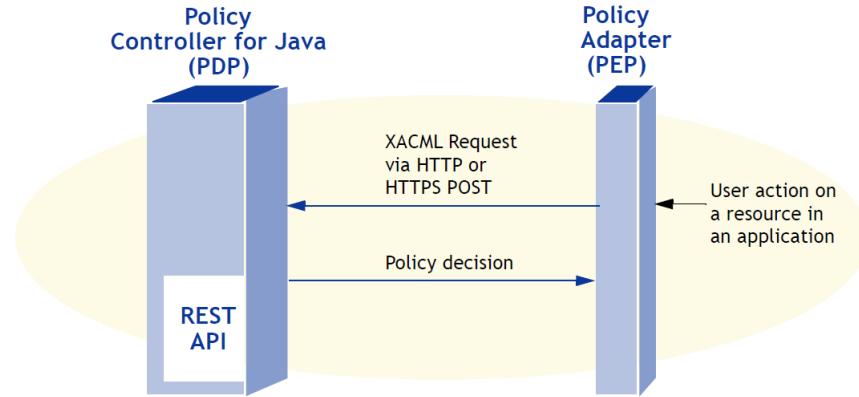


Figure 98: Standard message flow using the REST API

How to set up the REST API

To use the REST API, the Java Policy Controller must be installed and configured. For more information, see the NextLabs Control Center Installation and Upgrade Guide.

Related concepts

[Policy Controller configuration](#) on page 150

[PEP development process](#) on page 259

Building PEPs involves these steps.

Getting help

To access REST API reference information, with sample request and response payloads, go to the following URL.

`http://<host>:>58080>/dpc/authorization/pdp/help`

How to access the REST API

The URL for making requests is as follows.

`http://<host>:<port>/dpc/authorization/pdp`

`https://<host>:<SSL-enabled port>/dpc/authorization/pdp`

Using OAuth 2.0 to access REST APIs

The PDPs protect XACML REST APIs using two-legged OAuth.

A Servlet filter verifies that HTTP Headers have valid `Authorization` values before allowing clients to call the APIs. Authorization values use the OAuth 2.0 Bearer token.

Example

```
Authorization : Bearer <access_token>
```

The following figure shows how the PDP processes authorization requests.

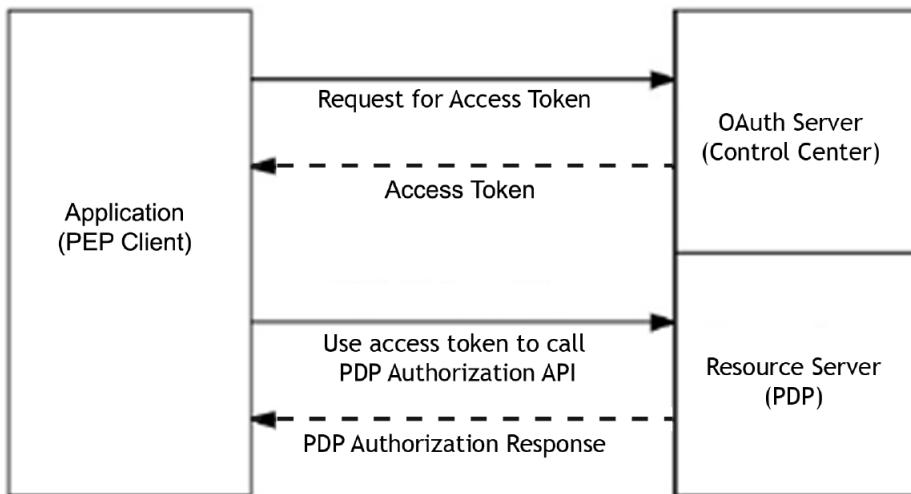


Figure 99: How authorization requests are processed

Obtain an access token

To obtain an access token, make a call to the PDP, which functions as the Authorization Server.

About authorization grant types requests, and responses

The system supports the Client Credentials authorization grant type.

Request endpoint

The following endpoint is used to make an access token request.

`https://<Control Center Host>/cas/token`

Access token request

Clients make requests to token endpoints by adding the following required and optional parameters using `application/x-www-form-urlencoded` with a character encoding of UTF-8 in the HTTP request entity-body:

- `grant_type` REQUIRED. Value MUST be set to `client_credentials`.
- `client_id` REQUIRED. The API Client ID.
- `client_secret` REQUIRED. The API Client Secret.
- `expires_in` OPTIONAL. The token returned expires in the seconds specified. Users can request a token valid for a maximum of one year, which is $3600 * 24 * 365$ seconds. If omitted, the default value, 3600 is used.

Example:

```
POST /oauth/token HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
grant_type=client_credentials&client_id=apiclient&client_secret=ABCDEFGABCDEF
```

Access token response

If the access token request is valid and authorized, the authorization server issues an access token.

Example

```
HTTP/1.1 200 OK
{
  "access_token":"2YotnFZFEjr1zCsicMWpAA",
  "token_type":"Bearer",
  "expires_in":3600,
}
```

Using an access token to access the XACML evaluation service

After obtaining an access token, clients can use that token to call the PDP XACML evaluation API.

Example

```
POST /dpc/authorization/pdp HTTP/1.1
Host: server.example.com
Authorization: Bearer 2YotnFZFEjr1zCsicMWpAA
... Other headers and payload
```

Types of requests

PEPs can send the following types of policy evaluation requests to the Policy Controller.

- Single policy evaluation query in a single request: Evaluate an action that a user has performed on a document resource.
- Multiple policy evaluation queries in a single request: Evaluate an action that a user has performed on multiple document resources.
- Ad hoc PQL policy requests, also called policy on-demand (POD) requests: Evaluate actions in conjunction with, or instead of, a deployed policy.
- SAML requests: Exchange authentication and authorization data between identity providers and service providers. The following example shows a SAML request.

About ad hoc or policy-on-demand (POD) policy requests

The REST API extends the XACML 3.0 specification to support ad hoc or policy-on-demand (POD) policy requests.

These requests are useful when you need to control access to documents using attributes and values that are not part of deployed policies. For example, a policy might restrict the sharing of documents to a specific location, such as the city of Seattle. If a Seattle team member moves to a different location, such as Kansas City, an ad hoc policy request could be used to grant that access.

To send an ad hoc or POD request, use the following category ID and attributes:

```
CategoryID: pod
AttributeID: pod-id
  Value: <Policy written in PQL>
AttributeID: pod-ignore-built-in
  Value: "true" or "false".
    - True applies this policy instead of the deployed policies.
    - False applies this policy in conjunction with deployed policies.
```

Request format

The REST API supports policy evaluation requests that conform to XACML 3.0 and are passed in XML or JSON format.

Requests are sent using the POST method of HTTPS. The parameters are:

```
Service="EVAL"  
DataType=<xml or json>  
Data=<XACML request data in JSON or XML>  
Version="1.0"
```

The XACML request data passed to the Cloud PDP contains event details, such as the subject (user), action, resource, and application.

For information about the XACML 3.0 standard, go to <http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf>.

Related concepts

[Examples of request data](#) on page 289

This section provides the following examples of request data.

Examples of request data

This section provides the following examples of request data.

Related concepts

[Request format](#) on page 289

The REST API supports policy evaluation requests that conform to XACML 3.0 and are passed in XML or JSON format.

Single-query request (XML): Example 1

This request data contains an action (EDIT_TKTS) that a subject (chris.webber) has performed in an application (Helpdesk).

Sample request payload: Example 1

```
{
  "Request": {
    "ReturnPolicyIdList": "true", "Category": [
      {"CategoryId": "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject",
       "Attribute": [
         {"AttributeId": "urn:oasis:names:tc:xacml:1.0:subject:subject-id",
          "Value": "chris.webber@hdesk.com",
          "IncludeInResult": "false"
        },
        {"AttributeId": "urn:oasis:names:tc:xacml:1.0:subject:assigned_prod_area",
          "Value": "Exchange Email",
          "IncludeInResult": "false"
        },
        {"AttributeId": "urn:oasis:names:tc:xacml:1.0:subject:department",
          "Value": "IT",
          "IncludeInResult": "false"
        }
      ],
      "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-category:resource",
      "Attribute": [
        {"DataType": "http://www.w3.org/2001/XMLSchema#anyURI",
         "AttributeId": "urn:oasis:names:tc:xacml:1.0:resource:resource-id",
         "Value": "Ticket:1103",
         "IncludeInResult": "false"
       }
     ]
   }
}
```

```

    "DataType": "http://www.w3.org/2001/XMLSchema#anyURI",
    "AttributeId": "urn:nextlabs:names:evalsvc:1.0:resource:resource-type",
    "Value": "support_tickets",
    "IncludeInResult": "false"
}, {
    "DataType": "http://www.w3.org/2001/XMLSchema#anyURI",
    "AttributeId": "urn:oasis:names:tc:xacml:1.0:resource:prod_name",
    "Value": "Exchange Email",
    "IncludeInResult": "false"
        }]
    }, {
    "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-category:action",
    "Attribute": [
        {
            "DataType": "http://www.w3.org/2001/XMLSchema#string",
            "AttributeId": "urn:oasis:names:tc:xacml:1.0:action:action-id",
            "Value": "EDIT_TKTS",
            "IncludeInResult": "false"
                }
        ]
    }, {
    "CategoryId": "urn:nextlabs:names:evalsvc:1.0:attribute-
category:application",
    "Attribute": [
        {
            "DataType": "http://www.w3.org/2001/XMLSchema#string",
            "AttributeId": "urn:nextlabs:names:evalsvc:1.0:application:application-id",
            "Value": "Helpdesk",
            "IncludeInResult": "false"
        }
    ], {
        "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-
category:environment",
        "Attribute": [
            {
                "DataType": "http://www.w3.org/2001/XMLSchema#dateTime",
                "AttributeId":
                    "urn:oasis:names:tc:xacml:1.0:environment:authentication_type",
                "Value": "multifactor",
                "IncludeInResult": "false"
                    }
            ]
        }
    }
}

```

Sample response payload: Example 1

The response permits the action.

```
{
    "Response": {
        "Result": [
            {
                "Decision": "Permit",
                "Status": {
                    "StatusMessage": "success",
                    "StatusCode": {
                        "Value": "urn:oasis:names:tc:xacml:1.0:status:ok"
                    }
                }
            },
            "Obligations": []
        ]
    }
}
```

Single-query request (XML): Example 2

Sample request payload: Example 2

This request data contains an action (VIEW_TKTS) that a subject (Chris Webber) has performed on a document resource (Ticket:1103).

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
    http://docs.oasis-open.org/xacml/3.0/xacml-core-v3-schema-wd-17.xsd"
  ReturnPolicyIdList="false">
  <Attributes Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
    <Attribute IncludeInResult="false"
      AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">chris.webber@hdesk.com</AttributeValue>
    </Attribute>
    <Attribute IncludeInResult="false"
      AttributeId="urn:oasis:names:tc:xacml:1.0:subject:department">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">IT</AttributeValue>
    </Attribute>
    </Attributes>
    <Attributes Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource">
      <Attribute IncludeInResult="false"
        AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#String">Ticket:1103</AttributeValue>
      </Attribute>
      <Attribute IncludeInResult="false"
        AttributeId="urn:nextlabs:names:evalsvc:1.0:resource:resource-type">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#String">support_tickets</AttributeValue>
      </Attribute>
      </Attributes>
      <Attributes Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action">
        <Attribute IncludeInResult="false"
          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">VIEW_TKTS</AttributeValue>
        </Attribute>
        </Attributes>
        <Attributes Category="urn:nextlabs:names:evalsvc:1.0:attribute-category:application">
          <Attribute IncludeInResult="false"
            AttributeId="urn:nextlabs:names:evalsvc:1.0:application:application-id">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#String">MSExcel</AttributeValue>
          </Attribute>
          </Attributes>
          <Attributes Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment">
            <Attribute AttributeId="authentication_type" Issuer=""
              IncludeInResult="false">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">multifactor</AttributeValue>
            </Attribute>
            </Attributes>
          </Request>
```

Sample response payload: Example 2

The response permits the action.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Response xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17">
  <Result>
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
      <StatusMessage>success</StatusMessage>
    </Status>
    <Obligations/>
  </Result>
</Response>
```

Multi-query request (XML): Example 3

Sample request payload: Example 3

This request contains two subjects (`chris.webber@hdesk.com` and `amy.tan@hdesk.com`) who have performed two different actions (`EDIT_TKTS` and `ASSIGN_TKTS`) respectively on the support tickets resource (Ticket:1103), using the Helpdesk application.

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  ReturnPolicyIdList="false" CombinedDecision="false">
  <Attributes xml:id="subject1">
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">chris.webber@hdesk.com</AttributeValue>
    </Attribute>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:department">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">IT</AttributeValue>
    </Attribute>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:assigned_prod_area">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Exchange Email</AttributeValue>
    </Attribute>
  </Attributes>
  <Attributes xml:id="subject2">
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">amy.tan@hdesk.com</AttributeValue>
    </Attribute>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:department">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">IT</AttributeValue>
    </Attribute>
  </Attributes>
</Request>
```

```

<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">IT</
AttributeValue>
</Attribute>
</Attributes>
<Attributes xml:id="action1"
Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action">
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#string">EDIT_TKTS</AttributeValue>
</Attribute>
</Attributes>
<Attributes xml:id="action2"
Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action">
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#string">ASSIGN_TKTS</AttributeValue>
</Attribute>
</Attributes>
<Attributes xml:id="application1"
Category="urn:nextlabs:names:evalsvc:1.0:attribute-category:application">
<Attribute
AttributeId="urn:nextlabs:names:evalsvc:1.0:application:application-id"
IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#string">Helpdesk</AttributeValue>
</Attribute>
</Attributes>
<Attributes xml:id="resource1"
Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource">
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#string">Ticket:1103</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:nextlabs:names:evalsvc:1.0:resource:resource-
type" IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#string">support_tickets</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:nextlabs:names:evalsvc:1.0:resource:resource-
dimension" IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">from</
AttributeValue>
</Attribute>
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:prod_name"
IncludeInResult="false">
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Exchange
Email</AttributeValue>
</Attribute>
</Attributes>
<Attributes xml:id="environment1"
Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment">
<Attribute IncludeInResult="false">
AttributeId="urn:oasis:names:tc:xacml:1.0:environment:authentication_type"
>
<AttributeValue DataType="http://www.w3.org/2001/
XMLSchema#dateTime">multifactor</AttributeValue>
</Attribute>
</Attributes>
<MultiRequests>
<RequestReference>
<AttributesReference ReferenceId="subject1"/>

```

```

<AttributesReference ReferenceId="resource1"/>
<AttributesReference ReferenceId="action1"/>
<AttributesReference ReferenceId="application1"/>
<AttributesReference ReferenceId="environment1"/>
</RequestReference>
<RequestReference>
<AttributesReference ReferenceId="subject2"/>
<AttributesReference ReferenceId="resource1"/>
<AttributesReference ReferenceId="action2"/>
<AttributesReference ReferenceId="application1"/>
<AttributesReference ReferenceId="environment1"/>
</RequestReference>
</MultiRequests>
</Request>

```

Sample response payload: Example 3

The response permits the action.

```

<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17">
  <Result>
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok" />
      <StatusMessage>success</StatusMessage>
    </Status>
    <Obligations />
  </Result>
  <Result>
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok" />
      <StatusMessage>success</StatusMessage>
    </Status>
    <Obligations />
  </Result>
</Response>

```

SAML request: Example 4

SAML is used to exchange authentication and authorization data between identity providers and service providers.

The following example shows a SAML request.

```

<xacml-samlp:XACMLAuthzDecisionQueryType InputContextOnly="true"
  IssueInstant="2011-10-31T06:44:57.766Z" ReturnContext="false" Version="2.0"
  xmlns:xacml-
  samlp="urn:oasis:names:tc:xacml:2.0:profile:saml2.0:v2:schema:protocol">
  <saml:Issuer SPProvidedID="SPPProvierId"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">https://
    identity.carbon.wso2.org</saml:Issuer>
  <xacml-context:Request xmlns:xacml-
  context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
    <xacml-context:Subject
      SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-
      subject" xmlns:xacml-
      context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
      <xacml-context:Attribute
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string">

```

```

<xacml-context:AttributeValue>chris.webber@hdesk.com</xacml-
context:AttributeValue>
</xacml-context:Attribute>
<xacml-context:Attribute AttributeId="department" DataType="http://
www.w3.org/2001/XMLSchema#string">
<xacml-context:AttributeValue>IT</xacml-context:AttributeValue>
</xacml-context:Attribute>
</xacml-context:Subject>
<xacml-context:Resource xmlns:xacml-
context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
<xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
<xacml-context:AttributeValue>Ticket:1103</xacml-context:AttributeValue>
</xacml-context:Attribute>
<xacml-context:Attribute
AttributeId="urn:nextlabs:names:evalsvc:1.0:resource:resource-type"
DataType="http://www.w3.org/2001/XMLSchema#string">
<xacml-context:AttributeValue>support_tickets</xacml-
context:AttributeValue>
</xacml-context:Attribute>
</xacml-context:Resource>
<xacml-context:Action xmlns:xacml-
context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
<xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
<xacml-context:AttributeValue>VIEW_TKTS</xacml-context:AttributeValue>
</xacml-context:Attribute>
</xacml-context:Action>
<xacml-context:Environment xmlns:xacml-
context="urn:oasis:names:tc:xacml:2.0:context:schema:os"/>
</xacml-context:Request>
</xacml-samlp:XACMLAuthzDecisionQueryType>

```

NextLabs-specific environment attributes

NextLabs specific-environment attributes can be used to specify what happens when authorization requests do not match deployed policies or when error conditions occur.

There are two ways to use these environment attributes:

- Including attributes in authorization requests sent by the PEP: Environment attributes included in authorization requests are applied only to the current request.
- Setting attributes as system properties of the PDP. Environment attributes that are configured in the PDP properties are applied to every request processed by the PDP.

There are two NextLabs-specific environment attributes:

- dont-care-acceptable
- error-result-acceptable

Related concepts

[dont-care-acceptable](#) on page 295

Used to specify what happens when authorization requests do not match deployed policies.

[error-result-acceptable](#) on page 296

Used to specify what happens when errors are encountered.

dont-care-acceptable

Used to specify what happens when authorization requests do not match deployed policies.

When this attribute is set to yes or true, the PDP returns Indeterminate when the request does not match any of the deployed policies.

To use `dont-care-acceptable` in an authorization request, pass the attribute with the value `yes`.

Example of the attribute sent in a request

```
{  
  "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-category:environment",  
  "Attribute": [  
    { "DataType": "http://www.w3.org/2001/XMLSchema#string", "AttributeId":  
      "urn:oasis:names:tc:xacml:1.0:environment:dont-care-acceptable", "Value":  
      "yes", "IncludeInResult": "false" }  
  ]  
}
```

To use `dont-care-acceptable` as a system variable, include the attribute with the value `true` as a JVM system variable to be applied at PDP startup. The PDP returns indeterminate for every request that does not match a policy.

Example of the attribute as a system variable

```
-Dnextlabs.dont.care.acceptable=true
```

Related concepts

[NextLabs-specific environment attributes](#) on page 295

NextLabs specific-environment attributes can be used to specify what happens when authorization requests do not match deployed policies or when error conditions occur.

`error-result-acceptable`

Used to specify what happens when errors are encountered.

If this attribute is set to yes or true, the PDP returns `Error` if it encounters an error while processing the request.

To use `error-result-acceptable` in a request, pass attribute with the value `yes`.

Example of the attribute set in a request

```
{  
  "CategoryId": "urn:oasis:names:tc:xacml:3.0:attribute-category:environment",  
  "Attribute": [  
    { "DataType": "http://www.w3.org/2001/XMLSchema#string", "AttributeId":  
      "urn:oasis:names:tc:xacml:1.0:environment:error-result-acceptable",  
      "Value": "yes", "IncludeInResult": "false" }  
  ]  
}
```

To use `error-result-acceptable` as a PDP environment variable, include the attribute with the value `true` as a Java system variable during the PDP startup. The attribute is applied globally, and the PDP returns `Error` whenever it encounters an error while processing any request.

Example of the attribute as a system variable

```
-Dnextlabs.error.result.acceptable=true
```

Related concepts

[NextLabs-specific environment attributes](#) on page 295

NextLabs specific-environment attributes can be used to specify what happens when authorization requests do not match deployed policies or when error conditions occur.

Overview of plug-ins

Developers can extend the functionality of the Control Center and Policy Controller through plug-ins, which are separately installed pieces of software written in Java.

About Control Center plug-ins

Control Center plug-ins, or extensions, are typically used by the ICENet server (DABS) component.

The ICENet server brokers communications between the Control Center and the enforcers. It sends policy information and configuration profiles to Policy Controllers, and receives policy activity logs from Policy Controllers.

A typical Control Center plug-in is the heartbeat plug-in, which can be used to send and receive additional data between Control Center and the Policy Controller.

Control Center plug-ins have the following requirements:

- They must be written in Java.
- They must be packaged as JARs.
- Each plug-in must have an associated properties file, which tells Control Center the name and location of the plug-in.
- The JAR files must be stored in the following location, either directly in the `plugins` folder, or in a subfolder within `plugins`.

`<Control Center install path>/plugins`

- The properties file must be stored in the following location:

`<Control Center install path>/plugins/config`

Related concepts

[About heartbeat plug-ins](#) on page 302

Policy Controllers are configured to send regular heartbeats to the Control Center to indicate that they are running normally.

Control Center plug-in code basics

All Control Center plug-ins must implement the following interface.

```
com.mycompany.destiny.container.dcc.plugin.IDCCServerPlugin  
The following code is the most basic plug-in code. The init( ) method is the  
only method in this interface.  
package com.nextlabs.plugins;  
import com.mycompany.destiny.container.dcc.plugin.IDCCServerPlugin;  
import  
    com.mycompany.destiny.server.shared.registration.IRegisteredDCCComponent;  
public class SimpleServerPlugin implements IDCCServerPlugin {  
    public void init(IRegisteredDCCComponent component) {  
    }  
}
```

The code must be compiled into a JAR file with a manifest that names the main class of the plug-in, as follows:

```
Provider-Class: com.nextlabs.plugins.SimpleServerPlugin
```

Related concepts

[Building a server heartbeat plug-in](#) on page 307

A Server Heartbeat plug-in must implement the following interface.

Control Center plug-in configuration

You must create a properties file that tells the Control Center about the plug-in.

The following is an example of the content in a properties file (named, for example, `SimpleServerPlugin.properties`) for the sample JAR created in the previous section:

```
name = SimpleServerPlugin
jar-path = [policy-server]/plugins/SimpleServerPlugin.jar
description = Basic plug-in example
applications = DABS
display_name = Simple Server Plugin
```

The following properties are required:

- name—If you build multiple Control Center plug-ins, each plug-in must have a unique name.
- jar-path—This property provides the location of the JAR. The `[policy-server]` string is automatically replaced with the path where Control Center is installed, which is typically `C:\Program Files\NextLabs\Policy Server`.
- applications—This property specifies the Control Center component into which the plug-in is loaded. Currently, DABS (the ICENet server) is the only Control Center component that supports plug-ins.

The properties file must be placed in the following location:

```
<Control Center install path>\plugins\config
```

Related concepts

[About heartbeat plug-ins](#) on page 302

Policy Controllers are configured to send regular heartbeats to the Control Center to indicate that they are running normally.

About Policy Controller plug-ins

The Policy Controller is the component that communicates with the Control Center and the Policy Enforcement Points (PEPs).

The Policy Controller receives policy information from Control Center and policy queries from PEPs. Its core functionality is as the Policy Decision Point (PDP) that evaluates policies and decides if a user action is allowed or denied. The Policy Controller sends policy decisions to the PEP and policy activity logs to Control Center.

The Policy Controller supports the following types of plug-ins:

- Heartbeat plug-ins that send information between the Policy Controller and Control Center
- Subject or Resource attribute provider plug-ins that provide additional subject or resource attributes for policy evaluation
- External service provider plug-ins that expose an external interface that is callable through C

A Policy Controller plug-in can be multiple types, for example, it can be both a heartbeat and a subject attribute provider plug-in, if both types of service are required. The Policy Controller loads and registers all plug-ins at start-up.

All Policy Controller plug-ins share the following characteristics:

- They are written in Java.
- They are packaged as JARs.
- They each have an associated properties file, which tells the Policy Controller the name and location of the plug-in.
- They each have JAR files that are stored in the following location:

```
<Policy Controller install folder>\Policy Controller\jservice\jar\<plug-in folder>
```

- They each have properties files that are stored in the following location:

<Policy Controller install folder>\Policy Controller\jservice\config

Policy Controller plug-in code basics

All Policy Controller plug-ins must implement the following interface.

```
com.mycompany(pf.domain.destiny.serviceprovider.IServiceProvider
```

The following code creates a client plug-in. The `init()` method is the only method in this interface.

```
package com.nextlabs.plugins;
import com.mycompany(pf.domain.destiny.serviceprovider.IServiceProvider;
public class SimpleClientPlugin implements IServiceProvider {
    public void init() {
    }
}
```

The code must be compiled into a JAR file with a manifest that names the main class of the plug-in, as follows:

```
Provider-Class: com.nextlabs.plugins.SimpleClientPlugin
```

Related concepts

[Building a subject attribute provider plug-in](#) on page 300

A Subject Attribute Provider plug-in must implement the following interface:

[Building a resource attribute provider plug-in](#) on page 301

A Resource Attribute Provider plug-in must implement the following interface.

[Building a client heartbeat plug-in](#) on page 303

A client heartbeat plug-in must implement the following interface.

Policy Controller plug-in configuration

You must create a properties file that tells the Policy Controller about the plug-in.

The following is an example of the content in a properties file (named, for example, `SimpleClientPlugin.properties`) for the sample JAR created in the previous section:

```
name = SimpleClientPlugin
jar-path = [NextLabs]/Policy Controller/jservice/jar/simpleplugin/
SimpleClientPlugin.jar
friendly name = User Attributes Service
description = User Attributes Plugin
```

The `name` property is required. If you build multiple Policy Controller plug-ins, each plug-in must have a unique name.

The `jar-path` property is also required to provide the location of the JAR. The `[NextLabs]` string is automatically replaced with the location of the `NextLabs` folder, which is typically `C:/Program Files/NextLabs`.

The properties file must be placed in the following location:

<installation folder>\Policy Controller\jservice\config

Related concepts

[About heartbeat plug-ins](#) on page 302

Policy Controllers are configured to send regular heartbeats to the Control Center to indicate that they are running normally.

Using attribute provider plug-ins

This section describes how to develop attribute provider plug-ins.

About attribute provider plug-ins

Attribute information used in policy evaluations typically comes either from the Policy Server indirectly as maps, which are static data that are enrolled in the Control Center database, or from a PEP.

There are, however, cases where a policy requires attribute information that cannot be enrolled, or it requires attribute information to be accessed dynamically at policy evaluation time. For these cases, you can build an Attribute Provider plug-in to provide attribute information to the policy evaluation engine at run time.

Consider the following policy example:

```
FOR OPEN
ON resource.fso.classified = "YES"
BY *
WHERE user.license != resource.fso.license
DO DENY
```

This policy denies a user from opening a classified document if the user's license does not match the document's license. The `WHERE` clause is specified in the Expression section of a policy definition as shown in the following figure.

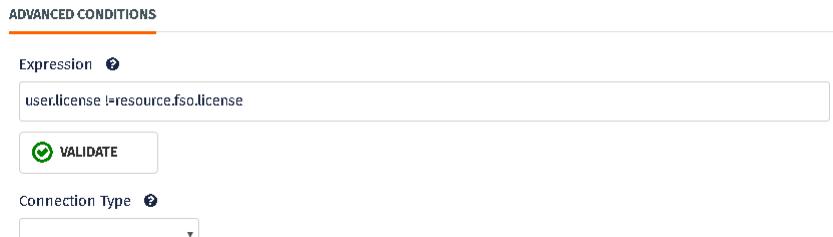


Figure 100: Evaluation condition specified in a policy

To evaluate this policy and match the license values dynamically, the Policy Controller requires the user's license in real time. If the PEP cannot supply the user license, a Subject Attribute Provider plug-in must retrieve it.

If a policy requires resource attribute information at policy evaluation time, you would build a Resource Attribute Provider plug-in to supply the information.

Building a subject attribute provider plug-in

A Subject Attribute Provider plug-in must implement the following interface:

```
com.mycompany(pf.domain.destiny.serviceprovider.ISubjectAttributeProvider)
```

The `ISubjectAttributeProvider` interface extends `IServiceProvider`, which, as noted in the Policy Controller plug-in code basics section, is the interface that all Policy Controller plug-ins use.

The Subject Attribute Provider plug-in code must implement the following methods:

- `init()`
- `getAttribute()`

Related concepts

[Policy Controller plug-in code basics](#) on page 299

All Policy Controller plug-ins must implement the following interface.

Retrieving the subject attribute

A Subject Attribute Provider plug-in typically retrieves user attributes. Implement `getAttribute()` to specify the user attribute to retrieve.

```
IEvalValue getAttribute(IDSubject subj, String attribute) throws
```

```
ServiceProviderException;
```

- `subj` is the class that specifies the subject, typically a user, whose attribute to retrieve. The method in `IDSubject` that you typically use is `getUid()`, which returns the user ID.
- `attribute` is the name of the attribute to retrieve.
- `getAttribute()` returns the result in an `IEvalValue` object. You build the return value using `build()`, a method of the `EvalValue` class.

The plug-in should return `NULL` if it is unable to retrieve the attribute information.

Basic code

The following is the basic code for a Subject Attribute Provider plug-in that provides user attributes.

```
import com.mycompany.framework.expressions.EvalValue;
import com.mycompany.framework.expressions.IEvalValue;
import
com.mycompany(pf.domain.destiny.serviceprovider.ServiceProviderException;
import com.mycompany(pf.domain.destiny.subject.IDSubject;
import com.mycompany(pf.domain.destiny.subject.SubjectType;
public class UserAttributeProvider implements ISubjectAttributeProvider {
    public void init() throws Exception {
        // Any set-up code should go here. If the user attribute is retrieved
        from a database, for example, this could open the database connection.
    }
    IEvalValue getAttribute(IDSubject subj, String attribute) throws
ServiceProviderException {
        if (subj.getSubjectType() != SubjectType.USER) {
            // This code only handles user attributes
            return IEvalValue.NULL;
        }
        // Typically the user sid, but depending on the PEP could be
        // the email address
        String id = subj.getUid();
        if (attribute.equals("license")) {
            // Retrieve the license information for the user specified
            // by "id"
            String value = ...
            // And turn it into an IEvalValue
            return EvalValue.build(value);
        } else if (attribute.equals("security_clearance")) {
            // A plug-in can provide information for as many different
            // attributes as needed
            long securityClearance = ...
            return EvalValue.build(securityClearance);
        }
        return IEvalValue.NULL;
    }
}
```

Building a resource attribute provider plug-in

A Resource Attribute Provider plug-in must implement the following interface.

```
com.mycompany(pf.domain.destiny.serviceprovider.IResourceAttributeProvider
```

The `IResourceAttributeProvider` interface extends `IServiceProvider`, which, as noted in the Policy Controller plug-in code basics section, is the interface that all Policy Controller plug-ins use.

The Resource Attribute Provider plug-in code must implement the following methods:

- `init()`
- `getAttribute()`

Related concepts

[Policy Controller plug-in code basics](#) on page 299

All Policy Controller plug-ins must implement the following interface.

Retrieving the resource attribute

Implement `getAttribute()` to specify the resource attribute to retrieve.

```
IEvalValue getAttribute(IResource resource, String attribute) throws  
ServiceProviderException;
```

- `resource` is the class that specifies the resource whose attribute to retrieve. The method in `IResource` that you typically use is `getIdentifier()`, which returns the resource name.
- `attribute` is the name of the attribute to retrieve.
- `getAttribute()` returns the result in an `IEvalValue` object.

The plug-in should return `NULL` if it is unable to retrieve the attribute information.

Basic code

The following is the basic code for a resource attribute provider plug-in.

```
import com.mycompany.framework.expressions.EvalValue;  
import com.mycompany.framework.expressions.IEvalValue;\  
import  
com.mycompany(pf.domain.destiny.serviceprovider.ServiceProviderException;  
import com.mycompany(pf.domain.destiny.epicenter.resource.IResource;  
...  
IEvalValue getAttribute(IResource resource, String attribute) throws  
ServiceProviderException {  
String name = resource.getAttribute("name");  
if (attribute.equals("someAttribute")) {  
// Get appropriate value ...  
String value = ...;  
return EvalValue.build(value);  
}  
return IEvalValue.NULL;  
}
```

Using heartbeat plug-ins

This section describes how to develop heartbeat plug-ins.

About heartbeat plug-ins

Policy Controllers are configured to send regular heartbeats to the Control Center to indicate that they are running normally.

When a heartbeat occurs, the Policy Controller receives any pending policy deployments or configuration updates from the Control Center.

A heartbeat plug-in uses this communication channel between the Control Center and the Policy Controller to send and receive other types of data. For example, you can build a heartbeat plug-in to send external authorization data, such as licenses, to all Policy Controllers, or to send installation or upgrade information about each Policy Controller to store in Control Center.

A heartbeat plug-in consists of two plug-ins:

- A client plug-in on the Policy Controller
- A server plug-in on the Control Center

The heartbeat communication between the Policy Controller and Control Center occurs as follows:

1. Before each heartbeat, the Policy Controller calls the `prepareRequest()` method in each registered heartbeat plug-in. The method's response and the name of the client plug-in are added to the heartbeat, which is sent to the Control Center.
2. When the Control Center receives the heartbeat, it takes each request, finds the server plug-in with a matching name, and calls the plug-in's `serviceHeartbeatRequest()` method with the data. The method's response is added to the heartbeat response, which is sent to the Policy Controller.
3. When the Policy Controller receives the heartbeat response from the Control Center, it calls the `processResponse()` method in the corresponding client plug-in.

The plug-ins on both sides are identified by matching names. It is best practice to also use the same name in the server and client properties file, that is, you should set the `name` property to the same value.

Related concepts

[About Control Center plug-ins](#) on page 297

Control Center plug-ins, or extensions, are typically used by the ICENet server (DABS) component.

[Control Center plug-in configuration](#) on page 298

You must create a properties file that tells the Control Center about the plug-in.

[Policy Controller plug-in configuration](#) on page 299

You must create a properties file that tells the Policy Controller about the plug-in.

Building a client heartbeat plug-in

A client heartbeat plug-in must implement the following interface.

```
com.mycompany(pf.domain.destiny.serviceprovider.IHeartbeatServiceProvider
```

The `IHeartbeatServiceProvider` interface extends the following interfaces:

- `IServiceProvider`, which, as noted in the Policy Controller plug-in code basics section, is the interface that all Policy Controller plug-ins use. This interface defines the `init()` method.
- `IHeartbeatListener`, which creates a listener that produces and processes information for and from a heartbeat. This interface defines the `prepareRequest()` and `processResponse()` methods.

The client heartbeat plug-in code must implement the following methods:

- `init()`
- `prepareRequest()`
- `processResponse()`

Related concepts

[Policy Controller plug-in code basics](#) on page 299

All Policy Controller plug-ins must implement the following interface.

Preparing the request

Before each heartbeat, the Policy Controller calls the `prepareRequest()` method in each registered heartbeat plug-in.

The method's request and the name of the client plug-in are added to the heartbeat, which is sent to the Control Center. The content of a request is a Java serializable object. The following is a simple example of a request that sends a string:

```
public Serializable prepareRequest(String id) {
    return "Hello";
}
```

Developers typically implement `prepareRequest()` to request data, for a specified ID, from the server to use in policy evaluations. Because it is important to avoid sending unnecessary and excessive

data back and forth, the request should include the timestamp of the last server response. The server plug-in can use this timestamp information to determine if new data needs to be sent back to the client.

Processing the response

When the Policy Controller receives the heartbeat response from the Control Center, it calls the `processResponse()` method in the client plug-in.

You implement this method to process the response, which is returned as a Java serializable object. In the following simple example, `processResponse()` prints a string and the string data returned by the server.

```
public void processResponse(String id, String data) {  
    System.out.println("This is the data from the server: " + (String) data);  
}
```

In most cases, the serialized object returned is more complex than a simple string. You can use the `SerializationUtils` class to deserialize the response object. However, because plug-ins are loaded by a specially created class loader, an object whose class is defined in that specially class loader is not visible to `SerializationUtils`, which is loaded by the main class loader. A convenient way to deserialize an object is to use a custom method that lets you supply the class loader to be used by deserialization, as shown in the following example:

```
public class heartbeatClient implements IHeartbeatServiceProvider {  
    // Define the classLoader variable  
    private ClassLoader classLoader = getClass().getClassLoader();  
    ...  
    public void processResponse(String id, String data) {  
        // Deserialize the object using the classloader  
        Serializable obj = SerializationUtils.unwrapSerialized(data,  
            classLoader);
```

Sample client plug-in code

The following is the code for a sample client heartbeat plug-in that is also a subject attribute provider plug-in.

This plug-in is a utility for passing dynamic user attributes to the Policy Controller for policy evaluation. It lets the policy designer specify, through a `userattributes.txt` file in a configuration folder, which user attributes should be sent.

The code after the `processResponse()` method is specific to this plug-in's implementation and is shown here for completeness; it is not essential to understanding how client heartbeat plug-ins work.

```
package com.nextlabs.plugins.userattributes;  
import java.io.BufferedReader;  
import java.io.ByteArrayInputStream;  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.io.ObjectInput;  
import java.io.ObjectInputStream;  
import java.io.ObjectOutput;  
import java.io.ObjectOutputStream;  
import java.io.Serializable;  
import java.util.HashMap;  
import java.util.List;  
import java.util.Map;  
import org.apache.commons.logging.Log;  
import org.apache.commons.logging.LogFactory;  
import sun.misc.BASE64Decoder;  
import sun.misc.BASE64Encoder;
```

```

import com.mycompany.destiny.agent.controlmanager.AgentTypeEnum;
import com.mycompany.destiny.agent.controlmanager.ControlMngr;
import com.mycompany.destiny.agent.controlmanager.IControlManager;
import com.mycompany.destiny.agent.ipc.IOSWrapper;
import com.mycompany.destiny.agent.ipc.OSWrapper;
import com.mycompany.framework.comp.ComponentManagerFactory;
import com.mycompany.framework.expressions.EvalValue;
import com.mycompany.framework.expressions.IEvalValue;
import com.mycompany.framework.expressions.IMultivalue;
import com.mycompany.framework.utils.SerializationUtils;
import com.mycompany(pf.domain.destiny.subject.IDSubject;
import
com.mycompany(pf.domain.destiny.serviceprovider.IHeartbeatServiceProvider;
import
com.mycompany(pf.domain.destiny.serviceprovider.ISubjectAttributeProvider;
import
com.mycompany(pf.domain.destiny.serviceprovider.ServiceProviderException;
import com.nextlabs.plugins.userattributes.dto.UserAttributesRequestDTO;
import com.nextlabs.plugins.userattributes.dto.UserAttributesResponseDTO;
public class UserAttributesClient implements IHeartbeatServiceProvider,
ISubjectAttributeProvider {
private long timestamp = 0;
private static String hostName;
private IOSWrapper osWrapper =
ComponentManagerFactory.getComponentManager().getComponent(OSWrapper.class);
private IControlManager controlMngr=
ComponentManagerFactory.getComponentManager().getComponent(ControlMngr.COMP_INFO);
private ClassLoader classLoader = getClass().getClassLoader();
private static AgentTypeEnum agentType;
private Map<String, IEvalValue[]> users = new HashMap<String, IEvalValue[]>();
private Map<String, Integer> attrToIndexMap = new HashMap<String, Integer>();
private static final Log log =
LogFactory.getLog(UserAttributesClient.class.getName());
public void init() {
hostName = osWrapper.getFQDN();
agentType =
AgentTypeEnum.getAgentTypeEnum(controlMngr.getAgentType().getValue());
try {
loadFromDisk();
} catch (FileNotFoundException e) {
log.info("No cached attribute information stored on disk");
} catch (IOException e) {
log.error("Unable to load attributes information from disk", e);
}
}
public Serializable prepareRequest(String id) {
if (UserAttributesRequestDTO.PLUGIN.equals(id)) {
String[] currentUsers;
if (agentType == AgentTypeEnum.DESKTOP) {
// get all users
currentUsers = currentUserIds();
} else {
currentUsers = new String[1];
currentUsers[0] = "*";
}
UserAttributesRequestDTO request = new UserAttributesRequestDTO(timestamp, hostName, currentUsers);
return request;
}
return null;
}

```

```

    }
    public void processResponse(String id, String data) {
        // We can't use the default classloader to deserialize here,      //
        because it can't see UserAttributesRequestDTO.
        Serializable obj = SerializationUtils.unwrapSerialized(data, classLoader);
        if (UserAttributesRequestDTO.PLUGIN.equals(id)) {
            if (obj == null) {
                return;
            }
            UserAttributesResponseDTO response = (UserAttributesResponseDTO) obj;
            update(response);
            try {
                saveToDisk(response);
            } catch (IOException e) {
                log.error("Failed to save attribute information to disk", e);
            }
        }
    }
    private synchronized void update(UserAttributesResponseDTO response) {
        if (response == null) {
            return;
        }
        timestamp = response.getTimestamp();

        int i = 0;
        for (String attribute : response.getAttributes()) {
            attrToIndexMap.put(attribute.toLowerCase(), i++);
        }
        for (UserAttributesResponseDTO.IDAndValues idAndVal :
            response.getUserInformation()) {
            String id = idAndVal.getUserId();
            List<IEvalValue> values = idAndVal.getValues();

            users.put(id, values.toArray(new IEvalValue[values.size()]));
        }
    }
    private void saveToDisk(UserAttributesResponseDTO response) throws
        IOException, FileNotFoundException {
        ObjectOutputStream oos = null;
        try {
            oos = getOutputStream();
            response.writeExternal(oos);
        } finally {
            if (oos != null) {
                oos.close();
            }
        }
    }
    private void loadFromDisk() throws IOException, FileNotFoundException {
        UserAttributesResponseDTO response = new UserAttributesResponseDTO();

        ObjectInputStream ois = null;
        try {
            ois = getInputStream();
            response.readExternal(ois);

            update(response);
        } finally {
            if (ois != null) {
                ois.close();
            }
        }
    }
}

```

```

public synchronized IEvalValue getAttribute(IDSubject subj, String
attribute) throws ServiceProviderException {
String id = subj.getUid();
log.debug("Getting attribute " + attribute + " for user " + id);
IEvalValue[] values = users.get(subj.getUid());
if (values == null) {
log.debug("No attributes recorded for user " + id);
return IEvalValue.NULL;
}
Integer index = attrToIndexMap.get(attribute.toLowerCase());
if (index == null) {
log.debug("Attribute " + attribute + " unknown for user " + id);
return IEvalValue.NULL;
}
if (index >= values.length) {
log.debug("Attribute " + attribute + " invalid for user " + id);
return IEvalValue.NULL;
}
IEvalValue val = values[index];
if (val.getValue() instanceof IMultivalue) {
StringBuilder sb = new StringBuilder("User " + id + ":" + attribute +
"=");
boolean first = true;
for (IEvalValue v : (IMultivalue)val.getValue()) {
if (!first) {
sb.append(", ");
}
first = false;
sb.append((String)v.getValue());
}
log.debug(sb.toString());
} else {
log.debug("User " + id + ":" + attribute + "=" + (String)val.getValue());
}
return val;
}
private ObjectInput getInputStream() throws IOException,
FileNotFoundException {
return new ObjectInputStream(new FileInputStream(getStoreFileName()));
}
private ObjectOutputStream getOutputStream() throws IOException,
FileNotFoundException {
return new ObjectOutputStream(new FileOutputStream(getStoreFileName()));
}
private String getStoreFileName() {
return new File("userattributes.bin").getAbsolutePath();
}
private String[] currentUserIds() {
return osWrapper.getLoggedInUsers();
}

```

Related concepts

[Sample server plug-in code](#) on page 308

Building a server heartbeat plug-in

A Server Heartbeat plug-in must implement the following interface.

```
com.mycompany.destiny.container.dcc.plugin.IDCCHearbeatServerPlugin
```

The IDCCHearbeatServerPlugin interface extends the following interfaces:

- [IDCCServerPlugin](#), which, as noted in the Control Center plug-in code basics section, is the interface that all Control Center plug-ins use. This interface defines the `init()` method.

- `IHeartbeatProvider`, which creates an object that provides the data requested by a client heartbeat plug-in. This interface defines the `serviceHeartbeatRequest()` method.

The server heartbeat plug-in code must implement the following methods:

- `init()`
- `serviceHeartbeatRequest()`

Related concepts

[Control Center plug-in code basics](#) on page 297

All Control Center plug-ins must implement the following interface.

Responding to the client request

When the Control Center receives the heartbeat and the request from the Policy Controller, it finds the server plug-in with a matching name, and calls the plug-in's `serviceHeartbeatRequest()` method with the name of the plug-in and the request data.

Implement this method to respond to the client request. The response must be a Java serializable object. The following is a simple example:

```
public Serializable serviceHeartbeatRequest(String name, String data) {
    System.out.println("This is the request we got from the client: " +
        data);
    return "Hello to you, too";
}
```

Sample server plug-in code

The following is the code for the server heartbeat plug-in that communicates with the client heartbeat plug-in, whose code is shown in the Sample client plug-in code section. This heartbeat plug-in example is also a subject attribute provider plug-in. The code after the `serviceHeartbeatRequest()` method is specific to this plug-in's implementation and is shown here for completeness; it is not essential to understanding how server heartbeat plug-ins work.

```
package com.nextlabs.plugins.userattributes;
import java.io.BufferedReader;
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.List;
import java.util.SortedSet;
import java.util.TreeSet;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;
import com.mycompany.destiny.container.dcc.DCCResourceLocators;
import com.mycompany.destiny.container.dcc.INamedResourceLocator;
import com.mycompany.destiny.container.dcc.ServerRelativeFolders;
import com.mycompany.destiny.container.dcc.plugin.IDCCHeartbeatServerPlugin;
import com.mycompany.destiny.server.shared.configuration.
IDABSConfigurationDO;
import com.mycompany.destiny.server.shared.configuration.
ITrustedDomainsConfigurationDO;
import
com.mycompany.destiny.server.shared.registration.IRegisteredDCCComponent;
```

```

import com.mycompany.destiny.server.shared.registration.ServerComponentType;
import com.mycompany.dictionary.Dictionary;
import com.mycompany.dictionary.DictionaryException;
import com.mycompany.dictionary.IElementField;
import com.mycompany.dictionary.ElementFieldData;
import com.mycompany.dictionary.IDictionary;
import com.mycompany.dictionary.IDictionaryIterator;
import com.mycompany.dictionary.IElementType;
import com.mycompany.dictionary.IEnrollment;
import com.mycompany.dictionary.IMLElementBase;
import com.mycompany.domain.enrollment.ElementTypeEnumType;
import com.mycompany.framework.comp.ComponentManagerFactory;
import com.mycompany.framework.comp.IComponentManager;
import com.mycompany.framework.configuration.DestinyConfigurationStoreImpl;
import com.mycompany.framework.configuration.IDestinyConfigurationStore;
import com.mycompany.framework.expressions.BooleanOp;
import com.mycompany.framework.expressions.CompositePredicate;
import com.mycompany.framework.expressions.Constant;
import com.mycompany.framework.expressions.EvalValue;
import com.mycompany.framework.expressions.IEvalValue;
import com.mycompany.framework.expressions.IPredicate;
import com.mycompany.framework.expressions.Multivalue;
import com.mycompany.framework.expressions.PredicateConstants;
import com.mycompany.framework.expressions.Relation;
import com.mycompany.framework.expressions.RelationOp;
import com.mycompany(pf).destiny.lib.DictionaryHelper;
import com.mycompany(pf).domain.destiny.common.ServerSpecManager;
import com.mycompany(pf).domain.destiny.subject.SubjectAttribute;
import com.mycompany(pf).domain.destiny.subject.SubjectType;
import com.nextlabs.plugins.userattributes.dto.UserAttributesRequestDTO;
import com.nextlabs.plugins.userattributes.dto.UserAttributesResponseDTO;
public class UserAttributesServer implements IDCCHeartbeatServerPlugin {
    private static final String ATTRIBUTES_FILE_NAME = "userattributes.txt";
    private static final Log log =
        LogFactory.getLog(UserAttributesServer.class.getName());
    // When we query the dictionary for attribute values, we also need some way
    to identify the user
    // Ideally the particular keys to use should come from querying the
    dictionary
    private static final List<String> fixedAttributes =
        Arrays.asList("windowsSid", "unixId", "principalName");
    private final List<String> attributes;
    private final IElementField[] elems;
    private final IElementType userType;
    private final IComponentManager componentManager;
    private final IDictionary dictionary;
    private final DictionaryHelper dictionaryHelper;
    private final IPredicate userCondition;
    public UserAttributesServer() {
        componentManager = ComponentManagerFactory.getComponentManager();
        dictionary = componentManager.getComponent(Dictionary.COMP_INFO);
        try {
            userType = dictionary.getType(ElementTypeEnumType.USER.getName());
        } catch (DictionaryException e) {
            throw new IllegalStateException("Unable to obtain user type from the
                dictionary.", e);
        }
        userCondition = dictionary.condition(userType);
        ServerSpecManager serverSpecManager =
            componentManager.getComponent(ServerSpecManager.COMP_INFO);
        dictionaryHelper = new DictionaryHelper(serverSpecManager, dictionary);
        attributes = loadAttributes();
        elems = createElements(attributes);
    }
}

```

```

public void init(IRegisteredDCCComponent notUsed) {
    return;
}
public Serializable serviceHeartbeatRequest(String providerName, String
requestData) {
if (UserAttributesRequestDTO.PLUGIN.equals(providerName)) {
Serializable data = unwrapSerialized(requestData);
if (data instanceof UserAttributesRequestDTO) {
UserAttributesRequestDTO request = (UserAttributesRequestDTO) data;
// Check to see if we have any attributes. The list should exist
// and have more than two entries (the first two being the various ids
// we send back)
if (attributes == null || attributes.size() <= fixedAttributes.size()) {
// No attributes
return null;
}
long lastModification = mostRecentChange();
if (request.upToDate(lastModification)) {
// No changes
return null;
}
UserAttributesResponseDTO response = new
UserAttributesResponseDTO(lastModification,
attributes.subList(fixedAttributes.size(), attributes.size()));
String[] users = request.getUserIds();
try {
if (users.length == 0 || "*".equals(users[0])) {
getUserInfoByDomain(request.getDomain(), response);
} else {
getUserInfoByUser(users, response);
}
} catch (DictionaryException e) {
log.error("Dictionary exception when getting user attribute information",
e);
return null;
}

return response;
}
}
return null;
}
private IELEMENTFIELD[] createElements(List<String> attributes) {
if (attributes == null) {
return new IELEMENTFIELD[0];
}
// Sanity check
String[] validFieldNames = userType.getFieldNames();

List<IELEMENTFIELD> elems = new ArrayList<IELEMENTFIELD>();
for (String attribute : attributes) {
if (isValidAttribute(attribute, validFieldNames)) {
elems.add(userType.getField(attribute));
} else {
log.warn("Attribute [" + attribute + "] is not known. Ignoring");
}
}

return elems.toArray(new IELEMENTFIELD[elems.size()]);
}
private boolean isValidAttribute(String attribute, String[]
validFieldNames) {
if (attribute == null) {
return false;
}

```

```

}
for (String validFieldName : validFieldNames) {
if (attribute.equals(validFieldName)) {
return true;
}
}
return false;
}
private void getUserInfoByUser(String[] userIds, UserAttributesResponseDTO
response) throws DictionaryException {
List<IPredicate> userIdentityPredicates = new ArrayList<IPredicate>();
for (String userId : userIds) {
userIdentityPredicates.add(dictionaryHelper.toDictionaryPredicate(new
Relation(RelationOp.EQUALS, (SubjectAttribute) SubjectAttribute.USER_UID,
Constant.build(userId)), null, SubjectType.USER));
}
IPredicate condition = new CompositePredicate(BooleanOp.AND, userCondition,
new CompositePredicate(BooleanOp.OR, userIdentityPredicates));
IDictionaryIterator<ElementFieldData> userData = null;
try {
userData = dictionary.queryFields(elems,
condition,
dictionary.getLatestConsistentTime(), null, null);
while (userData.hasNext()) {
buildResponse(response, userData.next());
}
} finally {
if (userData != null) {
userData.close();
}
}
}
private void buildResponse(UserAttributesResponseDTO response,
ElementFieldData data) {
Object[] values = data.getData();

List<IEvalValue> attributeValues = new ArrayList<IEvalValue>();

for (int i = fixedAttributes.size(); i < values.length; i++) {
IEvalValue v;
if (values[i] instanceof String || values[i] == null) {
v = EvalValue.build((String)values[i]);
} else if (values[i] instanceof Long) {
v = EvalValue.build((Long)values[i]);
} else if (values[i] instanceof String[]) {
v = EvalValue.build(Multivalue.create((String[])values[i]));
} else if (values[i] instanceof Long[]) {
v = EvalValue.build(Multivalue.create((Long[])values[i]));
} else {
throw new IllegalArgumentException("Element " + values[i] + " of unexpected
type");
}
attributeValues.add(v);
}

for (int i = 0; i < fixedAttributes.size(); i++) {
if (values[i] != null) {
response.addUserInfo((String)values[i], attributeValues);
}
}
}
}

```

```

private void getUserInfoByDomain(String rootDomain,
UserAttributesResponseDTO response) throws DictionaryException {
Collection<String> domains = expandDomainList(rootDomain);
List<IPredicate> domainPredicates = new
ArrayList<IPredicate>(domains.size());
for (String domain : domains) {
IEnrollment enrollment = dictionary.getEnrollment(domain);
if (enrollment != null) {
domainPredicates.add(dictionary.condition(enrollment));
}
}
IPredicate domainCondition;
if (domainPredicates.size() > 0) {
domainCondition = new CompositePredicate(BooleanOp.OR, domainPredicates);
} else {
domainCondition = PredicateConstants.TRUE;
}
IDictionaryIterator<ElementFieldData> usersInDomain = null;
try {
usersInDomain = dictionary.queryFields(elems,
new CompositePredicate(BooleanOp.AND, domainCondition, userCondition),
dictionary.getLatestConsistentTime(), null, null);
while (usersInDomain.hasNext()) {
buildResponse(response, usersInDomain.next());
}
} finally {
if (usersInDomain != null) {
usersInDomain.close();
}
}
}
private List<String> loadAttributes() {
List<String> attributes = new ArrayList<String>();
// We need these so that we can identify the user associated with the
attributes
attributes.addAll(fixedAttributes);
String attributesFile = getAttributesDataFileName();
if (! new File(attributesFile).isFile()) {
return null;
}
BufferedReader br = null;
try {
br = new BufferedReader(new FileReader(getAttributesDataFileName()));
String line;
while((line = br.readLine()) != null) {
if (line.length() > 0) {
attributes.add(line);
}
}
} catch (IOException e) {
log.error("Exception reading attributes.", e);
return null;
} finally {
if (br != null) {
try {
br.close();
} catch (IOException e) {
// Seriously?
}
}
}
return attributes;
}
private long mostRecentChange() {

```

```

long dictionaryChange = -1;
try {
    dictionaryChange = dictionary.getLatestConsistentTime().getTime();
} catch (DictionaryException e) {
}
long fileChange = new File(attributesDataFileName()).lastModified();
return Math.max(dictionaryChange, fileChange);
}
private String getAttributesDataFileName() {
INamedResourceLocator serverResourceLocator = (INamedResourceLocator)
componentManager.getComponent(DCCResourceLocators.SERVER_HOME_RESOURCE_LOCATOR);
return serverResourceLocator.getFullyQualifiedName
(ServerRelativeFolders.CONFIGURATION_FOLDER.getPathOfContainedFile(ATTRIBUTES_FILE_NAME));
}

/*
 * Copied nearly verbatim from PolicyQueryImpl.java
 */
private Collection<String> expandDomainList(String agentDomain) {
if (agentDomain == null || agentDomain.length() == 0) {
log.warn("Agent domain is null - using an empty domain list.");
return new ArrayList<String>();
}
SortedSet<String> res = new TreeSet<String>();
res.add(agentDomain);
try {
IDestinyConfigurationStore confStore = (IDestinyConfigurationStore)
componentManager.getComponent(DestinyConfigurationStoreImpl.COMP_INFO);
IDABSComponentConfigurationDO dabsConfig = (IDABSComponentConfigurationDO)
confStore.retrieveComponentConfiguration(ServerComponentType.DABS.getName());
if (dabsConfig != null) {
ITrustedDomainsConfigurationDO tdConfig =
dabsConfig.getTrustedDomainsConfiguration();
if (tdConfig != null) {
String[] trustedDomains = tdConfig.getTrustedDomains();
if (trustedDomains != null) {
for (int i = 0; i != trustedDomains.length; i++) {
if (trustedDomains[i] != null) {
String[] domains = trustedDomains[i].split(",");
for (int j = 0; j != domains.length; j++) {
domains[j] = domains[j].trim();
}
if (Arrays.asList(domains).contains(agentDomain)) {
log.info("Domain '" + agentDomain + "' is in the list of mutual trust: " +
trustedDomains[i]);
for (int j = 0; j != domains.length; j++) {
res.add(domains[j]);
}
}
}
}
}
} else {
log.warn("Unable to get trusted domain configuration.");
}
} else {
log.warn("Unable to get DABS configuration - trusted domains will not be
configured.");
}
} catch (Exception ignored) {
log.warn("Exception getting a list of trusted domains", ignored);
}
if (log.isInfoEnabled()) {
StringBuffer msg = new StringBuffer("Effective list of domains: ");

```

```

boolean first = true;
for (String domain : res) {
    if (!first) {
        msg.append(", ");
    } else {
        first = false;
    }
    msg.append(domain);
    if (agentDomain.equals(domain)) {
        msg.append("(REQUESTED)");
    }
}
log.info(msg);
}
return res;
}
public static Serializable unwrapSerialized(String str) {
    Serializable ret = null;
    try {
        ObjectInputStream ois = new ObjectInputStream(new ByteArrayInputStream(new
BASE64Decoder().decodeBuffer(str)));
        ret = (Serializable)ois.readObject();
    } catch(IOException e) {
        log.info(str, e);
    } catch (ClassNotFoundException e) {
        log.info(str, e);
    }

    return ret;
}
}

```

Related concepts

[Sample client plug-in code](#) on page 304

The following is the code for a sample client heartbeat plug-in that is also a subject attribute provider plug-in.

Report log views

Report developers can design custom reports against NextLabs policy activity data that has been written to the Activity Journal database.

Several public views have been created to transform data so that queries can be run against the report schema for third-party reporting tools, such as SAP® Crystal Reports®. This section describes this public schema.

Overview of report log views

The report log views are represented in the following figure. Information about the columns for each view is discussed in the View details section.

policy_log_v1	
id	numeric (19)
timeStamp	datetime
month	numeric (19)
day	numeric (19)
host_name	varchar (255)
user_name	varchar (255)
user_id	int (19)
application_name	varchar (255)
action	char (2)
policy_full_name	varchar (512)
policy_decision	varchar (512)
log_level	numeric (10)
source_resource_name	varchar (512)
source_resource_owner_id	varchar (255)
source_resource_prefix	varchar (512)
source_resource_path	varchar (512)
source_resource_short_name	varchar (512)
source_resource_create_date	numeric (19)
target_resource_name	varchar (512)

policy_custom_attribute_v1	
id	numeric (19)
policy_log_id	numeric (19)
attribute_name	varchar (255)
attribute_value	varchar (4000)

policy_obligation_log_v1	
id	numeric (19)
policy_log_id	numeric (19)
name	varchar (256)
attribute_one	varchar (1024)
attribute_two	varchar (1024)
attribute_three	varchar (1024)

Figure 101: Overview of report log views

Related concepts

[View Details](#) on page 315

This section provides more detailed information for the table column values associated with each of the Policy Log views.

Policy Log views

The main Policy Log view is `policy_log_v1`. This base view contains data related to policy enforcement, for example, the time of policy enforcement, impacted resources, actions, the policy name, the policy decision, and so on.

Two other views are associated with `policy_log_v1`: `policy_obligation_log_v1` and `policy_custom_attribute_v1`. The policy obligation log view contains information about custom obligations associated with a policy event. The attributes that are collected are dependent on each custom obligation (for example, for a system encryption custom obligation, this could be the path or filename of the encrypted file). The policy custom attribute log reflects resource attributes that are associated with the policy event, for example, file tags/classifications associated with resources, or attributes associated with subjects (such as citizenship for users).

View Details

This section provides more detailed information for the table column values associated with each of the Policy Log views.

Related concepts

[Overview of report log views](#) on page 314

Policy Log view

The following table provides column value information for `policy_log_v1`.

Table 58: policy_log_v1

Column Name	Description	Value Type	Character
id	Record identifier	numeric	19
timestamp	Time policy enforcement occurred	datetime	

Column Name	Description	Value Type	Character
month	Timestamp for month of policy enforcement	numeric	19
day	Timestamp for day of policy enforcement	numeric	19
host_name	Fully Qualified Domain Name (FQDN) of host where policy enforcement occurred	varchar	Varies with maximum 255
user_name	User for whom policy enforcement occurred	varchar	Varies with maximum 255
user_sid	SID for user for whom policy enforcement occurred	numeric	19
application_name	Application used to access resource (full path to application)	varchar	Varies with maximum 255
action	The policy action: A two-character abbreviation of a policy action.	char	2
policy_full_name	The full name of the policy, including the path in policy tree.	varchar	Varies with maximum 512
policy_decision	The enforcement decision: D: Deny A: Allow.	varchar	512
log_level	The logging level: 1: user events, 2: application events, 3: system events	int	10
source_resource_name	The path to the resource source for the policy. For example: file:///mydesktop]/protected/file.doc.	varchar	Varies with maximum 512
source_resource_owner_id	The SID of the user who owns the resource.	varchar	Varies with maximum 255
source_resource_prefix	The type of resource component in the policy: file, device, portal, or sap object.	varchar	Varies with maximum 512
source_resource_path	The location where resource is stored. This differs from the full path to the resource, as in source_resource_name, as it can be a path to a directory defined in the policy. For example / [mydesktop]/protected/	varchar	Varies with maximum 512
source_resource_short_name	The filename of the source resource. For example, file.doc	varchar	Varies with maximum 512

Column Name	Description	Value Type	Character
source_resource_create_date	The date the source resource was created.	numeric	19
target_resource_name	If the resource is a policy target, as in a copy to policy, the name of the target resource. This is displayed as the full path to the file. For example: file:/// [mydesktop]/protected/upload/file.doc	varchar	Varies with maximum 512

Policy Obligation Log view

The following table provides column value information for `policy_obligation_log_v1`.

Table 59: policy_obligation_log_v1

Column Name	Description	Value Type	Character Length
id	Record identifier	numeric	19
policy_log_id	The ID of the associated policy event record in the <code>policy_log_v1</code> view	numeric	19
name	Name of the custom obligation	varchar	Varies with maximum 255
attribute_one	Attribute of custom obligation (specific to obligation)	varchar	Varies with maximum 1024
attribute_two	Attribute of custom obligation (specific to obligation)	varchar	Varies with maximum 1024
attribute_three	Attribute of custom obligation (specific to obligation)	varchar	Varies with maximum 1024

Policy Custom Attribute view

The following table provides column value information for `policy_custom_attribute_v1`.

Table 60: policy_custom_attribute_v1

Column Name	Description	Value Type	Character Length
id	Maintained for backward compatibility with prior versions of Reporter. The value is always 0.	numeric	22
policy_log_id	The ID of the associated policy event record in the <code>pol_log_v1</code> view	numeric	19
attribute_name	Name of the attribute, for example, “security_classification”	varchar	Varies with maximum of 4000

Column Name	Description	Value Type	Character Length
attribute_value	Value of the attribute, for example, “restricted”	varchar	Varies with maximum of 4000
attribute_type	Type of the attribute.	varchar	Varies with maximum 255

Actions

The following table provides a list of action codes that may display in the action column in `policy_log_v1`.

Actions

Table 61: Action list

Code	Action	Code	Action
Ca	Change Attributes	Ac	Access restricted configuration file
As	Abnormal Agent Shutdown	Ai	Access restricted log files
Cs	Change Security	Ab	Access restricted binaries
Co	Copy	Ib	Invalid bundle
Cp	Paste	Br	Bundle Received
De	Delete	Ap	Access agent bundle
Em	Embed	Ex	Export
Mo	Move	At	Attach
Pr	Print	Ru	Run
Op	Open	Av	Audio/Video
Ed	Edit	Me	Meeting
Rn	Rename	Ps	Presence
Se	Send Email	Sh	Share
Si	Send IM	Re	Record
Au	Start Agent	Qu	Ask question
Ad	Stop Agent	Vo	Voice
Uu	User Login	Vi	Video
Ud	User logout	Jo	Join

Reports in NextLabs Reporter

Use the Reporter tool to view examples of the kinds of data stored in the Policy Logging views.

To do so, run a report, and drill down to its event details, as shown in the following figure.

The base view, `policy_log_v1`, provides information that populates the top-level report. When you click a specific policy enforcement event, a Log Details report opens and displays the details of that event.

The two related views, `policy_custom_attribute_v1` and `policy Obligation_log_v1`, populate the Custom Attributes and Obligation sections of the Log Details report.

The screenshot shows a 'Report Detail' window with two main sections: 'Custom Attribute' and 'Obligation'. The 'Custom Attribute' section is a table with columns 'DATE' and 'Value'. The 'Obligation' section is a table with columns 'POLICY_DECISION' and 'ACTION'. Both sections show multiple rows of data.

DATE	Value
2019-01-07 18:56	prod_name Exchange Email
2019-01-07 18:56	created_by barney.gumble
2019-01-07 18:55	category security
2019-01-07 18:55	priority P0
2019-01-07 18:55	ticket_id 2206
2019-01-07 18:54	description Heartbleed critical patch update has not been applied
2019-01-07 18:54	severity Major Loss of Service
2019-01-07 18:35	helpdesk Helpdesk
2019-01-07 18:35	assigned_prod_area Exchange Email
2019-01-07 18:20	department IT
2019-01-07 18:20	roles Support Representative
2019-01-07 18:20	Obligation LOG, notifyViolation message Policy Violation: Security Vulnerability \${ticket_id} can only be accessed by the owner \${assigned_to} ticket_id \$from.ticket_id assigned_to \$from.assigned_to policy_model_id 14
	id chris.webber@hdesk.com

POLICY_DECISION	ACTION
Denied	View
Denied	RUN
Denied	View
Denied	RUN
Denied	View
Denied	RUN
Denied	View
Denied	RUN
Denied	View
Denied	RUN

Figure 102: Policy Log Data in a NextLabs Policy Activity Report

Using the sample enforcer

This section explains how to use the sample enforcer, which illustrates the basic controls available in SDKs.

About the sample enforcer

The sample enforcer is a very simple application called File Manager.

The enforcer includes the source C++ project for this application (`SampleEnforcer.vcproj`), so you can make changes, compile, and then see the results. The primary value of the File Manager is to demonstrate how the APIs work in a real application; but you can also use the project as the basis for custom enforcers, by expanding it with additional calls and recompiling it.

The source code for this sample enforcer, with annotations, is provided in the *C and C++ API reference* section.

Related concepts

[C and C++ API reference](#) on page 328

This section provides a reference of API calls for C and C++.

[C SDK and sample enforcer files](#) on page 322

The following figure shows the contents of the two ZIP archives—one for the C SDK, and one for the sample enforcer.

Related tasks

[Setting up your development environment for the C SDK](#) on page 321

To set up your environment for the C SDK, copy the SDK files to your development environment.

Sample enforcer files

The Sample Enforcer directory contains all the files required for your sample enforcer application.

These include:

- `SampleEnforcer.vcproj`: This is the main project file for Visual C++ projects generated using an application wizard. It contains information about the version of Visual C++ that generated the file, and information about the platforms, configurations, and project features selected with the application wizard.
- `sample_enforcerSampleEnforcer.h`: This is the main header file for the application. It includes other project specific headers (including `Resource.h`) and declares the `CSampleEnforcerApp` application class.
- `SampleEnforcer.cpp`: This is the main application source file that contains the application class `CSampleEnforcerApp`.
- `SampleEnforcer.rc`: This is a listing of all of the Microsoft Windows resources that the program uses. It includes the icons, bitmaps, and cursors that are stored in the RES subdirectory. This file can be edited directly in Microsoft Visual C++. Your project resources are in 1033.
- `res\SampleEnforcer.ico`: This is the icon file used as the application's icon. It is included by the main resource file, `SampleEnforcer.rc.res\SampleEnforcer.rc2`. This file contains resources that are not edited by Microsoft Visual C++. You should place all resources not editable by the resource editor in this file.

Other standard files

The following are the Additional files.

- `Stdafx.h`, `Stdafx.cpp`: These files are used to build a precompiled header (PCH) file named `SampleEnforcer.pch` and a precompiled types file named `StdAfx.obj`.
- `Resource.h`: This is the standard header file, which defines new resource IDs. Microsoft Visual C++ reads and updates this file.

Dialog class

The Sample Enforcer folder also contains one dialog class, `CSampleEnforcerDlg`.

This is contained in two files, `SampleEnforcerDlg.h` and `SampleEnforcerDlg.cpp`. This class defines the behavior of your application's main dialog. The dialog's template is in `SampleEnforcer.rc`, which can be edited in Microsoft Visual C++.

Other features

- The application includes support for ActiveX controls.
- The application wizard uses "TODO:" to indicate parts of the source code you should add to or customize.
- If your application uses MFC in a shared DLL, you will need to redistribute the MFC DLLs. If your application is in a language other than the operating system's locale, you will also have to redistribute the corresponding localized resources `MFC80XXX.DLL`. For more information on both of these topics, please see the section on redistributing Visual C++ applications in MSDN documentation.

Compiling and starting the sample enforcer

Before you use the sample enforcer, you must compile the code, either as delivered, or with modifications.

You do this from the main project file, `SampleEnforcer.vcproj`.

To start the sample enforcer, locate and double-click the executable, `SampleEnforcer.exe`. This displays the File Manager interface, shown in the following figure.

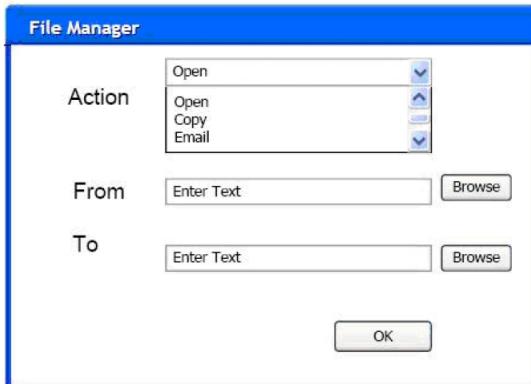


Figure 103: File Manager interface

! **Important:** You must install the SDK before you can open the sample enforcer. If you see an error regarding missing DLLs, the SDK is not installed.

The sample enforcer, called File Manager, demonstrates how to enforce a number of basic actions, such as Open, Copy, Move, and Delete, on file-type resources. It also includes controls specifying the source location of a resource and, for appropriate actions, such as Move, the target location.

Installing the Policy Adapter SDK

This section provides instructions for installing the Policy Adapter SDK.

The installation instructions differ, depending on whether you install the Java SDK, the C SDK, or the REST API.

The Policy Adapter SDK is planned to be deprecated in an upcoming release.

Related concepts

[PEP development process](#) on page 259

Building PEPs involves these steps.

Platform requirements

To test the Policy Adapter SDK, you must have an instance of Control Center and a Policy Controller installed.

About Policy Controllers

Policy Controllers are the generic portion of policy enforcers that reside on any host, server or desktop, where you want to enforce an application or system.

Policy Controllers provide a number of management, policy decision-making, event logging, and other functions that are independent of the kind of application or system being enforced upon. In a live system, Policy Controllers work with a policy adapter, which is the custom software that intercepts the events of the application or system being enforced, and carries out the policy enforcement decisions of Policy Controllers. For installation instructions, see NextLabs Control Center Installation and Upgrade Guide.

Setting up your development environment for the C SDK

To set up your environment for the C SDK, copy the SDK files to your development environment.

1. Obtain the `PolicyAdapterSDK.zip` and `SampleAdapter.zip` files.
2. Copy the files to a convenient location in your development environment.
3. Optional: Sample Enforcer for the C SDK: See the *About the sample enforcer* section.

Note: The REST API is supported by the Policy Controller for Java only. No installation is required because the REST API is part of the Policy Controller for Java.

Related concepts

[About the sample enforcer](#) on page 319

The sample enforcer is a very simple application called File Manager.

C SDK and sample enforcer files

The following figure shows the contents of the two ZIP archives—one for the C SDK, and one for the sample enforcer.

The C SDK archive contains all the required libraries (DLL and LIB files) along with the header file. For a description of the sample enforcer files, see the *About the sample enforcer* section.

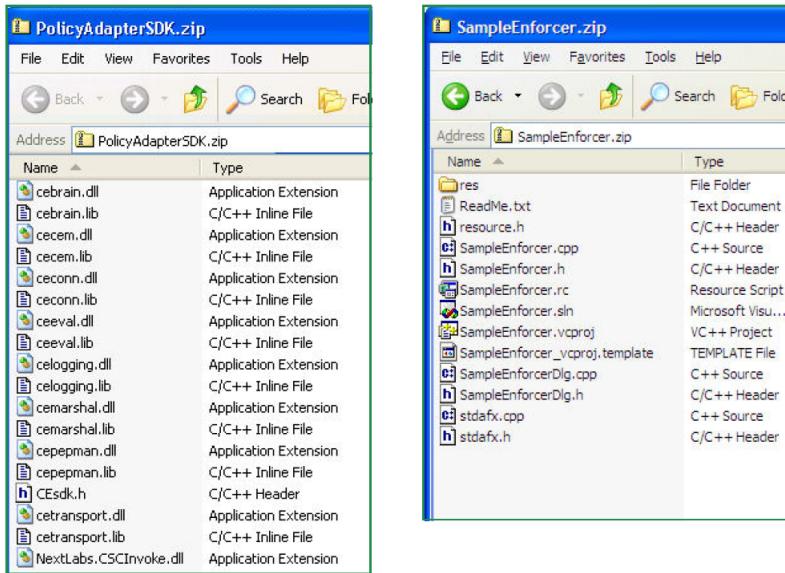


Figure 104: Custom Adapter C SDK and sample enforcer files

Related concepts

[About the sample enforcer](#) on page 319

The sample enforcer is a very simple application called File Manager.

Installing and configuring the Java RMI SDK

This section explains how to install and configure the Java RMI (Remote Method Invocation) SDK.

Obtain the PolicyAdapterJavaSDK.zip file from NextLabs Support. This archive contains the following files:

- nlJavaSDK2.jar: The archive file that contains the API classes
- nlJavaSDK2-doc.jar: The archive file that contains the Javadoc describing the API
- JavaSDKService.properties: The properties file used to configure the SDK
- SDKTest.java: Sample program file that shows use of the API

1. Copy nlJavaSDK2.jar to the following Policy Controller folder:

<install directory>\NextLabs\Policy Controller\jbservice\jar\javasdk

2. Copy JavaSDKService.properties to the following folder:

<install directory>\NextLabs\Policy Controller\jbservice\config

3. Provide the following information in JavaSDKService.properties:

```
name = JavaSDKService
jar-path = <install directory>/Policy Controller/jbservice/jar/javasdk/
nlJavaSDK2.jar
friendly_name = JavaSDK Service
description = JavaSDK Service
```

```

category = API
rmi_registry_port = <port number>

```

- Configure a firewall exception for the RMI registry port, which is used for communication between the client and server. The default port number is 1099.

Java SDK files

The Java SDK provides a set of classes, which you use to develop your policy enforcement point. The following figure shows the classes in `nlJavaSDK2.jar`.

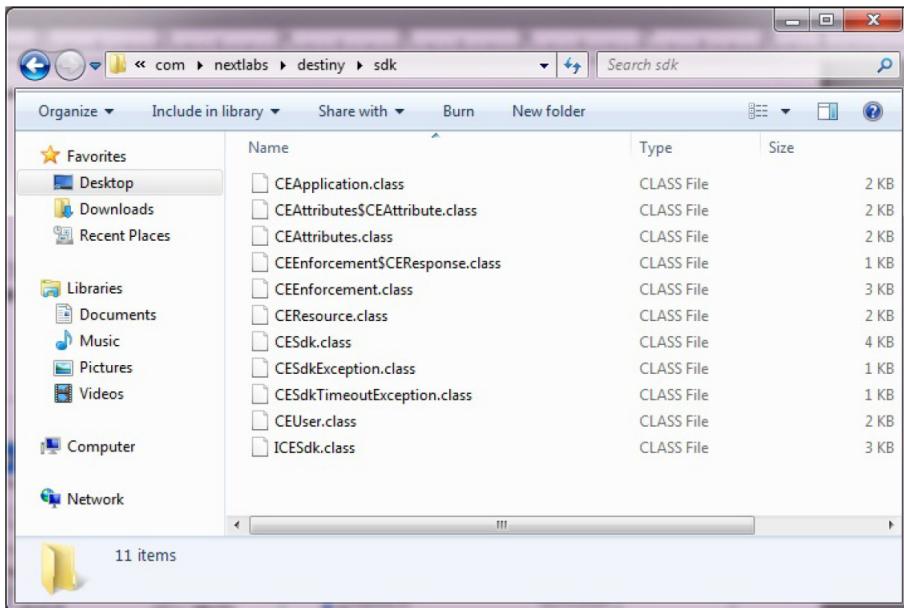


Figure 105: Classes in Custom Enforcer Java SDK

Set up Java Policy Controller Authentication in on-premise deployment

This section describes how to enable authentication for Java Policy Controller's REST API in on-premise deployment.

The setup is using Apache HTTP server and works under Linux environment. This documentation uses RHEL 7 as an example.

- Control Center instance: Control Center is required for the deployment, (only 8.0 version onwards are supported) in this document the Control Center server's hostname will be referred as `{CC_HOST}`.
- Hostname: The server where Java Policy Controller is installed should have a resolvable hostname/FQDN, this document uses `{JPC_HOST}` to represent the hostname value.
- SSL Certificate: A valid HTTP SSL Certificate is required, you can purchase one or generate a self-signed certificates. The certificate's subject CN should contain the server's hostname/FQDN (`{JPC_HOST}`). To generate self-signed certificates:

- Use following command: (replace `{JPC_HOST}` with actual value):

```

openssl req -x509 -nodes -days 3650 -newkey rsa:2048 \
-subj "/C=US/ST=CA/L=San Mateo/O=NextLabs/OU=CompliantEnterprise/
CN={JPC_HOST}" \
-keyout /tmp/server.key -out /tmp/server.crt

```

- Move the file `server.key` and `server.crt` to a folder and set proper file permission to the files. In this document, the location of the folder containing the two files is referred as `{SSL_PATH}`. Beware of file permissions of the certificates, they must be readable by the Apache HTTP Server.

- RHEL7's yum repos: The deployment described in this document using RHEL 7 requires additional yum repos enabled including rhel-server-optinal, rhel-server-extras, rhel-server-rhscl and epel. Also, building a module (mod_auth_cas) for Apache HTTP Server is required. Check the IT policy whether those are permitted before trying the deployment. Enabling yum repo is out of the scope of this documentation. Refer to RedHat documentation.
 - Install Java Policy Controller in Tomcat or JBoss EAP server. For instructions, see NextLabs Control Center Installation and Upgrade Guide.
1. Verify the REST API is running properly by access `http://[JPC_HOST]:58080/dpc/authorization/pdp`. You should be able to see a web page containing help content for the REST APIs.
 2. Install Apache HTTP Server and required Apache modules. Apache HTTP server version 2.4 is required, and required Apache modules are:

- ssl_module
- perl_module
- auth_cas_module

Apache HTTP Server 2.4 is in RHEL7's rhel-server-rhscl repo, for how to enable "rhel-server-rhscl" repo and installation of the package, you can refer to <https://www.softwarecollections.org/en/scls/rhscl/httpd24/>

To install Apache HTTP Server 2.4, you can use following command: (enable required yum repos first)

```
sudo yum install httpd24
```

 **Note:** The apache config files are in: /opt/rh/httpd24/root/etc/httpd and the service name is httpd24-httpd

This documentation uses {APACHE_CONFIG_ROOT} to represent the apache config path (/opt/rh/httpd24/root/etc/httpd).

3. Install the ssl_module for Apache HTTP Server 2.4, you can use following command: (enable required yum repos first)

```
sudo yum install httpd24-mod_ssl
```

4. Install perl_module for Apache HTTP Server 2.4, you can use following command: (enable required yum repos first)

```
sudo yum install rh-perl520-mod_perl
```

5. Build and install the auth_cas_module for Apache HTTP Server 2.4 in RHEL 7:

- a. Install required packages to build the module.

```
sudo yum install httpd24-httpd-devel gcc openssl-devel automake libtool libcurl-devel
```

- b. Download mod_auth_cas source.

If your Control Center's Web SSL certificates are valid (not self-signed), you can use official mod_auth_cas 1.1 release available on the GitHub website.

Normally on-premise deployment's Control Center server's web certificates are self-signed. Official 1.1 release doesn't support self-signed HTTPS CAS server while official releases of mod_auth_cas prior of 1.1 have bug working with Apache HTTP Server 2.4. But klausdieterkrannich's fork fixed the bug and works fine with Apache HTTP Server 2.4.

```
wget -P /tmp https://github.com/klausdieterkrannich/mod_auth_cas/archive/master.zip
cd /tmp
unzip master.zip
# under tmp there will be a folder called mod_auth_cas-master
cd mod_auth_cas-master
```

6. Build and Install the module.

```
# under /tmp/mod_auth_cas-master folder  
autoreconf -ivf  
scl enable httpd24 bash  
.configure && make && sudo make install
```

The following message appears:

```
Libraries have been installed in: /opt/rh/httpd24/root/usr/lib64/httpd/  
modules
```

7. In the /opt/rh/httpd24/root/usr/lib64/httpd/modules folder you will find the mod_auth_cas module file: mod_auth_cas.so.

 **Note:** This documentation uses {MOD_AUTH_CAS_SO} to represent the location of mod_auth_cas module file (/opt/rh/httpd24/root/usr/lib64/httpd/modules/mod_auth_cas.so).

8. Configure Apache HTTP Server using the following commands:

a. Remove unnecessary files:

```
# replace {APACHE_CONFIG_ROOT} with actual path  
cd {APACHE_CONFIG_ROOT}  
sudo rm -f ./conf.d/welcome.conf ./conf.d/userdir.conf ./conf.d/ssl.conf
```

b. Create CAS cookie folder:

```
cd ` {APACHE_CONFIG_ROOT}`  
sudo mkdir cas && sudo chmod 777 cas
```

c. Create Perl scripts:

```
# replace {APACHE_CONFIG_ROOT} with actual path  
cd ` {APACHE_CONFIG_ROOT}`  
sudo mkdir -p perl/JPC && sudo chmod -R 775 perl/JPC
```

9. Create a file under {APACHE_CONFIG_ROOT}/perl named startup.pl with content: (replace {APACHE_CONFIG_ROOT} with actual path):

```
use lib qw({APACHE_CONFIG_ROOT}/perl);  
1;
```

10. Create a file under {APACHE_CONFIG_ROOT}/perl/JPC named Unauthorized.pm with content:

```
package JPC::Unauthorized;  
use strict;  
use warnings;  
use Apache2::RequestRec ();  
use Apache2::RequestIO ();  
use Apache2::Response ();  
use APR::Table ();  
use Apache2::Const -compile => qw(OK AUTH_REQUIRED);  
sub handler {  
    my $r = shift;  
    my $response_text = '';  
    if ($r->headers_in->{Accept} =~ /xml/) {  
        $r->content_type('application/xml');  
        $response_text = 'AccessDeniedAccess Denied';  
    } else {  
        $r->content_type('application/json');  
        $response_text = '{"Error":{"Code":"AccessDenied","Message":"Access  
Denied"}}';  
    }  
}
```

```

    }
    $r->custom_response(Apache2::Const::AUTH_REQUIRED, $response_text);
    return Apache2::Const::AUTH_REQUIRED;
}
1;

```

11.Create JPC conf file under {APACHE_CONFIG_ROOT}/conf.d named jpc.conf with content: (replace {MOD_AUTH_CAS_SO}, {APACHE_CONFIG_ROOT}, {CC_HOST} and {JPC_HOST} with actual values):

```

<IfModule !mod_ssl.c>
LoadModule ssl_module modules/mod_ssl.so
</IfModule>
Listen 443 https
SSLPassPhraseDialog builtin
SSLSessionCache shmc:/run/sslcache(512000)
SSLSessionCacheTimeout 300
SSLRandomSeed startup file:/dev/urandom 256
SSLRandomSeed connect builtin
SSLCryptoDevice builtin
SSLProtocol all -SSLv3
SSLProxyProtocol all -SSLv3
SSLHonorCipherOrder On
SSLCipherSuite ALL:!ADH:!EXP:!LOW:!RC2:!3DES:!SEED:!RC4:+HIGH:+MEDIUM
<IfModule !mod_auth_cas.c>
LoadModule auth_cas_module {MOD_AUTH_CAS_SO}
</IfModule>
CASVersion 2
CASCookiePath {APACHE_CONFIG_ROOT}/cas/
CASLoginURL https://{{JPC_HOST}}/401
CASValidateURL https://{{CC_HOST}}/cas/serviceValidate
CASValidateServer Off
PerlRequire {APACHE_CONFIG_ROOT}/perl/startup.pl
<VirtualHost *:443>
CustomLog logs/jpc.log combined
KeepAlive On
Servername {{JPC_HOST}}
SSLEngine On
SSLProxyEngine On
ProxyRequests Off
SSLCertificateFile {{SSL_PATH}}/server.crt
SSLCertificateKeyFile {{SSL_PATH}}/server.key
<Location /dpc/PDPCConnector/go>
AuthType CAS
require valid-user
ProxyPass "http://127.0.0.1:58080/dpc/authorization/pdp"
</Location>
<Location /dpc/authorization/pdp>
AuthType CAS
require valid-user
ProxyPass "http://127.0.0.1:58080/dpc/authorization/pdp"
</Location>
<Location /cas/>
ProxyPass "https://{{CC_HOST}}/cas/"
ProxyPassReverse "https://{{CC_HOST}}/cas/"
</Location>
<Location /401>
SetHandler perl-script
PerlResponseHandler JPC::Unauthorized
</Location>
</VirtualHost>

```

12.Start the server using the following command:

```
sudo service httpd24-htpd start
```

13.Optional: To configure the server to start when the system starts, use this command:

```
sudo systemctl enable httpd24-httpd
```

Configuring attribute plug-ins

Attribute plug-ins can be configured for resource and subject attributes as needed.

The Embedded PDP loads and registers attribute plug-ins on application startup. Plug-ins are configured in the `jservice` folder of the `dpc` directory of the Policy Controller.

The `dpc` directory must be configured.

1. Using Java, write the plug-in, and package it as a JAR file.

2. Place the JAR files in the following location:

```
\dpc\jservice\jar\<plug-in folder>
```

3. Create a properties file that has the name and location of the plug-in. The Policy Controller uses this file to locate the plug-in.

4. Place the properties file in the following location:

```
\dpc\jservice\config
```

5. Place any the plug-in dependencies in the `WEB-INF/lib` folder of the application.

Related tasks

[Configuring the embedded Policy Controller for Java on JBoss servers](#) on page 271

The embedded Policy Controller for Java is supported on JBoss servers. This section explains how to set up the web application.

[Configuring the embedded Policy Controller for Java on Tomcat servers](#) on page 271

The embedded Policy Controller for Java is supported on Tomcat servers. This section explains how to set up the web application.

After you install

By default, the heartbeat rate of the desktop enforcer is set to 60 minutes, which is appropriate for a live production environment.

For testing and learning purposes, however, you should change this to 1 minute, which will allow you to define, deploy and test policies with shorter delays. Even better is to use the Push Deploy feature, which lets you refresh policies more quickly by ignoring the heartbeat altogether and deploying updated policies immediately.

Installing the PEP

After you have developed and tested a PEP (policy enforcement point), you can install it together with the Policy Controller on enforcement point hosts according to your enforcer's design.

Each PEP requires a Policy Controller installed on the same host, and the Policy Controller should be installed first.

There are no requirements for configuring your PEP to work with the Policy Controller. A PEP developed with the C SDK opens a connection to the Policy Controller through the Initialize call. A PEP developed with the Java SDK specifies the Policy Controller in the constructor of the CESdk class. A PEP developed with the REST API specifies the Policy Controller in its HTTPS request.

Required DLLs (C SDK)

Make sure that when you install your PEP code, written in C, to the server where you will be enforcing policies, you manually include the following DLLs, which are required for various adapter functions.

The following table summarizes the functions of these required DLLs.

Table 62: DLLs required by the Policy Adapter C SDK

DLL	Function
cebrain.dll	Implementations of cross-platform string operations, thread, thread pool, synchronization, etc.
cecem.dll	Miscellaneous. Currently it has CEString_XXX functions.
ceconn.dll	Provides SDK CEConn_XXX (connection) functions
ceeval.dll	Provides SDK CEEval_XXX (policy evaluation) functions
celogging.dll	Provides SDK event logging functions
cemarshal.dll	Provides marshal/unmarshal RPC call between PEP and PDP
cepepm.dll	Provides Management module on PEP side
cetransport.dll	Manages the socket transport layer. It is for the socket communication between PEP and PDP

These DLLs are provided in a ZIP archive.

C and C++ API reference

This section provides a reference of API calls for C and C++.

Related concepts

[About the sample enforcer](#) on page 319

The sample enforcer is a very simple application called File Manager.

Policy evaluation APIs

The CEEVALUATE APIs handle requests to the PDP to evaluate whether a given user action is governed by a current policy.

These APIs include:

- CEEVALUATE_CheckFile()
- CEEVALUATE_CheckMessageAttachment()
- CEEVALUATE_CheckPortal()
- CEEVALUATE_CheckResources()
- CEEVALUATE_CheckResourcesEx()

All the functions, except CEEVALUATE_CheckResourcesEx(), send a single policy evaluation query and receive a single response in one request to the Policy Controller. CEEVALUATE_Check- ResourcesEx() sends multiple policy evaluation queries and receives multiple responses in one request to the Policy Controller. This function also supports sending ad hoc PQL policies that can be evaluated in conjunction with, or instead of, the deployed policies.

The CEEVALUATE_FreeEnforcement() call has a special function of freeing the enforcement output of the other calls.

CEEVALUATE_CheckFile()

This call requests the Policy Controller to evaluate an operation that some user has performed on a file, to determine whether it is covered by a currently deployed policy.

Accordingly, its arguments provide information about the resource, the user, the PC, and the operation. The following table describes the valid arguments of this call.

Syntax:

```
CEResult_t CEEVALUATE_CheckFile(CEHandle handle,
CEAction_t operation,
```

```

CEString sourceFullName,
CEAttributes * sourceAttributes,
CEString targetFullName,
CEAttributes * targetAttributes,
CEint32 ipNumber,CEUser *user,
CEApplication *app,
CEBoolean performObligation,
CENoiseLevel_t noiseLevel,
CEEnforcement_t * enforcement,
CEint32 timeout_in_millisec);

```

Table 63: CEEVALUATE_CheckFile arguments

Name	Direction	Type	Description	Req/ Opt
handle	Input	CEHandle	The handle from CECNN_Initialize().	R
action	Input	CEaction	The action that was attempted on the file resource.	R
sourceFullName <input type="text"/>	Input	CEString	The fullname filename that uniquely identifies the source resource.	R
sourceAttributes	Input	CEAttributes	Associate attributes of the source.	R
targetFullName <input type="text"/>	Input	CEString	The fullname filename that uniquely identifies the target resource, if required by action.	O
targetAttributes	Input	CEAttributes	Associate attributes of the target, if required by action.	O
performObligation <input type="checkbox"/>	Input	CEBoolean	Specifies whether the obligation defined in the policy should be performed.	R
noiseLevel	Input	CENoiseLevel_t	Specifies the noise level to be used for this evaluation.	R
ipNumber	Input	CEInt32	IP address of machine where the action was performed. Must be in network byte order: For example, 127.0.0.1 = 1x(256^3) + 0x(256^2) + 0x(256) + 127 0.0.0.0 indicates local access.	R
user	Input	CEUser	User who performed the specified action. If this is not supplied, the default value from CECNN_Initialize is used.	O
app	Input	CEApplication	The application that is associated with the specified action. If this is not supplied, the value from CECNN_Initialize is used.	O

Name	Direction	Type	Description	Req/ Opt
timeout_in_milliseconds	Input	CEInt32	A nonnegative integer, which specifies the number of milliseconds the API should wait before returning a TIMEOUT result. To generate a synchronized blocking call, use the value CE_INFINITE.	O
Enforcement	Output	CEEenforcement_t	page 360) to free this value in all cases.	R

The CEEVALUATE_CheckFile return values table describes the allowable return values.

Table 64: CEEVALUATE_CheckFile return values

Name	Description
CE_RESULT_SUCCESS	The call completed successfully
CE_RESULT_INVALID_PARAMS	Invalid handle to the function call or missing required parameters
CE_RESULT_GENERAL_FAILED	General failure of the call
CE_RESULT_INVALID_COMBINATION	Invalid combination of operation and the source/target arguments
CE_RESULT_CONN_FAILED	No connection to the server
CE_RESULT_TIMEOUT	Timeout after the time specified by the user has elapsed
CE_RESULT_INVALID_ACTION_ENUM	Out of range action is passed into the function call
CE_RESULT_EMPTY_SOURCE	Invalid source is passed into the function call
CE_RESULT_MISSING_MODIFIED_DATE	Key “CE_ATTR_LASTWRITE_TIME” is missing from the attributes
CE_RESULT_INVALID_EVAL_ACTION	Unsupported action is passed into the corresponding EVAL function class. (e.g. IM_FILE in the checkPortal)
CE_RESULT_EMPTY_SOURCE_ATTR	NULL is passed into the sourceAttributes argument
CE_RESULT_EMPTY_ATTR_KEY	Null or empty string is passed into the attributes keys
CE_RESULT_EMPTY_ATTR_VALUE	Null or empty string is passed into the attribute values
CE_RESULT_NULL_CEHANDLE	Invalid CEHandle is passed into the function

Sources targets, and attributes

Depending on the action supplied to the function call, the requirement of the source, target and attributes are different.

The valid actions and corresponding sourceFileName and targetFileName requirement are shown in the following figure.

Table 65: Actions and source/target resources

Action	Source URL	Target URL
CE_ACTION_READ	Required	Optional
CE_ACTION_DELETE	Required	Optional
CE_ACTION_MOVE	Required	Required
CE_ACTION_COPY	Required	Required
CE_ACTION_EDIT	Required	Optional
CE_ACTION_EXPORT	Required	Optional
CE_ACTION_ATTACH	Required	Optional

CEEVALUATE_CheckMessageAttachment()

This call requests the Policy Controller to evaluate a user action that involves a document attached to a message such as an email or instant message, to determine whether it is covered by a currently deployed policy.

Accordingly, its arguments provide information about the attached file, the message sender, the message recipient, the PC, and the action. The following tables describe the valid arguments of this call and the allowable return values.

Syntax:

```
CEResult_t
CEEVALUATE_CheckMessageAttachment(CEHandle handle,
CEAction_t operation, CESTring sourceFullFileName,
CEAttributes * sourceAttributes, CEint32 numRecipients,
CEString *recipients, CEint32 ipNumber, CEUser *user,
CEAttributes * userAttributes,
CEApplication *app,
CEAttributes * appAttributes,
CEBoolean performObligation,
CENoiseLevel_t noiseLevel,
CEEnforcement_t * enforcement,
CEint32 timeout_in_millisec);
```

Table 66: CEEVALUATE_CheckMessageAttachment arguments

Name	Direction	Type	Description	Req/ Opt
handle	Input	CEHandle	The handle from CECONN_Initialize().	R
action	Input	CEAction	The action that was attempted on the message. The valid actions for this call are CE_ACTION_REPLY, CE_ACTION_FORWARD, and CE_ACTION_NEW_EMAIL.	R
sourceFullFileName	Input	CEString	The URL of the source portal resource.	R
sourceAttributes	Input	CEAttributes *	Attributes of the source.	O

Name	Direction	Type	Description	Req/ Opt
handle	Input	CEHandle	The handle from CECONN_Initialize().	R
action	Input	CEAction	The action that was attempted on the message. The valid actions for this call are CE_ACTION_REPLY, CE_ACTION_FORWARD, and CE_ACTION_NEW_EMAIL.	R
sourceFullFileName	Input	CEString	The URL of the source portal resource.	R
sourceAttributes	Input	CEAttributes *	Attributes of the source.	O

Table 67: CEEVALUATE_CheckMessageAttachment return values

Name	Description
CE_RESULT_SUCCESS	The call completed successfully
CE_RESULT_INVALID_PARAMS	Invalid Handle to the function call or missing required parameters
CE_RESULT_GENERAL_FAILED	General failure of the call
CE_RESULT_INVALID_COMBINATION	Invalid combination between the operation and the source/target arguments
CE_RESULT_CONN_FAILED	No connection to the server
CE_RESULT_TIMEOUT	Timeout after the time specified by the user has elapsed
CE_RESULT_INVALID_ACTION_ENUM	Out of range action is passed into the function call
CE_RESULT_EMPTY_SOURCE	Invalid source is passed into the function call
CE_RESULT_MISSING_MODIFIED_DATE	Key “CE_ATTR_LASTWRITE_TIME” is missing from the attributes
CE_RESULT_INVALID_EVAL_ACTION	Unsupported action is passed into the corresponding EVAL function class (for example, IM_FILE in the checkPortal)
CE_RESULT_EMPTY_SOURCE_ATTR	Null is passed into the sourceAttributes argument
CE_RESULT_EMPTY_ATTR_KEY	Null or empty string is passed into the attributes keys
CE_RESULT_EMPTY_ATTR_VALUE	Null or empty string is passed into the attribute values
CE_RESULT_NULL_CEHANDLE	Invalid CEHandle is passed into the function

CEEVALUATE_CheckPortal()

This call requests the Policy Controller to evaluate an operation that some user has performed on some portal resource, to determine whether it is covered by a currently deployed policy.

Accordingly, its arguments provide information about the resource (source and, optionally, target), the user, the PC, and the operation. The following tables describe the valid arguments of this call and the allowable return values.

Syntax:

```
CEResult_t CEEVALUATE_CheckPortal(CEHandle handle,
CEAction_t operation,
CEString sourceURL,
CEAttributes * sourceAttributes,
CEString targetURL,
CEAttributes * targetAttributes,
CEint32 ipNumber,
CEUser *user,
CEBoolean performObligation,
CENoiseLevel_t noiseLevel,
CEEnforcement_t * enforcement,
CEint32 timeout_in_millisec);
```

Table 68: CEEVALUATE_CheckPortal arguments

Name	Direction	Type	Description	Req/ Opt
		C / C++		
handle	Input	CEHandle	The handle from CECNN_Initialize().	R
action	Input	CEAction	The action that was attempted on the portal resource.	R
sourceURL	Input	CEString	The URL of the source portal resource	R
targetURL	Input	CEString	The URL of the target portal resource, if appropriate	R
perform Obligation	Input	CEBoolean	Specifies which obligation should be performed	R
source Attributes	Input	CEAttributes *	Associate attributes of the source.	O
target Attributes	Input	CEAttributes *	Associate attributes of the target, if any.	O
noiseLevel	Input	CENoiselevel_t	Desirable noise level to be used for this evaluation.	R
ipNumber	Input	CEInt32	The IP address of client machine.	O
user	Input	CEUser	The User who accessed this URL.	R
timeout_in_millisec		CEInt32	A nonnegative integer, which specifies the number of milliseconds the API should wait before returning a TIMEOUT result.	R
			To generate a synchronized blocking call, use the value CE_INFINITE.	

Name	Direction	Type	Description	Req/ Opt
Enforcement	Output	CEEnforcement	The result of the policy for enforcement. This is where delegated obligations are defined. You should use the CEEvaluate_FreeEnforcement call (page 360) to free this value in all cases.	R

Table 69: CEEVALUATE_CheckPortal return values

Name	Description
CE_RESULT_SUCCESS	The call completed successfully
CE_RESULT_INVALID_PARAMS	Invalid Handle to the function call or missing required parameters
CE_RESULT_GENERAL_FAILED	General failure of the call
CE_RESULT_INVALID_COMBINATION	Invalid combination between the operation and the source/target arguments
CE_RESULT_CONN_FAILED	No connection to the server
CE_RESULT_TIMEOUT	Timeout after the time specified by the user has elapsed
CE_RESULT_INVALID_ACTION_ENUM	Out of range action is passed into the function call
CE_RESULT_EMPTY_SOURCE	Invalid source is passed into the function call
CE_RESULT_MISSING_MODIFIED_DATE	Key “CE_ATTR_LASTWRITE_TIME” is missing from the attributes
CE_RESULT_INVALID_EVAL_ACTION	Unsupported action is passed into the corresponding EVAL function class (e.g., IM_FILE in the checkPortal).
CE_RESULT_EMPTY_SOURCE_ATTR	NULL is passed into the sourceAttributes argument
CE_RESULT_EMPTY_ATTR_KEY	Null or empty string is passed into the attributes keys
CE_RESULT_EMPTY_ATTR_VALUE	Null or empty string is passed into the attribute values
CE_RESULT_NULL_CEHANDLE	Invalid CEHandle is passed into the function

CEEVALUATE_CheckResources()

This call requests the Policy Controller to evaluate an operation that some user has performed on some document resource, to determine whether it is covered by a currently deployed policy.

Accordingly, its arguments provide information about the resource (source and, optionally, target), the user, the PC, and the operation. The following tables describe the valid arguments of this call and the allowable return values

Syntax:

```
CEResult_t CEEVALUATE_CheckResources(CEHandle handle,
const CEString operation,
const CEResource * source,
const CEAttributes * sourceAttributes, const CEResource * target,
```

```

const CEAttributes * targetAttributes, const CEUser *user,
CEAttributes * userAttributes, CEApplication *app, CEAttributes *
appAttributes, CESTring *recipients,
CEint32 numRecipients, const CEint32 ipNumber,
const CEBoolean performObligation,
const CENoiseLevel_t noiseLevel,
CEEEnforcement_t * enforcement,
const CEint32 timeout_in_millisec);

```

Table 70: CEEVALUATE_CheckResources arguments

Name	Direction	Type	Description	Req/ Opt
handle	Input	CEHandle	The handle from CECONN_Initialize().	R
operation	Input	CEString	The user action that was attempted on the resource.	R
source	Input	CEResource *	The source resource, expressed in CEResource type.	R
sourceAttributes	Input	CEAttributes *	Associate attributes of the source resource. May be null if the source has no associate attributes.	O
target	Input	CEResource *	The target resource, if any.	R
targetAttributes	Input	CEAttributes *	Associate attributes of the target. May be null if the target has no associate attributes.	O
user	Input	CEUser	The user who accessed the resource.	R
userAttributes	Input	CEAttributes *	Associate attributes of the user.	O
app	Input	CEApplication	The application that was used to access this resource.	R
appAttributes	Input	CEAttributes *	Associate attributes of the application.	O
recipients	Input	CEString *	A string array of recipients of the message with the data attachment.	O
numRecipients	Input	CEint32	The number of message recipients. It is optional, which means its value can be 0. Negative values are treated as 0.	O
ipNumber	Input	CEInt32	The IP address of client machine.	O
perform Obligation	Input	CEBoolean	Specifies which obligation should be performed.	R
noiseLevel	Input	CENoiselevel_t	The desirable noise level to be used for this evaluation.	R

Name	Direction	Type	Description	Req/ Opt
enforcement	Output	CEEnforcement	The result of the policy for enforcement. This is where delegated obligations are defined. You should use the CEEvaluate_FreeEnforcement call (page 360) to free this value in all cases.	R
timeout_in_millisec	Input	CEInt32	A nonnegative integer, which specifies the number of milliseconds the API should wait before returning a TIMEOUT result. To generate a synchronized blocking call, use the value CE_INFINITE.	R

Table 71: CEEVALUATE_CheckResources return values

Name	Description
CE_RESULT_SUCCESS	The call completed successfully
CE_RESULT_NULL_CEHANDLE	Invalid CEHandle passed into the function

CEEVALUATE_CheckResourcesEx()

This call requests the Policy Controller to evaluate multiple policy queries about user operations performed on document resources, to determine whether they are covered by deployed policies.

You can also use this call to send ad hoc PQL policies to the Policy Controller. The following tables describe the valid arguments of this call and the allowable return values.

Syntax:

```
CEResult_t CEEVALUATE_CheckResourcesEx(CEHandle handle,
CERequest * requests,
CEint32 numRequests,
CEString additionalPQL,
CEEnforcement_t * enforcement,
CEint32 timeout_in_millisec);
```

Table 72: CEEVALUATE_CheckResourcesEx arguments

Name	Direction	Type	Description	Req/ Opt
handle	Input	CEHandle	The handle from CECOMP_Initialize().	R
requests	Input	CERequest	An array of CERequest objects.	R
numRequests	Input	CEint32	The number of requests.	R
additionalPQL	Input	CEString	The PQL policy to evaluate in addition to, or instead of, deployed policies.	O

Name	Direction	Type	Description	Req/ Opt
ignoreBuiltInPolicyInput	Input	CEBoolean	If true, the PQL policy specified in the additionalPQL argument is evaluated and the deployed policies are ignored. If false, the PQL and deployed policies are both evaluated.	O
ipNumber	Input	CEInt32	The IP address of client machine.	O
enforcement	Output	CEEEnforcement	The result of the policy for enforcement. This is where delegated obligations are defined. You should use	R
timeout_in_milliseconds	Input	CEInt32	A non-negative integer, which specifies the number of milliseconds the API should wait before returning a TIMEOUT result. To generate a synchronized blocking call, use the value CE_INFINITE.	R

Table 73: CEEVALUATE_CheckResourcesEx return values

Name	Description
CE_RESULT_SUCCESS	The call completed successfully
CE_RESULT_NULL_CEHANDLE	Invalid CEHandle is passed into the function

CEM_CreateResource()

This call creates a resource of a specified name and type, which is passed as a value for the source and target arguments in CEEVALUATE_CheckResources().

Syntax:

```
CEResource * CEM_CreateResource(const char * resourceName, const char * resourceType);
```

Table 74: CEM_CreateResource() arguments

Name	Direction	Type	Description
resourceName	Input	const char *	The name of the resource, for example, "C:\myFile.txt"
resourceType	Input	const char *	The type of resource. For example, files on disk are "fso" and SharePoint pages are "portal".

CEEVALUATE_FreeEnforcement()

This special API provides the basic service that frees a specified CE type CEEEnforcement_t, which is an output from the policy enforcement APIs, from the system.

The following table describes the valid arguments of this call.

Syntax:

```
CEResult_t  
CEEVALUATE_FreeEnforcement(CEEnforcement_t enforcement);
```

Table 75: CEEvaluate_FreeEnforcement arguments

Name	Direction	Type	Description
enforcement	Input	Special defined	Frees the enforcement_t result from the system. Should be called after any of the policy enforcement APIs.

String-handling APIs

The CEM APIs are basic service calls dedicated to handling CEtype strings.

They control certain functions that are required for mapping CE strings, used in the communication between the Policy Adapter and the Policy Controller, to standard C or C++ format.

CEM_AllocateString()

This call provides a basic service to allocate memory for mapping a CEtype string to a C or C++ string, so that it can be used by the system.

This call is required for every instance when you call an API that uses arguments of a CEString data type. In each case, the allocated string should be released once the call is made, using the CEM_FreeString call.

Syntax:

```
CEString CEM_AllocateString (const char * str);  
CEString CEM_AllocateString (const TCHAR * str);
```

Table 76: CEM_AllocateString arguments

Name	Direction	Type	Description
str	Input	const char */ const TCHAR*	Pointer to the original C standard string

Table 77: CEM_AllocateString return values

Name	Description
CEString	NULL if the call failed; nonNULL if the call was successful

CEM_FreeString()

This call frees a CE-type string from the system.

Syntax:

```
CEResult_t CEM_FreeString (CEString cestr);
```

Table 78: CEM_FreeString arguments

Name	Direction	Type	Description
cestr	Input	CEString	Specifies which CE-type string should be freed

Table 79: CEM_FreeString return values

Name	Description
CE_RESULT_SUCCESS	Call completed successfully
CE_RESULT_FAILED	Call resulted in general failure

Table 80: CEM_FreeString return values

Name	Description
CE_RESULT_SUCCESS	Call completed successfully
CE_RESULT_FAILED	Call resulted in general failure

CEM_ReallocateString()

This call reallocates the existing string to a new string.

Syntax:

```
CEResult_t CEM_ReallocateString (CEString cestr,  
const char * newstr);  
CEResult_t CEM_ReallocateString (CEString cestr,  
const TCHAR * newstr);
```

Table 81: CEM_ReallocateString arguments

Name	Direction	Type	Description
cestr	Input/Output	CEString	Specifies which CEtype string should be reallocated
newstr	Input	const char */ const TCHAR*	Specifies the new string to which cestr should be reallocated

Table 82: CEM_ReallocateString return values

Name	Description
CE_RESULT_SUCCESS	Call completed successfully
CE_RESULT_FAILED	Call resulted in general failure

CEM_GetString()

Basic service to retrieve the standard C string from the CE-type string.

Syntax:

```
C/C++ Windows:  
const char *  
CEM_GetString (CEString cestr);  
C/C++ Linux:  
const TCHAR *  
CEM_GetString (CEString cestr);  
C/C++ Java:  
string *  
CEM_GetString (CEString cestr);
```

Table 83: CEEvaluate_AllocateString arguments

Name	Direction	Type	Description
cestr	Input	CEString	CE-type string to retrieve

Table 84: CEM_AllocateString return values

Name	Description
CEString	NULL if the call failed due to invalid input; non-NULL if the call was successful

Security APIs

The CEPROTECT APIs described in this section handle the enforcer's security functions.

CEPROTECT_LockKey()

Protects the relevant registry key from being modified by any other process. This call is valid only on Windows.

Syntax:

```
CEResult_t CEPROTCT_LockKey (CEHandle handle,  
CEKeyRoot_t root,  
CEString key);
```

Table 85: CEPROTCT_LockKey arguments

Name	Direction	Type	Description
handle	Input	CEHandle	Connection handle from CECONN_initialize()
root	Input	CEKeyRoot_t	Root of the registry group containing the key to be protected
key	Input	CEString	Key to be protected, in the following format: level1\\level2\\level3\\key

Table 86: CEPROTCT_LockKey return values

Name	Description
CE_RESULT_SUCCESS	Call completed successfully
CE_RESULT_INVALID_PARAMS	Invalid Handle or root to the call
CE_RESULT_GENERAL_FAILED	Failed to lock the key
CE_RESULT_CONN_FAILED	No connection to the server
CE_RESULT_FUNCTION_NOT_AVAILABLE	This function is not available (when called from platform other than Windows)

CEPROTECT_UnlockKey()

Disables the protection provided by a registry key in the system by the CEPROTCT_LockKey call. This call is valid only on Windows.

Syntax:

```
CEResult_t CEPROTCT_UnlockKey (CEHandle handle,
```

```
CEKeyRoot_t root,
CEString key);
```

Table 87: CEPROTECT_UnLockKey arguments

Name	Direction	Type	Description
handle	Input	CEHandle	Connection handle from CECNN_initialize()
root	Input	CEKeyRoot_t	Root of the registry group containing the key to be unlocked
key	Input	CEString	Key to be unlocked, in the following format: level1\\level2\\level3\\key

Table 88: CEPROTECT_UnLockKey return values

Name	Description
CE_RESULT_SUCCESS	Call completed successfully
CE_RESULT_INVALID_PARAMS	Invalid Handle or root to the call
CE_RESULT_GENERAL_FAILED	Failed to lock the key
CE_RESULT_CONN_FAILED	No connection to the server
CE_RESULT_FUNCTION_NOT_AVAILABLE	This function is not available (when called from platform other than Windows)

Special data types

The Policy Adapter C SDK relies on a number of special data types, which are identifiable by their “CE” prefix and, for some, by a “_t” suffix as well. They include:

- CEString
- CEAtributes
- CEResult_t
- CEKeyRoot_t
- CEPROTECTMode
- CEHandle
- Ceint32
- CEBoolean
- CEAction_t
- CEEncforcement_t
- CENoiseLevel_t
- CEPEP_t
- CERequest

CEString

CEString is an opaque structure to the public API. The application developer needs to use the CEM_ APIs to manipulate it. The structure itself takes care of the variable length for efficient communication.

CEAtributes

One or more CEAtributes are passed with certain data types, such as users or resources. Each CEAtribute takes the form of a key and a value, both of which are CEStrings.

The following table lists all keys and required formats of their corresponding values.

Table 4.1034: CEAtributes, keys, and value formats

Table 89: CEAttributes, keys, and value formats

Key	Description	Attribute Of	Format of Value
CE_ATTR_CREATE_TIME	The time the resource was created. This is a standard Windows file property.	Resource	POSIX time in milliseconds
CE_ATTR_LASTACCESS_TIME	The most recent time when the resource was accessed by any user.	Resource	POSIX time in milliseconds
CE_ATTR_LASTWRITE_TIME	The most recent time when any additional data was written to the resource by any user.	Resource	POSIX time in milliseconds
CE_ATTR_OWNER_NAME	The name of the User who owns the resource.	Resource	string
CE_ATTR_OWNER_ID	The SID or UID of the user who owns the resource.	Resource	string
CE_ATTR_FULLPATH	The absolute path of the resource, in its source location.	Resource	string
CE_ATTR_LASTACTION	The most recent action that was performed on this resource, by any user.	User	string
CE_ATTR_POLICY_LASTUPDATE_TIME	The timestamp when current policy was last updated.	Policy	POSIX time in milliseconds
Delegated Obligations Only			
CE_ATTR_OBLIGATION_COUNT (OPTIONAL)	The count of number of delegated obligations associated with current policy.	Policy	Nonnegative number in CCString
CE_ATTR_OBLIGATION_NAME (Mandatory if CE_ATTR_OBLIGATION _COUNT is present)	The name of the Xth Delegated Obligation. This is the same as the NAME string passed in the obligation command line.	Policy	string

CEResult_t

The following table lists all supported result messages that may be returned in response to API calls.

Table 90: CEResult messages

Message	ID	Description
CE_RESULT_SUCCESS	0	The call completed successfully
CE_RESULT_GENERAL_FAILED	-1	Failed to complete the call
CE_RESULT_CONN_FAILED	-2	Failed to connect to the PDP server
CE_RESULT_INVALID_PARAMS	-3	Invalid parameters
CE_RESULT_VERSION_MISMATCH	-4	Incompatible API version

Message	ID	Description
CE_RESULT_FILE_NOT_PROTECTED	-5	Removing a protected file that's not in the protected list
CE_RESULT_INVALID_PROCESS	-6	Invalid process to be protected.
CE_RESULT_INVALID_COMBINATION	-7	Invalid combination between the operation and the source or target file
CE_RESULT_PERMISSION_DENIED	-8	Insufficient permission to perform an operation
CE_RESULT_FILE_NOT_FOUND	-9	File doesn't exist in the PDP record
CE_RESULT_FUNCTION_NOT_AVAILABLE	-10	Function is not available
CE_RESULT_TIMEOUT	-11	Time out from the API call
CE_RESULT_SHUTDOWN_FAILED	-12	On multi-threaded programming, outstanding thread still connecting to the PDP
CE_RESULT_INVALID_ACTION_ENUM	-13	Out of range action is passed into the function call
CE_RESULT_EMPTY_SOURCE	-14	Invalid source is passed into the function call
CE_RESULT_MISSING_MODIFIED_DATA	-15	Key "CE_ATTR_LASTWRITE_TIME" is missing from the attributes
CE_RESULT_NULL_CEHANDLE	-16	Invalid CEHandle is passed into the function
CE_RESULT_INVALID_EVAL_ACTION	-17	Unsupported action is passed into the corresponding EVAL function class. (for example, IM_FILE in the checkPortal)
CE_RESULT_EMPTY_SOURCE_ATTR	-18	NULL is passed into the sourceAttributes argument
CE_RESULT_EMPTY_ATTR_KEY	-19	Null or empty string is passed into the attributes keys
CE_RESULT_EMPTY_ATTR_VALUE	-20	Null or empty string is passed into the attribute values
CE_RESULT_EMPTY_PORTAL_USER	-21	NULL is passed into the user argument of the Portal evaluation
CE_RESULT_EMPTY_PORTAL_USERID	-22	NULL is passed into the userid field of the portal evaluation

CEKeyRoot_t

The table lists all supported values for the Key Root data type.

Table 91: Key root values

Message	ID	Description
CE_KEYROOT_CLASSES_ROOT	0	HKEY_CLASSES_ROOT
CE_KEYROOT_CURRENT_USER	1	HKEY_CURRENT_USER
CE_KEYROOT_LOCAL_MACHINE	2	HKEY_LOCAL_MACHINE
CE_KEYROOT_USERS	3	HKEY_USERS
CE_KEYROOT_CURRENT_CONFIG	4	HKEY_CURRENT_CONFIG

CEProtectMode

CEProtectMode is a bitmask for protecting a file or a process.

The following table lists all supported values for this data type.

Table 92: CEProtectMode values

Value	ID	Description
CE_PROTECT_READ	0x0	Protect the file from read
CE_PROTECT_WRITE	0x1	Protect the file from write
CE_PROTECT_DELETE	0x2	Protect the file from being deleted
CE_PROTECT_MOVE	0x4	Protect the file from being moved
CE_PROTECT_RENAME	0x8	Protect the file from being renamed
CE_PROTECT_METADATA_CHANGE	0x10	Protect the file property or security from being changed
CE_PROTECT_KILL	0x20	Protecting the process from being killed

CEHandle

Handle assigned to various objects for the linking and ID purposes.

Ceint32

Data structure used to represent integers, such as IP addresses, timeout values, or counts of entities, such as users or obligations.

CEBoolean

CEBoolean is a binary value, used for marking either/or cases, such as whether an obligation should be imposed in a given policy evaluation.

Table 93: CEBoolean values

Value	ID	Description
CEFalse	0	False
CETrue	1	True

CEAction_t

CE_Actions are a data type that represents any action a user might perform, which will be checked for policy coverage through one of the CEEVALUATE calls.

The following table lists all supported values for this data type.

Table 4.10-39: CEAction values

Value	ID	Description
CE_ACTION_READ	1	Reading a resource (file)
CE_ACTION_DELETE	2	Deleting a resource (file)
CE_ACTION_MOVE	3	Moving a resource (file) from one location to different location
CE_ACTION_COPY	4	Copy a resource (file) to another resource (file)
CE_ACTION_WRITE	5	Write to a resource (file)
CE_ACTION_RENAME	6	Renaming a resource (file)

Value	ID	Description
CE_ACTION_CHANGE_ATTR_FILE	7	Changing the attributes on a file
CE_ACTION_CHANGE_SEC_FILE	8	Changing the security settings on a file
CE_ACTION_PRINT_FILE	9	Printing a file
CE_ACTION_PASTE_FILE	10	Pasting a file (or content of file) to the client PEP application
CE_ACTION_EMAIL_FILE	11	Emailing out a file
CE_ACTION_IM_FILE	12	IM out a file
CE_ACTION_EXPORT	13	Export a resource
CE_ACTION_IMPORT	14	Import a resource
CE_ACTION_CHECKIN	15	Check in a resource
CE_ACTION_CHECKOUT	16	Check out a resource
CE_ACTION_ATTACH	17	Attaching a resource
CE_ACTION_RUN	18	Running a process

CEEEnforcement_t

CEEenforcements are all possible enforcement decision results that can be returned from the PDP to the PEP.

Syntax:

```
struct _CEEEnforcement_t {
    CEResponse_t result;
    CEAttributes * obligation
```

Table 94: CEResponse values

Value	ID	Description
CEDeny	0	Action denied
CEAllow	1	Action allowed
CEDontCare	2	No decision

Obligation can be null from the return CEEenforcement, in which case no obligation needs to be performed on the client.

CENoiseLevel_t

CENoiseLevel specifies the action level to use for a policy evaluation.

The following table lists the values.

Table 95: CENoiseLevel values

Value	ID	Description
CE_NOISE_LEVEL_MIN	0	Use the minimum action level
CE_NOISE_LEVEL_SYSTEM	1	All actions are allowed with no obligations
CE_NOISE_LEVEL_APPLICATION	2	Evaluate the policy, but not the obligations

Value	ID	Description
CE_NOISE_LEVEL_USER_ACTION	3	Evaluate the policy and run the obligations that apply
CE_NOISE_LEVEL_MAX	4	Use the maximum action level

CEPEP_t

CEPEP is a data type that expresses which of the valid types of PEPs is involved in a given case.

The following table describes CEPEP types.

Table 96: CEPEP types

Value	ID	Description
CE_PEP_NONE	0x0	No PEP
CE_PEP_KERNEL	0x1	CE kernel level PEP
CE_PEP_FS	0x2	CE kernel level file system PEP
CE_PEP_NETWORK	0x4	CE kernel level network PEP
CE_PEP_API	0x8	CE user level library PEP (for example, win32)
CE_PEP_APPLICATION	0x10	CE user level application PEP

CERequest

CERequest contains all the information for a policyevaluation query that is sent in a multiquery request to the Policy Controller. A multi-query request is sent using CEEVALUATE_CheckResourcesEx().

```
typedef struct _CERequest {
    CCString operation;
    CEResource *source;
    CEAttributes *sourceAttributes;
    CEResource *target;
    CEAttributes *targetAttributes;
    CEUser *user;
    CEAttributes *userAttributes;
    CEApplication *app;
    CEAttributes *appAttributes;
    CCString *recipients;
    CEint32 numRecipients;
    CENamedAttributes *additionalAttributes;
    CEint32 numAdditionalAttributes;
    CEBoolean performObligation;
    CENoiseLevel_t noiseLevel;
} CERequest;
```

Chapter

15

Appendix

Topics:

- FAQs and troubleshooting
- Password and users
- SSL certificates

FAQs and troubleshooting

General FAQs

Question	Answer
The system automatically logged me out when I was working on a policy. Why did this happen, and will I lose my work?	For security, sessions are automatically logged out after 30 minutes of inactivity. The session timeout period cannot be changed. When you are logged in to Control Center and working on policies, your work is automatically saved as you go; you do not need to explicitly save your work. If you are interrupted while working on a policy, or you want to work on another task and return to authoring the policy later, you can stop and continue the authoring process as desired; your work is saved as a draft.

Troubleshooting FAQs

Issue	Solution
Out-of-memory errors occur during data intensive operations	Change memory settings. See Changing memory settings on page 349.
The Status tab indicates a problem with a control center component.	Restart the Policy Server service. See Restarting the Control Center Policy Server on page 350.

Changing memory settings

Out-of-memory errors might occur during data-intensive operations, such as enrolling large amounts of data, generating a report with more than 50,000 records, or synchronizing a large number of activity logs.

If this happens, change the JVM memory settings in the Windows Registry.

1. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type **regedit**, and press **Enter**.
2. In the Registry Editor, navigate to the following folder:

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Apache Software Foundation\Procrun 2.0\CompliantEnterpriseServer\Parameters\Java

3. Open Options and look for the following settings. Increase the values as needed.

-Xms256M

-Xmx1024M

-XX:MaxPermSize=256M

Depending on the load on Control Center, try one of the following settings:

-Xmx2048M

-Xmx3072M

-Xmx4096M

Restarting the Control Center Policy Server

Whether running on a single host, on distributed hosts, or in a failover cluster, the Control Center Policy Server service must be manually started after the initial installation.

By default, it is not set to restart automatically if it fails. To enable the Policy Server service to restart automatically, change the Startup Type from **Manual** to **Automatic**.

You can use the **Status** tab to monitor whether the individual components are running normally. If one fails and you need to restart it, you can do so by restarting the Policy Server service, regardless of whether that service represents more than one Control Center component.

Similarly, if you ever need to stop any server process, you must stop the service even though this stops all Control Center components running on that host.

For Windows:

1. Go to **Control Panel > Administrative Tools > Services**.
2. Find Control Center Policy Server in the list of services.
3. Right-click the service name, then select **Start**.
4. To stop Control Center, right-click the service name, then select **Stop**.

For RHEL:

1. To stop Control Center, use `stop-policy-server.sh` located in `<installation folder>/PolicyServer`.
2. Type the one of the following commands:
 - `sudo ./stop-policy-server.sh`
 - `sudo service CompliantEnterpriseServer stop`
3. To check if the Tomcat process has started, type the following command:
`ps -ef | grep tomcat`
4. To start Control Center, use `start-policy-server.sh` located in `<installation folder>/PolicyServer`.
5. Type one of the following commands as root user:
 - `sudo ./start-policy-server.sh`
 - `sudo service CompliantEnterpriseServer start`

Related tasks

[Recovering your username or password](#) on page 30

If you have forgotten your username or password, follow these steps to have it emailed to you or reset.

Disabling the resource type checking in policy evaluations

By default, Control Center takes resources type into account while evaluating policies.

Previously, if a policy had an expression like "`resource.fso.license != "ABC"`" and you queried with "license" set to "XYZ", but a resource type of "document", the expression would evaluate to true.

You can disable the resource type checking in policy evaluations by changing the default behavior for the Policy Controller, by setting the `nextlabs.evaluation.uses.resource.type` property to false.

1. Stop the Java Policy Controller or Policy Controller.
2. Add the `Dnextlabs.evaluation.uses.resource.type` java property.
 - For Java Policy Controller, follow these following steps:
 - a. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type regedit, and press **Enter**.
 - b. In the Registry Editor, navigate to the following folder:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Apache Software Foundation  
\Procrun 2.0\Tomcat8\Parameters\Java
```

- c. Insert the following command:

```
-Dnextlabs.evaluation.uses.resource.type= false
```

- For Policy Controller, follow these steps:

- a. Open the Registry Editor. You can do this by clicking the Windows Start button, then in the search box, type regedit, and press Enter.

- b. In the Registry Editor, navigate to the following folder:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services  
\ComplianceEnforcerService\Parameters
```

- c. Insert the following command:

```
-Dnextlabs.evaluation.uses.resource.type= false
```

- For Linux Java Policy Controller, add the following CATALINA_OPTS environment variable:

```
-Dnextlabs.evaluation.uses.resource.type=false
```

- 3. Restart the Java Policy Controller or Policy Controller.

Securing the JDBC connection between Control Center and Oracle Database

If the certificate is issued by a trusted CA, you do not need to follow these steps.

After installing Oracle 12c, generate a wallet containing a self-signed certificate and export the public certificate to the Java TrustStore used by Control Center. If the certificate is not issued by a trusted CA, follow these steps to import the certificate to Java TrustStore used by Control Center.

1. Copy the above exported database server certificate (for example, server-certificate.cer) to the following folder:

```
<CC_installation_folder>\PolicyServer\java\jre\lib\security
```

2. Open a command prompt and navigate to the following folder:

```
<CC_installation_folder>\PolicyServer\java\jre\lib\security
```

3. Run the following command to import the certificate.

```
<CC_installation_folder>\PolicyServer\java\bin\keytool.exe -import -alias  
<certalias> -file server-certificate.crt -keystore cacerts -storepass  
<changeit>
```

 **Note:** The alias name must be unique. To ensure this, one option is to use the database's hostname as the <certalias>.

4. Find and replace the JDBC connection string in configuration.xml (5 instances), cc-console-app.properties, and cas.properties. Your new JDBC connection string should be something similar to the following:

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS) (HOST=<hostname>)  
(PORT=1521)) (CONNECT_DATA=(SID=<sid>)))
```

 **Note:** Use the same values for <hostname> and <sid> that you specified during the installation.

Extending datatypes in Oracle 12c

In Oracle 12c, you can increase the maximum sizes of VARCHAR2, NVARCHAR2 and RAW data types to a maximum of 32767 bytes (not characters).

 **Note:** Switching to the extended data types is a one-way operation and once switched, it is not possible to switch back without some form of database recovery.

1. Check the existing value of MAX_STRING_SIZE parameter. By default this will have the value "STANDARD".
SHOW PARAMETER MAX_STRING_SIZE;

2. Shutdown the database.
SHUTDOWN IMMEDIATE;

3. Restart the database in UPGRADE mode.
STARTUP UPGRADE;

4. Change the value of MAX_STRING_SIZE parameter to EXTENDED.

```
ALTER SYSTEM SET MAX_STRING_SIZE=EXTENDED SCOPE=SPFILE;
```

5. Shutdown the database.
SHUTDOWN IMMEDIATE;

6. Restart the database in UPGRADE mode.
STARTUP UPGRADE;

7. Run the `rdbms/admin/ut132k.sql` script.

```
@?/rdbms/admin/ut132k.sql
```

 **Note:** You must be connected AS SYSDBA to run the script.

8. Shutdown the database.
SHUTDOWN IMMEDIATE;

9. Restart the database in NORMAL mode.
Command: STARTUP;

10. Verify whether the MAX_STRING_SIZE parameter value has changed correctly.
SHOW PARAMETER MAX_STRING_SIZE;

11. Test by creating a table.

```
CREATE TABLE TEST_TABLE (
  DATA VARCHAR2(32767)
);
DROP TABLE TEST_TABLE;
```

Password and users

Control Center uses several different passwords for various restricted activities, and these passwords are maintained in different ways.

This appendix summarizes how passwords work, how the default values are set, and how you can change them.

Related concepts

[Enrollment utility password requirements](#) on page 97

Some enrollment utilities require users to enter the password of the super user Administrator account, which is initially configured during Control Center installation. This password can be changed as needed.

Enforcer Profile Security Passwords

Each policy enforcer profile has a security password defined for it to protect the enforcer from tampering.

This password is not associated with any user. Rather, it enables anyone to shut down and uninstall any enforcer using that profile. The password is required by the `StopDesktopEnforcer.exe` and `StopFileServerEnforcer.exe` tools, and also during an uninstallation using the standard Windows Add/Remove Programs tool.

Initial Value

All enforcers are assigned a default profile, depending on their type.

You can create different, custom profiles and assign them to enforcers at any time, instead of the default profiles. For both the initial default profiles and any that you create, a default password string password applies, until you provide a different password string. This means that this default password applies to an enforcer unless you have:

- Changed this default password of the default profiles, or
- Defined a new profile with a different password and assigned the new profile to the enforcer.

Because this default password can represent a security gap, NextLabs strongly recommends that you change it for both these default profiles as soon as Control Center is installed.

How to Change

You can change the password for any enforcer profile, including the two default profiles.

You do this by opening the Administrator console, going to the Policy Enforcer Configuration tab, selecting the profile in the list on the left, and typing the new password in the Administrative Password and Confirm Password fields.

Managing Passwords

You should change the passwords of the default profiles as soon as you install Control Center, since leaving the default could represent a security hole.

Like application user passwords, enforcement passwords are never visible anywhere, and cannot be looked up by anyone. This means you must keep track of passwords somewhere outside of Control Center, or change them if they are lost.

This is especially important for enforcer profiles, since there may be many profiles, each with a different password. If an administrator needs to shut down a desktop enforcer, for example, he needs to find out which profile is assigned to that enforcer, but he then needs to be able to find out the password for that profile.

Utility Security Password

The utility security password enables anyone to use several Control Center utilities.

The password must be provided as the `-w` argument in the command line used to launch each utility.

Related concepts

[About Property Manager](#) on page 100

The Property Manager utility (`propertymgr.bat`) enables you to define custom properties for User and Computer components, and for user and host groups.

Initial Value

By default, the utility security password is the same as the password string defined for application superuser during initial installation.

How to Change

Anyone logged into the Reporter console or the Administrator console as this superuser can manually change this password by clicking on the Change Password link at the upper right of the main screen.

After you change this password, it applies to all utilities that require it; you cannot define different passwords for separate utilities.

Database Password

A username and password is required whenever Control Center connects to the database it is using to store Activity Journal data.

This connection is performed transparently during normal Control Center operation. If the location of the Activity Journal changes, however, this connection must be reconfigured.

These settings are stored in the ConnectionPools section of the configuration XML file. This section has three ConnectionPool sections, one each for the Policy Master, Activity Journal, and Management DB—in that order.

Initial Value

This connect string and password are set in response to prompts from the install wizard, during the initial installation of Control Center.

How to Change

If you need to change the database connection at any time after initial installation of Control Center, edit the <UserName>, <Password>, and <ConnectionString> elements in this file, for the activity.connection.pool (the second one in the section).

As with other passwords, you can use the crypt.jar utility to generate a new encrypted password.



```
<ConnectionPools>
  <ConnectionPool>
    <Name>policyframework.connection.pool</Name>
    <Username>root</Username>
    <Password>66437346074625490d7810755776</Password>
    <ConnectionString>jdbc:postgresql://GRANDE9:5432/pf</ConnectionString>
    <DriverClassName>org.postgresql.Driver</DriverClassName>
    <MaxPoolSize>30</MaxPoolSize>
  /ConnectionPool>

  <ConnectionPool>
    <Name>activity.connection.pool</Name>
    <Username>qa3</Username>
    <Password>4c4f414c3d4c2e4f7841</Password>
    <ConnectionString>jdbc:oracle:thin:@HostName:1521:orcl</ConnectionString>
    <DriverClassName>oracle.jdbc.driver.OracleDriver</DriverClassName>
    <MaxPoolSize>30</MaxPoolSize>
  /ConnectionPool>

  <ConnectionPool>
    <Name>management.connection.pool</Name>
    <Username>root</Username>
    <Password>5866627a695054665162187a05690b7e00656b4a</Password>
    <ConnectionString>jdbc:postgresql://GRANDE5:5432/management</ConnectionString>
    <DriverClassName>org.postgresql.Driver</DriverClassName>
    <MaxPoolSize>30</MaxPoolSize>
  /ConnectionPool>
</ConnectionPools>
```

Figure 106: Database connection pool settings

SSL certificates

All internal communication among various software components of the Control Center platform is conducted using SSL encryption.

Two types of security certificates are used to ensure security throughout Control Center:

- Component-to-component certificates provide certification between Control Center software components, such as between policy enforcers and ICENet Servers or between distributed Control Center components.
- Web application certificates support HTTPS for the Reporter console and Administrator console.

The certificates are stored in a keystore (.jks) file, whose location is specified in the configuration file `server.xml`. SSL certificates are issued during Control Center installation, are self-signed by NextLabs, Inc., and expire in ten years.

You can use Control Center without making any changes to its security system. It is recommended, however, that you change the default password to the keystores for Enrollment Manager, Property Manager, and Key Management. This password is the word `password`, which is not secure.

In addition, some organizations prefer to use their own SSL certificates instead of the NextLabs certificates. You can replace any of the SSL certificates as needed.

Related concepts

[Configuration file structure](#) on page 146

Overview of communication process

On a default installation, certificates (keystores and truststores) are already distributed at all sender and receiver ends to authenticate communication between Control Center components.

The following figure illustrates where the certificates are on a default installation.

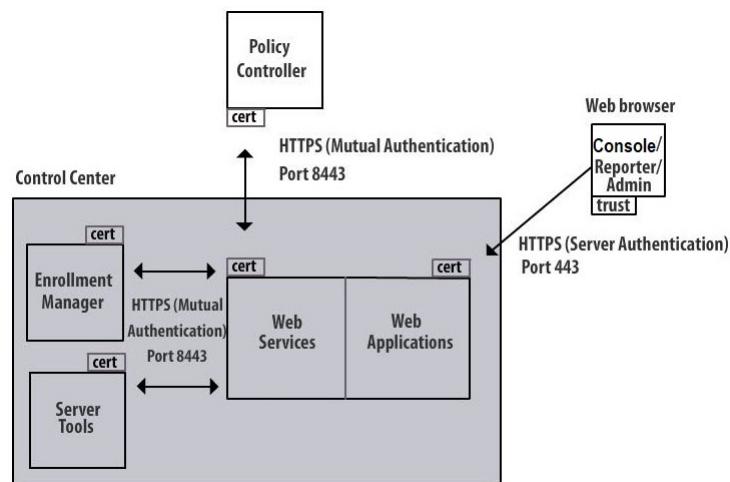


Figure 107: SSL authentication across Control Center components

About keystore and truststore files

The keystore and truststore files are used together to authenticate server-to-client communication.

The `keystore.jks` file is used to store private and public keys used for encryption, and the `truststore.jks` file contains the Key Authentication Chain, which verifies that the key has not been changed without authorization.

Overview of security certification

Authentication occurs among Control Center components, as well as, between Control Center and the applications it communicates with.

The following table lists the components and their associated keystore and truststore files, and the required configuration procedures after the replacement of certificates.

Table 97: Overview of security certifications

Component	Default Keystore Name	Stored in Truststore(s)	Other Requirements
Web Components	web-keystore.jks	<newweb-truststore.jks>	Must edit <code>server.xml</code> with the paths to the new keystore and truststore

Component	Default Keystore Name	Stored in Truststore(s)	Other Requirements
Internal Server Components (DCC)	dcc-keystore.jks	<newdcc-truststore.jks>	Must edit server.xml with the paths to the new keystore and truststore
Policy Controller, First Connection	temp_agent-keystore.jks	<newdcc-truststore.jks>	Does not need to be signed by an external certificate authority (CA) as it is not used operationally; a copy of the cert is supplied by NextLabs support
Policy Controller, Regular Connection	agent-keystore.jks	<newdcc-truststore.jks> and Agent-truststore.jks	The new file must be named Agent-keystore.jks All existing Agent.jks files (not Temp_Agent files) should be deleted on the Policy Controller host, at: <install dir>\Policy Controller\config\security
			The Policy Controller must be re-registered with the Policy Server
Policy Server Tools: Import/ Export and Location Importer	policyauthor-keystore.jks	<newdcc-truststore.jks>	The new file must be named PolicyAuthor-keystore.jks
Enrollment	enrollment-keystore.jks	<newdcc-truststore.jks>	The new file must be named enrollment-keystore.jks The existing copy of the keystore must be overwritten at <installdir>\PolicyServer\tools\enrollment\security
Web Components	web-keystore.jks	<newweb-truststore.jks>	Must edit server.xml with the paths to the new keystore and truststore

Using Java keytool to manage certificates

Java keytool is a key and certificate management utility, which you can use to manage or replace any of the NextLabs certificates.

All key management procedures in this section use keytool. In all commands presented in this section, values that are user-defined appear in angle brackets (<>). Values that are not in angle brackets are not user-defined and must be entered exactly. For more information about keytool commands, see the Java keytool documentation.

Replacing Control Center certificates

All Control Center components require certificates to authenticate communication between the components.

You can replace Control Center certificates in the following ways:

- [Generate certificates using a new KeyStore and certificate](#)
- [Generate certificates using existing keypair from Control Center](#)
- [Generate certificates using an existing keypair from your company](#)

You can replace any of these certificates. Each certificate is associated with a unique alias. When running a `keytool` command that requires the `-alias` parameter, specify the correct alias associated with the certificate you are generating or importing. The following table lists the Control Center components and their certificate aliases.

Table 98: Control Center components and certificate aliases

Component	Certificate Alias
Web Applications (Administrator console and Reporter console)	Web
Web services components (internal server components)	dcc
Policy Controller, first connection	Temp_Agent
Policy Controller, regular connections	Agent
Enrollment Manager, Property Manager	Enrollment

When replacing certificates, the procedures must be performed on the Control Center host where the Policy Server is installed.

If you replace the certificate for the Policy Controller component in Control Center, you must also update the certificate on all devices where Policy Controller is running.

Generating a new certificate

Follow these steps to replace NextLabs certificates with new certificates.

1. Generate a new KeyStore.
2. Generate a new certificate from the KeyStore and have signed by an external authority.
3. Import the certificates into the pertinent TrustStore files.
4. Overwrite old TrustStore and KeyStore files in all relevant locations.

Web component certificates

Web components include the Reporter console and the Administrator console browser-based web applications, which, by default, use port 443 to communicate with the server.

These components store the default certificates in two files:

- `web-keystore.jks`
- `web-truststore.jks`

Generating certificates using a new KeyStore and certificate

Follow these steps to configure new web component certificates using new KeyStore and certificate.

1. Generate the KeyStore. In the Java Keytool example, this is done using the following command.

```
keytool -genkeypair
-dname "cn=<commonName>, OU=<organizationUnit>, O=<localityName>,
S=<stateName>, C=<CountryName>""
-alias Web -keypass <mypasswd>
-storepass <mypasswd>
-keyalg RSA
```

```
-keystore <myweb.jks>
-ext "SAN=DNS:<commonName>"
-validity <validDays>
```

2. Generate a certificate request that can be submitted to a third-party certificate authority (CA). For example,

```
keytool -certreq
-alias Web
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb.jks>
-file <myweb.csr>
```

3. Submit the certificate request to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:

- a. Submit an advanced certificate request for a User certificate that enables you to supply a base-64-encoded CMC or PKCS #10 file.
- b. Copy and paste the certificate request from the certificate request file, `myweb.csr`, generated in step 2. To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST.

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIICgzCCAKECAQAwfzELMAkGA1UEBhMCVVMxGzAJBgNVBAgTAKNBMRiwEAYDVQQHEw1TY
W4gtTWF0ZW8xEzARBgNVBAoTCkJsdWVKdW5nbGUxHDAaBgNVBAsTE0NvbXBsaWFudEVudGV
yCHJpc2UxHDAa
...
7ZpvSDn94tEBPayBaAAMAsGByqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9D1RzcQi0
7E0QIUaM9W8V6UuYbIzBrR5v9XTu9cv94=
-----END NEW CERTIFICATE REQUEST-----
```

- c. Select a certificate template where the subject line matches the subject in the `.csr` file. The subject line should not be changed.
- d. For latest browsers to accept the certificate correctly, add the extension Subject Alternative Name (SAN) with a DNS attribute that refers to the FQDN of the server. It should be the same as the Common Name (CN).
- e. When prompted, download the issued certificate. The reply from CA should be in a format that contains both the signed certificate and the CA certificate.

The certificate is named `certnew.cer`. You can change the certificate name (in this example, `myweb.cer`).

 **Important:** If the downloaded certificate file contains only a signed certificate and not a CA certificate, you must manually concatenate the signed certificate and the CA certificate.

4. Copy the new certificate into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

5. Create a web truststore and import the new signed certificate into it. For example,

```
keytool -import
-alias Web
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb-truststore.jks>
-file <myweb.cer>
```

 **Note:** When you specify the `-keystore <myweb-trust.jks>` option in the command above, you are actually generating a new TrustStore file.

6. Import the CA certificate into the TrustStore.

```
keytool -import -alias ca  
-keypass <mypasswd>  
-storepass <mypasswd>  
-keystore <myweb-truststore.jks>  
-file <myca.cer>
```

7. Copy the new TrustStore into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

8. Import the CA reply file that you downloaded in step 3e into the keystore that you created in step 1. When prompted to install reply, type yes.

```
keytool -import -alias web  
-keystore <myweb.jks>  
-trustcacerts  
-file <ca-reply-file>  
-keypass <mypasswd>  
-storepass <mypasswd>
```

 **Note:** Ensure that the `is web` and `must` be applied to the keystore and not the truststore. If you get the following error, it means that your `<ca-reply-file>` is not complete.

```
keytool error: java.lang.Exception: Failed to establish chain from  
reply
```

In such a situation, you must manually concatenate the `<ca-reply-file>` with your CA cert and rerun the step 8.

9. View the keystore entry using the following command:

```
keytool -list -alias web  
-keystore <myweb.jks>  
-storepass <mypasswd>  
-v
```

The process returns something similar to the following:

```
Keystore type: JKS  
Keystore provider: SUN  
  
Your keystore contains 1 entry  
  
Alias name: web  
Creation date: Aug 7, 2018  
Entry type: PrivateKeyEntry  
Certificate chain length: 2  
Certificate[1]:  
Owner: CN=democc02w12r2.qapf1.qalab01.nextlabs.com,  
OU=CompliantEnterprise, O=NextLabs, ST=California, C=US  
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,  
ST=Singapore, C=SG  
Serial number: 49237803  
Valid from: Tue Aug 07 12:34:57 SGT 2018 until: Mon Nov 05 12:34:57 SGT  
2018  
Certificate fingerprints:  
MD5: E1:F8:BF:20:98:3C:26:A5:FC:93:85:82:DF:28:A4:9E  
SHA1: EB:2E:39:3C:69:B1:A1:84:5E:E8:6C:80:3B:F7:DB:F4:4C:12:FF:00  
SHA256:  
19:D2:D3:B7:67:C1:12:2B:A8:C2:89:36:66:4C:7F:3C:1D:64:37:8B:84:B2:7E:B1:  
DF:85:6D:A6:CD:45:12:CB  
Signature algorithm name: SHA256withRSA
```

```

Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 33 01 06 62 6B 2E 8E 33   99 C6 70 28 A6 F1 45 93  3..bk..3..p(..E.
0010: 6B A1 06 BD                                         k...
]
]

#2: ObjectId: 2.5.29.17 Criticality=false
SubjectAlternativeName [
  IPAddress: 10.23.58.76
  DNSName: democc02w12r2.qapf1.qalab01.nextlabs.com
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: FB EE C5 10 BE 6F 1E 38   2E 73 14 7A 6A B5 00 C7  ....o.8.s.zj...
0010: E7 63 59 E8                                         .cy.
]
]

Certificate[2]:
Owner: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Serial number: 5b613509
Valid from: Wed Aug 01 12:20:25 SGT 2018 until: Thu Aug 01 12:20:25 SGT
2019
Certificate fingerprints:
      MD5: E9:49:45:86:9A:4E:92:41:E2:E2:AE:10:05:63:D6:3A
      SHA1: BD:58:51:8F:07:F0:9C:9E:AD:F0:90:D9:28:5F:B0:BB:5A:89:7A:B4
      SHA256:
52:04:C4:A6:0F:8A:8F:18:21:AD:12:34:41:06:6B:D7:98:4F:03:27:A8:
68:8A:F6:95:AD:C7:B9:DA:9A:0D:00
Signature algorithm name: SHA256withRSA
Version: 3

```

10. Copy the keystore into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

11. Using the `mkpassword.bat` utility, supplied at `<installdir>\Policy- Server\tools\crypt`, encrypt the passwords for the keystore and truststore files (`myweb.jks` step 1 and `myweb-trust.jks` step 5). This command must be run for each password, where `<mypasswd>` refers to the password you designated in steps 1 and 5 above:

```
mkpassword.bat -w <mypasswd>
```

The process returns something similar to the following:

```
"7542615506752a4200613b55295d16771d730b7e"
```

 **Note:** This step is necessary because the Control Center uses the passwords to open the files and use their contents, and the passwords should not be accessible in plain text.

12. Modify the configuration file for the Tomcat web application server, `server.xml`, to point to the new keystore and truststore files, and to include the new passwords. By default, this file is placed on the Management Server host, at:

```
<install dir>\PolicyServer\server\configuration
```

Open this file, locate the <CE-Apps> section, and supply values for the attributes shown in the following figure. Be sure to place all values inside double quotes:

- keystoreFile: Name of the new keystore file <myweb.jks>
- keystorePass: Password of the new keystore file, after encrypting with the makepassword.bat utility
- truststoreFile: Name of the new truststore file <myweb-trust.jks>
- truststorePass: Password of the new truststore file, after encrypting with the makepassword.bat utility
- keystoreType: Do not change this value from JKS
- truststoreType: Do not change this value from JKS

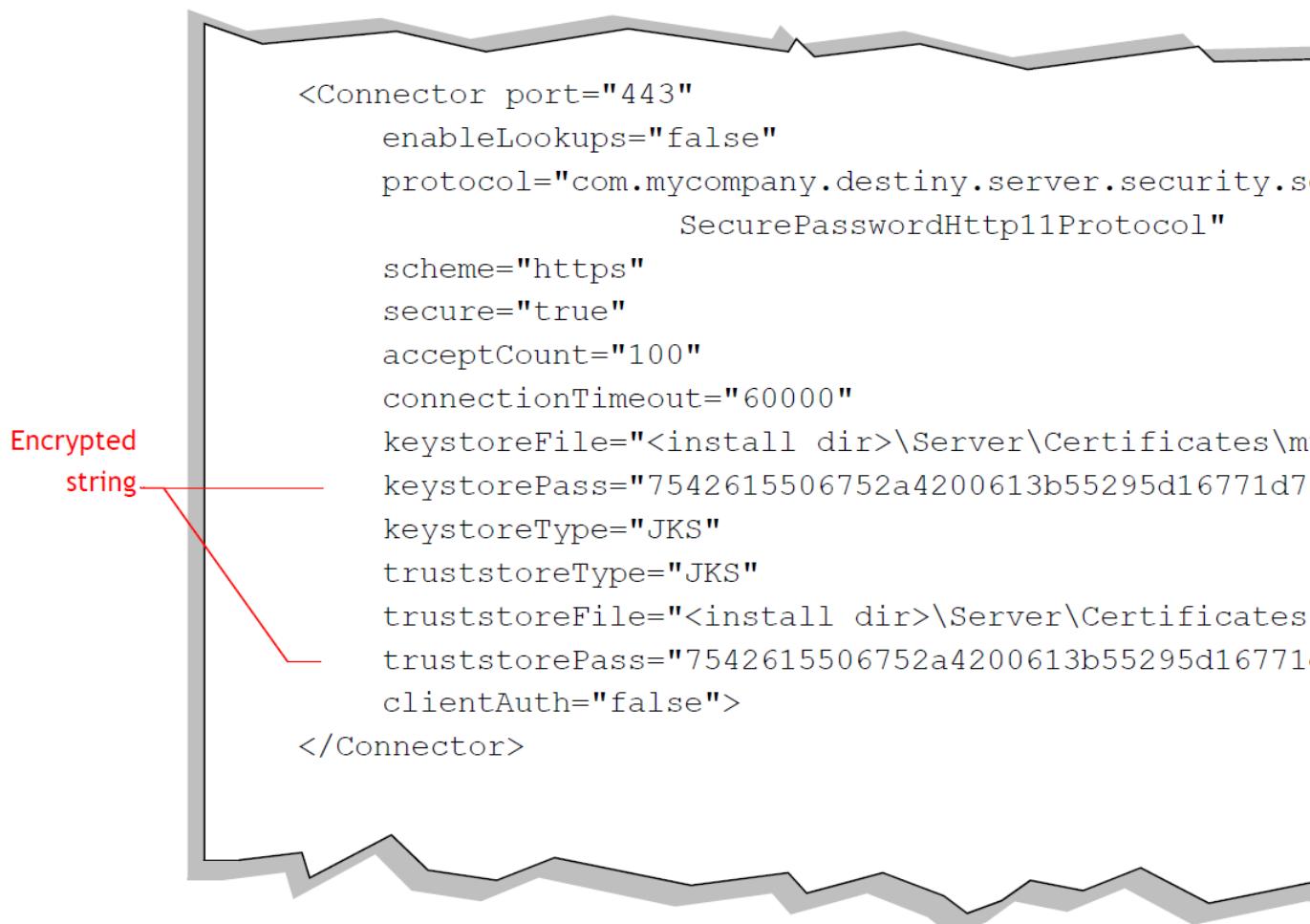


Figure 108: Configuring the web connection port (443)

Generating certificates using existing keypair from Control Center

1. Generate a certificate request that can be submitted to a third-party certificate authority (CA) from an existing keystore which is web-keystore.jks.

 **Note:** The password for keystore is entered during installation.

```
keytool -certreq -alias web
-keypass <mypasswd>
-storepass <mypasswd>
-keystore web-keystore.jks
```

```
-file <myweb.csr>
```

2. Submit the certificate request (the .csr file generated in Step 1) to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:

- a. Submit an Advanced request for a User certificate that enables you to supply a base-64- encoded CMC or PKCS #10 file.
- b. Copy and paste the certificate request from the newly generated certificate request file (myweb.csr above). To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST.

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIICgzCCAKECAQAwfzELMAKGA1UEBhMCVVMxCzAJBgNVBAgTAkNBMRIwEAYDVQQHEwl
TYW4gTWF0ZW8xEzARBgNVBAoTCkJsdWVKdW5nbGUxDAAgNVBAsTE0NvbXBsaWFudE
VudGVycHJpc2UxHDAa
...
7ZpvSDn94tEBPgayBaAAMAsGBYqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9D1Rzc
Qi07E0QIUaM9W8V6UuYbIzBrR5v9XTu9cV94=
-----END NEW CERTIFICATE REQUEST-----
```

- c. Select a certificate template where the subject line matches the subject in the .csr file. The subject line should not be changed.
- d. For latest browsers to accept the certificate correctly, add the extension Subject Alternative Name (SAN) with a DNS attribute that refers to the FQDN of the server. It should be the same as the Common Name (CN).
- e. When prompted, download the issued certificate. The reply from CA should be in a format that contains both the signed certificate and the CA certificate.

The certificate is named certnew.cer. You can change the certificate name (in our example, myweb.cer).

 **Important:** If the downloaded certificate file contains only a signed certificate and not a CA certificate, you must manually concatenate the signed certificate and the CA certificate.

3. Import the new signed certificate into web-truststore.jks. For example,

```
keytool -import -alias web-sign
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb-truststore.jks>
-file <myweb.cer>
```

4. Import the CA certificate into the truststore. For example,

```
keytool -import -alias ca
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb-truststore.jks>
-file <myca.cer>
```

5. Copy the new truststore into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

6. Import the CA reply file that you downloaded in step 2e into web-keystore.jks. When prompted to install reply, type yes.

```
keytool -import -alias web
-keystore web-keystore.jks
-trustcacerts
-file <ca-reply-file>
-keypass <mypasswd>
```

```
-storepass <mypasswd>
```

 **Note:** Ensure that the `is web` and `must be applied to the keystore` and not the trustore. If you get the following error, it means that your `<ca-reply-file>` is not complete.

keytool error: java.lang.Exception: Failed to establish chain from reply

In such a situation, you must manually concatenate the `<ca-reply-file>` with your CA cert and rerun the step 8.

7. View the keystore entry using the following command:

```
keytool -list -alias web  
-keystore <myweb.jks>  
-storepass <mypasswd>  
-v
```

The process returns something similar to the following:

```
Keystore type: JKS  
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: web
Creation date: Aug 7, 2018
Entry type: PrivateKeyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=democc02w12r2.qapf1.qalab01.nextlabs.com,
OU=CompliantEnterprise, O=NextLabs, ST=California, C=US
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Serial number: 49237803
Valid from: Tue Aug 07 12:34:57 SGT 2018 until: Mon Nov 05 12:34:57 SGT
2018
Certificate fingerprints:
    MD5: E1:F8:BF:20:98:3C:26:A5:FC:93:85:82:DF:28:A4:9E
    SHA1: EB:2E:39:3C:69:B1:A1:84:5E:E8:6C:80:3B:F7:DB:F4:4C:12:FF:00
    SHA256:
        19:D2:D3:B7:67:C1:12:2B:A8:C2:89:36:66:4C:7F:3C:1D:64:37:8B:
        84:B2:7E:B1:DF:85:6D:A6:CD:45:12:CB
        Signature algorithm name: SHA256withRSA
        Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 33 01 06 62 6B 2E 8E 33      99 C6 70 28 A6 F1 45 93  3..bk..3..p(..E.
0010: 6B A1 06 BD                           k...
]
]

#2: ObjectId: 2.5.29.17 Criticality=false
SubjectAlternativeName [
    IPAddress: 10.23.58.76
    DNSName: democc02w12r2.qapf1.qalab01.nextlabs.com
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
```

```

KeyIdentifier [
 0000: FB EE C5 10 BE 6F 1E 38    2E 73 14 7A 6A B5 00 C7  ....o.8.s.zj...
 0010: E7 63 59 E8                .cY.
]
]

Certificate[2]:
Owner: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Serial number: 5b613509
Valid from: Wed Aug 01 12:20:25 SGT 2018 until: Thu Aug 01 12:20:25 SGT
2019
Certificate fingerprints:
MD5:  E9:49:45:86:9A:4E:92:41:E2:E2:AE:10:05:63:D6:3A
SHA1: BD:58:51:8F:07:F0:9C:9E:AD:F0:90:D9:28:5F:B0:BB:5A:89:7A:B4
SHA256:
52:04:C4:A6:0F:8A:8F:18:21:AD:12:34:41:06:6B:D7:98:4F:03:27:A8:68:
8A:F6:95:AD:C7:B9:DA:9A:0D:00
Signature algorithm name: SHA256withRSA
Version: 3

```

8. Copy the keystore into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

Generating certificates using an existing keypair from your company

Use this method if you are using a different tool to generate the keypair before importing into Control Center.

Control Center only supports keystore of type JKS. Therefore, you must convert the keypair into the .jks format. The keytool command only supports importing the keypair from .p12 format, so assumption here is that the provided keypair is in the .p12 format.

1. Import the .p12 file. For example,

```

keytool -importkeystore
-srckeystore <p12 format keypair>
-destkeystore <myweb.jks>
-srcstorepass <p12 file password>
-deststorepass <destination store password>
-destkeypass <destination key password>
-srcalias <keypair alias in p12>
-destalias web
-v

```

2. Generate a certificate request that can be submitted to a third-party certificate authority (CA). For example,

```

keytool -certreq
-alias Web
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb.jks>
-file <myweb.csr>

```

3. Submit the certificate request to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:

- Submit an Advanced request for a User certificate that enables you to supply a base-64- encoded CMC or PKCS #10 file.

- b. Copy and paste the certificate request from the newly generated certificate request file (`myweb.csr` above). To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST.

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIICgzCCAKECAQAwfzELMAkGA1UEBhMCVVMxGzAJBgNVBAgTAkNBMRIwEAYDVQQHEw1TYW
4gTWF0ZW8xEzARBgNVBAoTCkJsdWVKdW5nbGUxHDAaBgNVBAsTE0NvbXBsaWFudEVudGVy
cHJpc2UxHDAa
...
7ZpvSDn94tEBPgayBaAAMAsGByqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9DlRzcQi0
7E0QIUaM9W8V6UuYbIzBrR5v9XTu9cv94=
-----END NEW CERTIFICATE REQUEST-----
```

- c. Select a certificate template where the subject line matches the subject in the `.csr` file. The subject line should not be changed.
d. For latest browsers to accept the certificate correctly, add the extension Subject Alternative Name (SAN) with a DNS attribute that refers to the FQDN of the server. It should be the same as the Common Name (CN).
e. When prompted, download the issued certificate. The reply from CA should be in a format that contains both the signed certificate and the CA certificate.

The certificate is named `certnew.cer`. You can change the certificate name (in our example, `myweb.cer`).

 **Important:** If the downloaded certificate file contains only a signed certificate and not a CA certificate, you must manually concatenate the signed certificate and the CA certificate.

4. Copy the new certificate into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

5. Create a web truststore and import the new signed certificate into it. For example,

```
keytool -import
-alias Web
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb-truststore.jks>
-file <myweb.cer>
```

 **Note:** When you supply the `-keystore <myweb-trust.jks>` option in the command above, you are actually generating the new truststore file.

6. Import the CA certificate into the truststore.

```
keytool -import -alias ca
-keypass <mypasswd>
-storepass <mypasswd>
-keystore <myweb-truststore.jks>
-file <myca.cer>
```

7. Copy the new truststore into the following location on the Policy Server host:

```
<install dir>\PolicyServer\server\certificates
```

8. Import the CA reply file that you downloaded in step 3e into the keystore that you created in step 1. When prompted to install reply, type yes.

```
keytool -import -alias web
-keystore <myweb.jks>
-trustcacerts
-file <ca-reply-file>
-keypass <mypasswd>
```

```
-storepass <mypasswd>
```

 **Note:** Ensure that the `is web` and `must be applied to the keystore` and not the trustore. If you get the following error, it means that your `<ca-reply-file>` is not complete.

```
keytool error: java.lang.Exception: Failed to establish chain from reply
```

In such a situation, you must manually concatenate the `<ca-reply-file>` with your CA cert and rerun the step 8.

9. View the keystore entry using the following command:

```
keytool -list -alias web  
-keystore <myweb.jks>  
-storepass <mypasswd>  
-v
```

The process returns something similar to the following:

```
Keystore type: JKS  
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: web
Creation date: Aug 7, 2018
Entry type: PrivateKeyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=democc02w12r2.qapf1.qalab01.nextlabs.com,
OU=CompliantEnterprise, O=NextLabs, ST=California, C=US
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Serial number: 49237803
Valid from: Tue Aug 07 12:34:57 SGT 2018 until: Mon Nov 05 12:34:57 SGT
2018
Certificate fingerprints:
    MD5: E1:F8:BF:20:98:3C:26:A5:FC:93:85:82:DF:28:A4:9E
    SHA1: EB:2E:39:3C:69:B1:A1:84:5E:E8:6C:80:3B:F7:DB:F4:4C:12:FF:00
    SHA256:
        19:D2:D3:B7:67:C1:12:2B:A8:C2:89:36:66:4C:7F:3C:1D:64:37:8B:84:B2:
        7E:B1:DF:85:6D:A6:CD:45:12:CB
        Signature algorithm name: SHA256withRSA
        Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 33 01 06 62 6B 2E 8E 33      99 C6 70 28 A6 F1 45 93  3..bk..3..p(..E.
0010: 6B A1 06 BD                           k...
]
]

#2: ObjectId: 2.5.29.17 Criticality=false
SubjectAlternativeName [
    IPAddress: 10.23.58.76
    DNSName: democc02w12r2.qapf1.qalab01.nextlabs.com
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
```

```

KeyIdentifier [
 0000: FB EE C5 10 BE 6F 1E 38    2E 73 14 7A 6A B5 00 C7 ....o.8.s.zj...
 0010: E7 63 59 E8                .cY.
]
]

Certificate[2]:
Owner: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Issuer: CN=KentCA.NextLabs.com, OU=PS, O=NextLabs, L=Singapore,
ST=Singapore, C=SG
Serial number: 5b613509
Valid from: Wed Aug 01 12:20:25 SGT 2018 until: Thu Aug 01 12:20:25 SGT
2019
Certificate fingerprints:
MD5:  E9:49:45:86:9A:4E:92:41:E2:E2:AE:10:05:63:D6:3A
SHA1: BD:58:51:8F:07:F0:9C:9E:AD:F0:90:D9:28:5F:B0:BB:5A:89:7A:B4
SHA256:
52:04:C4:A6:0F:8A:8F:18:21:AD:12:34:41:06:6B:D7:98:4F:03:27:A8:68:8A:F6:
95:AD:C7:B9:DA:9A:0D:00
Signature algorithm name: SHA256withRSA
Version: 3

```

10. Copy the keystore into the following location on the Policy Server host:

<install dir>\PolicyServer\server\certificates

11. Using the `mkpassword.bat` utility, supplied at <install dir>\Policy- Server\tools\crypt, encrypt the passwords for the keystore and truststore files (myweb.jks step 1 and myweb-trust.jks step 5). This command must be run for each password, where <mypasswd> refers to the password you designated in steps 1 and 5 above:

`mkpassword.bat -w <mypasswd>`

The process returns something similar to the following:

"7542615506752a4200613b55295d16771d730b7e"

 **Note:** This step is necessary because the Control Center uses the passwords to open the files and use their contents, and the passwords should not be accessible in plain text.

12. Modify the configuration file for the Tomcat web application server, `server.xml`, to point to the new keystore and truststore files, and to include the new passwords. By default, this file is placed on the Management Server host, at:

<install dir>\PolicyServer\server\configuration

Open this file, locate the <CE-Apps> section, and supply values for the attributes shown in the following figure. Be sure to place all values inside double quotes:

- **keystoreFile:** Name of the new keystore file <myweb.jks>
- **keystorePass:** Password of the new keystore file, after encrypting with the `makepassword.bat` utility
- **truststoreFile:** Name of the new truststore file <myweb-trust.jks>
- **truststorePass:** Password of the new truststore file, after encrypting with the `makepassword.bat` utility
- **keystoreType:** Do not change this value from JKS
- **truststoreType:** Do not change this value from JKS

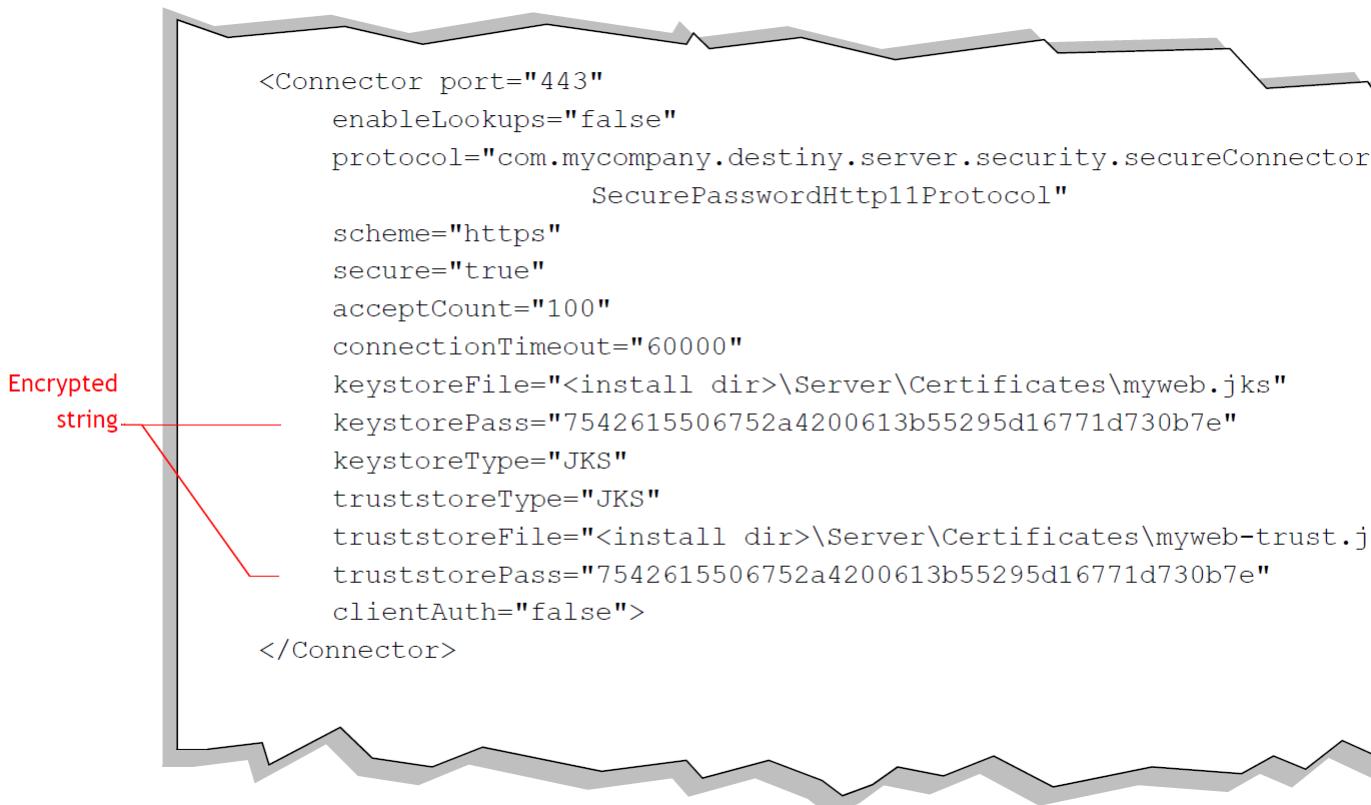


Figure 109: Configuring the web connection port (443)

Encrypting web services components

Encrypting web services communication among the Control Center internal server components requires two files.

- <dcc-keystore.jks>
- <dcc-truststore.jks>

Control Center stores the required DCC certificate in its DCC truststore. In addition, the DCC truststore must contain the certificates for communication with other Control Center components. This means anytime you create a DCC truststore, you must insert certificates for each of the following:

- DCC
- Agent
- Temp-Agent
- PolicyAuthor
- Enrollment

Follow these steps to configure a new internal server (DCC) certificate.

1. Generate the keystore using the following command:

```

keytool -genkeypair
-dname "cn=<commonName>, OU=<organizationUnit>, O=<localityName>,
S=<stateName>, C=<CountryName>"
-alias DCC
-keypass <password>
-storepass <password>
-keyalg DSA
-keystore <mydcc.jks>

```



Note: The presence of both a -keypass and a -storepass is a Java keystore requirement. The two passwords limit access to the keystore: the -storepass allows the viewing of the keystore, and the -keypass allows keys to be used for signing. For more information, see the documentation for the Java keytool.

The -dname value must be unique for all keystores that are inserted into the DCC truststore.

2. Generate a certificate request that can be submitted to a third-party certificate authority (CA). For example,

```
keytool -certreq  
-alias DCC -keypass <password>  
-storepass <password>  
-keystore <myDCC.jks>  
-file <myDCC.csr>
```

3. Submit the certificate request to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:

- a. Submit an Advanced request for a User certificate that enables you to supply a base-64-encoded CMC or PKCS #10 file.
- b. Copy and paste the certificate request from the newly generated certificate request file (myweb.csr above). To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST, as in the following example:

```
-----BEGIN NEW CERTIFICATE REQUEST-----  
MIICgzCCAkECAQAwfzELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAKNBMRiwEAYDVQQHEw1TYW4gTWFOZ  
W8xEzARBgNVBAoTCkJsdWVKdW5nbGUxHDAaBgnVBAsTE0NvbXBsaWFudEVudGVycHJpc2UxHDAa  
...  
7ZpvSDn94tEBPgayBaAAMAsGByqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9D1RzcQi07E0QIUa  
M9W8V6UuYbIzBrR5v9XTu9cV94=  
-----END NEW CERTIFICATE REQUEST-----
```

- c. Select a certificate template where the subject line matches the subject in the .csr file. The subject line should not be changed.
- d. When prompted, download the issued certificate. The certificate is named certnew.cer. You can change the certificate name, which is myDCC.cer in this example.
4. Copy the new certificate into the following location on the Policy Server host:
`<install dir>\PolicyServer\server\certificates`

5. Create a DCC truststore and import the signed certificate into it. For example,

```
keytool -import  
-alias DCC -keypass <password>  
-storepass <password>  
-keystore <myDCC-truststore.jks>  
-file <myDCC.cer>
```



Note: The option -keystore <myDCC-truststore.jks> actually generates the new DCC truststore file. To insert other certificates into the new truststore, you refer to the name you give the new DCC truststore <myDCC-truststore.jks>.

6. Create a Agent truststore and insert the DCC certificate into it. The name of the Agent truststore file must be Agent-truststore.jks. For example,

```
keytool -import  
-alias DCC -keypass <password>  
-storepass <password>  
-keystore Agent-truststore.jks  
-file <myDCC.cer>
```

7. Using the `mkpassword.bat` utility, supplied at <installdir>/PolicyServer/tools/ crypt, encrypt the passwords for the keystore and truststore files (<myDCC-keystore.jks> generated in step 1 and <myDCC-truststore.jks> generated in step 5). This command must be run for each password, where <mypasswd> refers to the password you designated when in step 1 and step 5 above:

```
mkpassword.bat -w <mypasswd>
```

The process returns something similar to

```
"7542615506752a4200613b55295d16771d730b7e"
```

 **Note:** This step is necessary because the Control Center uses the passwords to open the files and use their contents, but the passwords should not be accessible in plain text.

8. Modify the configuration file for the Tomcat web application server, `server.xml`, to point to the new keystore and truststore files, and to include the new passwords. By default, this file is placed on the Management Server host, at:

```
<INSTALLDIR>\server\configuration
```

Open this file, locate the <Engine name=CE-core> section, and supply values for the attributes shown in the following figure, taking care to place all values inside double quotes:

- **keystoreFile:** Name of the new keystore file <myDCC-keystore.jks>
- **keystorePass:** Password of the new keystore file, after encrypting with the `makepassword.bat` utility
- **truststoreFile:** Name of the new truststore file <myDCC-truststore.jks>
- **truststorePass:** Password of the new truststore file, after encrypting with the `makepassword.bat` utility
- **keystoreType:** Do not change this value from JKS
- **truststoreType:** Do not change this value from JKS



Figure 110: Configuring the internal connection port (8443)

Related tasks

[Configuring the agent certificate](#) on page 371

The names of the keystore and truststore files must be `agent-keystore.jks` and `agent-truststore.jks`.

[Configuring the temp agent certificate](#) on page 371

To support the automatic registration of a new Policy Controller, the embedded temporary certificates must be trusted.

Importing and configuring Policy Controller certificates

The Policy Controller requires two certificates: the temp agent certificate and the agent certificate.

During installation, a desktop or server Policy Controller registers with the server using the temp agent certificate. After this initial registration, the new Agent-keystore.jks and Agent-truststore.jks are automatically downloaded to the Policy Controller. The agent certificate is then used from that point on, instead of temp agent.

The procedures for configuring these two certificates are different. Both files must be imported into the new truststore you created for the DCC. But, in order for the new Agent certificates to be downloaded to the client after registration, you must delete the old Agent keystore and truststore files.

Configuring the temp agent certificate

To support the automatic registration of a new Policy Controller, the embedded temporary certificates must be trusted.

To do this, import the default temp agent certificate into the new DCC truststore. Follow these steps.

1. Obtain the Temp_Agent.cer file from NextLabs support.
2. Import the signed certificate into the new DCC truststore. For example,

```
keytool -import  
-alias Temp_Agent  
-storepass <DCCpassword>  
-keystore <mydcc-truststore.jks>  
-file <temp_agent.cer>
```

 **Note:** When you supply the `-keystore <mydcc-truststore.jks>` option above, you are referencing the new DCC truststore file. The `-storepass` password must be the same one you created for the DCC keystore.

Related tasks

[Encrypting web services components](#) on page 368

Encrypting web services communication among the Control Center internal server components requires two files.

Configuring the agent certificate

The names of the keystore and truststore files must be `agent-keystore.jks` and `agent-truststore.jks`.

Follow these steps to configure a new agent certificate.

1. Generate the keystore using the following command:

```
keytool -genkeypair  
-dname "cn=<commonName>, OU=<organizationUnit>, O=<localityName>,  
S=<stateName>, C=<CountryName>"  
-alias agent  
-keypass <password>  
-storepass <DCCkeypass>  
-keyalg DSA  
-keystore agent-keystore.jks
```

 **Note:** The `-dname` value must be unique for all keystores that are inserted into the DCC truststore. The `-storepass` password must be the same one you created for the DCC keystore.

2. Generate a certificate request that can be submitted to a third-party CA. For example,

```
keytool -certreq  
-alias agent  
-keypass <password>  
-storepass <DCCkeypass>  
-keystore agent-keystore.jks -file Agent.csr
```

3. Submit the certificate request to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:
 - a. Submit an Advanced request for a User certificate that enables you to supply a base-64- encoded CMC or PKCS #10 file.
 - b. Copy and paste the certificate request from the newly generated certificate request file (`Agent.csr` above). To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST:


```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIICgzCCAKECAQAwfzELMAkGA1UEBhMCVVMxGzAJBgNVBAgTAKNBMRiwEAYDVQQHEw1TYW4gTWF0Z
W8xEzARBgNVBAoTCkJsDWVKdW5nbGUxHDAaBgnVBAsTE0NvbXBsaWFudEVudGVycHJpc2UxHDAa
...
7ZpvSDn94tEBPgayBaAAMAsGBYqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9D1RzcQi07E0QIUa
M9W8V6UuYbIzBrR5v9XTu9cV94=
-----END NEW CERTIFICATE REQUEST-----
```
 - c. Select a certificate template where the subject line matches the subject in the .csr file. (The subject line should not be changed.)
 - d. When prompted, download the issued certificate. The new certificate is named `cert-new.cer`. You can change the certificate name (in our example, `Agent.cer`)
4. Copy the new certificate into the following location on the Policy Server host:
`<install dir>PolicyServer\server\certificates`

5. Import the signed certificate into the DCC truststore. For example:

```
keytool -import
-alias agent
-storepass <DCCkeypass>
-keystore <mydcc-truststore.jks>
-file <Agent.cer>
```

 **Note:** When you supply the `-keystore <mydcc-truststore.jks>` option in the command above, you are referencing the new DCC truststore file.

 **Note:** The `-storepass` must be the same ones you created for the DCC keystore.

6. Import the signed certificate into the new Agent truststore. The name of the Agent truststore file must be `Agent-truststore.jks`. For example:

```
keytool -import
-alias agent
-storepass <DCCkeypass>
-keystore agent-truststore.jks
-file <Agent.cer>
```

 **Note:** When you supply the `-keystore agent-truststore.jks` option in the command above, you are referencing the new Agent truststore file. The `-storepass` must be the same one you created for the DCC keystore.

Related tasks

[Encrypting web services components](#) on page 368

Encrypting web services communication among the Control Center internal server components requires two files.

Replacing the temp agent certificate in the keystore file

The `temp_agent-keystore.jks` includes a self-signed certificate.

You can replace this certificate with one of your own if you prefer. Replacing the temp agent certificate is not required.



Note: During installation, a desktop or server Policy Controller registers with the server using the temp agent certificate. After this initial registration, the new `agent-key-store.jks` and `agent-truststore.jks` are automatically downloaded to the Policy Controller. The agent certificate is then used from that point on, instead of the temp agent certificate.

Creating a temp agent keystore with a self-signed certificate

1. Generate the keystore and certificate:

```
keytool -genkeypair  
-keyalg DSA  
-alias temp_agent  
-keystore temp_agent-keystore.jks  
-storepass <password> -keypass <password>  
-validity <days>  
-dname "CN=Temporary Agent, OU=CompliantEnterprise, O=<company>, L=<city>, ST=<state>, C=<country>"
```

2. Extract the certificate from the newly generated keystore:

```
keytool -exportcert  
-keystore temp_agent-keystore.jks  
-alias temp_agent -file temp_agent.cer  
-keypass <password> -storepass <password>
```

The `temp_agent-keystore.jks` file with a self-signed certificate is created.

3. Proceed to Certificate Replacement and Installation.

Creating a Temp Agent Keystore with Your Own Certificate

1. Copy the existing `temp_agent-keystore.jks` from your Policy Server certificates folder into your working folder.
2. Delete the existing temp certificate from the keystore:

```
keytool -delete -alias temp_agent -keystore temp_agent-keystore.jks
```

3. Add your certificate to the keystore:

```
keytool -import  
-alias temp_agent  
-file mycert.cer  
-keystore temp_agent-keystore.jks  
-keypass <password> -storepass <password>
```

The `temp_agent-keystore.jks` file with a self-signed certificate is created.

4. Proceed to Certificate Replacement and Installation.

Certificate Replacement and Installation

1. Copy the existing `dcc-truststore.jks` file from your Policy Server certificates folder to your working folder.
2. Delete the old temp agent certificate from the dcc truststore:

```
keytool -delete -alias temp_agent -keystore dcc-truststore.jks
```

3. Add the new certificate to the DCC truststore:

```
keytool -import  
-alias temp_agent  
-file temp_agent.cer  
-keystore dcc-truststore.jks  
-keypass <password>  
-storepass <password>
```

 **Note:** If you are using your own certificate, use its name instead of `temp_agent.cer`.

4. Copy `dcc-truststore.jks` to the Policy Server certificates folder.
5. Copy `temp_agent-keystore.jks` to the Policy Server certificates folder.
6. Copy `temp_agent-keystore.jks` to the Policy Controller certificates folder.

Configuring Enrollment Manager and Property Manager certificates

By default, the certificate for the Enrollment Manager is stored in `enrollment-keystore.jks`.

Use the following procedure to configure a new certificate.

1. Generate the keystore. It must be named `enrollment-keystore.jks`. The default -key- pass and -storepass value is `password`, but should be changed. In our Java Keytool example, this is done using the following command:

```
keytool -genkeypair -dname "cn=<commonName>, OU=<organizationUnit>, O=<localityName>, S=<stateName>, C=<CountryName>" -alias Enrollment -keypass password -storepass password -keyalg DSA -keystore enrollment-keystore.jks
```

 **Note:** The `-dname` value must be unique for all keystores that are inserted into the DCC truststore. The `-keypass` and `-storepass` values are `password` unless this default is changed. To change the default password for the certificates for this server component, see Replacing Control Center certificates.

2. Generate a certificate request that can be submitted to a third-party CA. For example,

```
keytool -certreq  
-alias Enrollment  
-keypass password  
-storepass password  
-keystore enrollment-keystore.jks  
-file <myEnrollment.csr>
```

3. Submit the certificate request to the CA. The procedure for doing this depends on your CA, but generally includes the following steps:

- a. Submit an Advanced request for a User certificate that enables you to supply a base-64- encoded CMC or PKCS #10 file.
- b. Copy and paste the certificate request from the newly generated certificate request file (`myEnrollment.csr` above). To ensure the entire request is included, copy everything from BEGIN NEW CERTIFICATE REQUEST to END NEW CERTIFICATE REQUEST:

```
-----BEGIN NEW CERTIFICATE REQUEST-----  
MIICgzCCAkECAQAwfzELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAkNBMRIwEAYDVQQHEw1TYw4gTWF0Z  
W8xEzARBgNVBAoTCkJsdWVKdW5nbGUxHDAaBgnVBAsTE0NvbXBsaWFudEVudGVycHJpc2UxHDAa  
...  
7ZpvSDn94tEBPgayBaAAMAsGByqGSM44BAMFAAMvADAsAhRv5iI3rNWbd5mGZ9D1RzcQi07E0QIUa  
M9W8V6UuYbIzBrR5v9XTu9cV94=  
-----END NEW CERTIFICATE REQUEST-----
```

- c. Select a certificate template where the subject line matches the subject in the .csr file. Do not change the subject line.
- d. When prompted, download the issued certificate. The new certificate is named `cert-new.cer`. You can change the certificate name, which is `myEnrollment.cer` in this example.

4. Copy the new certificate into the following locations on the Policy Server host:

- <install dir>\PolicyServer\server\certificates
- <install dir>\PolicyServer\tools\enrollment\security

5. Import the signed certificate into the DCC truststore. For example:

```
keytool -import  
-alias Enrollment
```

```
-storepass <dcckeypass>
-file <myEnrollment.cer>
-keystore <myDCC-truststore.jks>
```

 **Note:**

When you supply the `-keystore <myDCC-truststore.jks>` option in the command above, you are referencing the new DCC truststore file. The `-storepass` password should be the same one you defined for the DCC keystore.

6. Replace the `enrollment-keystore.jks` in `<installDir>\PolicyServer\tools\enrollment\security` with the new one which is located in `installDir>\PolicyServer\server\certificates`.
7. Restart the Control Center Policy Server service.

Importing an Existing Certificate

This section describes the procedure for replacing a certificate by importing an existing certificate.

The previous section, Generating a new certificate, described the procedures for replacing certificates for each Control Center component by generating new keystores and new certificates. Use this procedure if your organization requires that you use your own certificates.

-  **Note:** The example keytool commands in the following procedure demonstrate replacing the DCC certificates in the default DCC keystore (`dcc-keystore.jks`) and truststore (`dcc-truststore.jks`). If you are replacing the certificate for a different component, you would specify different values for the certificate's alias and keystore.

1. Make backups of the NextLabs keystore and truststore that contain the certificate you want to replace. For example, to replace the DCC certificate, back up `dcc-keystore.jks` and `dcc-truststore.jks`. These files are in the following location on the host where Policy Server is installed:

```
<Install Directory>\PolicyServer\server\certificates
```

2. Copy the keystore and truststore to another folder, and run the keytool utility from this folder.

3. Delete the current certificate from the keystore. The following is an example of the keytool command to run:

```
keytool -delete -alias dcc -keystore dcc-keystore.jks -storepass <password>
```

4. Delete the current certificate from the truststore. The following is an example of the command to run:

```
keytool -delete -alias dcc -keystore dcc-truststore.jks -storepass <password>
```

5. Import the keystore that contains the certificate you want to use into the NextLabs key-store. The following is an example of the command to run:

```
keytool -importkeystore -srckeystore <source keystore> -destkeystore dcc-keystore.jks -srcstorepass <source keystore password> -deststorepass <NextLabs keystore password> -srcalias <alias of certificate to use> -destalias dcc
```

6. Change the certificate's password. The password you specify for `-new` must be the same password you specify for `-storepass`.

```
keytool -keypasswd -alias dcc -new <password> -keystore dcc-keystore.jks -storepass <password> -keypass <current password>
```

7. Import the certificate from your truststore by importing the truststore into the NextLabs truststore.

```
keytool -importkeystore -srckeystore <source truststore> -destkeystore dcc-truststore.jks -srcstorepass <sourcetruststore password> -deststorepass <NextLabs truststore password> -srcalias <alias of certificate to use> -destalias dcc
```

8. Copy the updated keystore (`dcc-keystore.jks`) and truststore (`dcc-truststore.jks`) to the following location on the host where Policy Server is installed:

`<Install Directory>\PolicyServer\server\certificates`

Replacing Policy Controller Certificates

The certificates necessary for Policy Controllers are distributed to Policy Controllers after their first registration with the Policy Server.

In order to download the new version of these certificates, you must re-register with the Policy Controller. This must be done on all devices where Policy Controllers are installed. Follow these steps.

1. On the host where the Policy Controller is installed, stop the Policy Server Enforcer Service using **Control Panel > Administrative Tools > Services > Policy Server Enforcer Service**.
2. Locate the `bundle.bin` file at `<installdir>\Policy Controller`. Delete it.
3. Locate the old certificates that were distributed on the default installation at `<install-dir>\Policy Controller\config\security`. Delete the following files:
 - `agent-keystore.jks`
 - `agent-truststore.jks`
 - `agent-secret-keystore.jceks`

Configuring communication between Control Center and Policy Controller when using middleware plug-ins

 **Note:** Perform the steps in this section only if the Policy Controller and Control Center version is 8.7 or later.

1. On Control Center, generate a `bundleSigning.cer` file. For example:
 - a. `keytool -genkey -alias bundleSigning -keystore dcc-keystore.jks -dName "CN=BundleSigning, OU=CompliantEnterprise, O=NextLabs, L=San Mateo, ST=CA, C=US" -keyalg DSA -keypass <passwd> -storepass <passwd> -validity 3650`
 - b. `keytool -selfcert -alias bundleSigning -keystore dcc-keystore.jks -keypass<passwd> -storepass <passwd> -validity 3650`
 - c. `keytool -export -alias bundleSigning -keystore dcc-keystore.jks -keypass <passwd> -storepass <passwd> -file bundleSigning.cer`
2. On Control Center 8.7, import the `bundleSigning.cer` file into the `agent-truststore.jks` file , by running the following command:
`keytool.exe -importcert -alias bundleSigning -keystore <installed-location>\PolicyServer\server\certificates agent-truststore.jks -storepass <passwd> -file <copied-location>\bundleSigning.cer`
3. Restart Control Center.
4. On the Policy Controller 8.7, perform one of the following steps:
 - If the Policy Controller is already registered, follow these steps:
 - Re-register the Policy Controller:
 - a. Stop the Policy Controller. This requires the Administrator password.
 - b. Delete the following files or any other files except the `temp_agent-keystore.jks` file.
 - `config/security/agent-keystore.jks`
 - `config/security/agent-secret-keystore.jceks`
 - `config/security/agent-truststore.jks`
 -  **Important:** Make sure you do not delete the `temp_agent-keystore.jks` file.
 - c. Delete the `bundle.bin` file.

d. Delete the registration file.

config/registration.info

e. Start the Policy Controller.

- Alternatively, if you do not want to re-register the Policy Controller, follow these steps:

a. Stop the Policy Controller. This requires the Administrator password.

b. Delete the bundle.bin file if it exists.

c. Import the bundleSigning.cer to the agent-truststore.jks file that is available at config/security.

```
keytool.exe -importcert -alias bundleSigning -keystore  
agent-truststore.jks -storepass <passwd> -file <copied-  
location>\bundleSigning.cer
```

d. Start the Policy Controller.

- If you have newly installed Policy Controller, restart the Policy Controller.

Glossary

ABAC

Attribute based access control. ABAC uses attributes, or properties, such as information about the user or the environment, in authorization requests for access to assets. Access is allowed or denied based on policies. For more information, go to:https://nccoe.nist.gov/projects/building_blocks/attribute_based_access_control

Active Control Policy Language (ACPL)

The language used internally to represent, store, and manage components and policies. Referred to as ACPL ("ACK-pull") it is also used to express advanced conditions in component and policy properties.

Action components

Components used to define actions in policies. Action components serve as the verbs in policies: read, copy, print, cut and paste, attach to email, and so on. After they are defined, action components are available as the logical verbs in ACPL, the language in which policies are expressed. Action components are defined in the Components section of the console.

Activity Journal

One of the four data stores used by Control Center. The Activity Journal stores information sent by all enforcers about how subjects access and use information, and also on policy enforcement. When you use the Reporter console to generate reports, you are querying data from the Activity Journal. The Activity Journal can be maintained either on the internal database supplied with Control Center, or on an external Oracle, PostgreSQL, or Microsoft SQL database.

Policy Assistants

A set of optional software plug-ins with supporting material and utilities, designed to automate and simplify specific workflows connected with specific user actions, such as sending email with attachments or copying data to removable media. Policy Assistants function in combination with specific enforcers, predefined custom obligations, and predefined policy objects.

ad hoc policies

See policy-on-demand.

Administrator console

One of the Control Center user interfaces. The Administrator console enables administrators to configure and monitor the Control Center system.

alerts

Notifications that are sent to administrators when specified policy enforcement activities occur. Together with monitors, alerts can provide continuous coverage of key policy enforcements in an enterprise, as well as a notification system that warns administrators when action is required.

auditor

A logical component of Policy Enforcer software, responsible for capturing document access and use by policy subjects, as well as all policy enforcement events, and writing them to the Activity Journal database.

template

One of the set of irreducible actions, such as open, delete, copy content, or email, that are available for combining into Activity Journal. Each action component can consist of one or more basic actions.

bundle

See policy bundle.

cancel

One of the actions you can take to change the deployment status of policies and components. You can cancel an object if it has been submitted and scheduled for deployment, but not if it has already been deployed. Deployed objects must be deactivated before they can be canceled.

Control Center

The enterprise-wide platform where policies are constructed, stored, and managed, and where audit data is gathered and managed. Control Center comprises three major pieces—the management server, the Report Server, and ICENet server—plus several data stores, which are generally stored in a dedicated, external database. NextLabs Entitlement Management and Enterprise Data Protection

products can be installed on top of this platform to provide the enforcer software and specialized applications for solving clearly defined business problems.

components

The building blocks of policies. Components represent classes or categories of entities in the physical network environment.

connection file

A plain-text file used by the Enrollment Manager utility. The Connection File provides the information Control Center needs to connect to the LDAP directory to be used as an enrollment source. The name and path of this file are specified as a command line argument in several enrollment-related commands, and is then stored in internal tables for use in updating and synchronizing. A template for this file is provided during installation; values in the file can be modified as needed.

deactivate

The act of changing a policy or component from a deployed state to an inactive state. Deactivated objects can be reactivated later or deleted as needed. Objects that have been deployed must be deactivated before they can be deleted.

definition file

A plain-text file that is used in connection with the Enrollment Manager utility to provide the information Control Center needs to map the data you are enrolling from LDAP directories or LDIF files, to the format Control Center uses in its Information Network Directory tables. This file is specified as a command line argument for both enrolling and updating/synchronizing procedures. Like the connection file, it is provided as a template that users can edit and save for their use.

Delegated Administration

A method of managing user access to the Control Centers. Delegated Administration enables administrators to manage user access to the Control Center consoles as well as policies, and policy objects, by creating rules based on user attributes and conditions. Using rules and attributes mirrors the way policies are used to control access to resources and eliminates the need to create and apply roles to users or groups of users.

delete

See deactivate.

deployed

One of the possible states of a policy or component. When an object is deployed, it is distributed to all relevant enforcers in the system. It then is used to govern the way components access and use resources.

Desktop Enforcers

One of two types of policy enforcers. Desktop Enforcers are applications that run on Windows desktop or laptop PCs. These Enforcers monitor and enforce how any subject using the PC accesses and uses document resources.

discretionary policies

See policy-on-demand.

effect

The enforcement action to be taken when a policy is enforced. Effects include Deny and Allow.

enforcer

See policy enforcers.

enforcer status

The state of the policy enforcer. Policy enforcers can be in either of two states: running or not running. You can check this with the local controls for Desktop Enforcers, and for all enforcers by checking the latest heartbeat displayed on the Policy Enforcer Status tab in the Administrator console. When an enforcer is running, this tab displays a policy status, which indicates whether the most recently submitted versions of policies have been deployed to that enforcer.

enrollment

The process by which an organization's existing information network of users, hosts, and other entities are imported from LDAP directories or LDIF files, into Control Center.

enrollment manager

A system utility that provides a unified tool for enrolling data from LDAP directories or LDIF files, updating and synchronizing enrollments of users, groups, and hosts.

File Server Enforcer

One of several types of predefined policy enforcers; a collection of software processes that run on Windows file servers to monitor and control the way all users access the files stored on that server.

filter file

A plain-text file where users can define any set of LDAP filters. This file can be specified as a command line argument in enrollment procedures, so the enrollment is performed only on entities meeting the filter criteria.

heartbeats

Regular messages sent along the communications channel between Policy Controllers and Control Center. Policy Controllers send regular heartbeats to Control Center to indicate that they are running normally. When a heartbeat occurs, the Policy Controller receives any pending policy deployments or configuration updates from Control Center.

Heartbeat plug-ins use this communication channel between Control Center and Policy Controllers to send and receive other types of data. For example, you can build a heartbeat plug-in to send external authorization data, such as licenses, to all Policy Controllers or to send installation or upgrade information about each Policy Controller to store in Control Center.

ICENet

A protocol used for communication between policy enforcers and the Control Center. All such communication passes through the ICENet server on the set of software servers and other modules that constitute the heart of the Control Center platform. The main components of the Control Center are the Policy Management Server and Policy Master data store, the Management Server and Information Network Directory, the Intelligence Server and Activity Journal, one or more ICENet Servers, and the Reporter console and Administrator console servers. Each enforcer is equipped with an ICENet client.

ICENet client

The component that handles remote communication with the ICENet server.

ICENet server

The Control Center component that manages communication between all policy enforcers and the set of software servers and other modules that constitute the heart of the Control Center platform. The main components of the Control Center are the Policy Management Server and Policy Master data store, the Management Server and Information Network Directory, the Intelligence Server and Activity Journal, one or more ICENet servers, and the Reporter console and Administrator console servers. In a production environment, the ICENet server is usually installed on a different host than the rest of the components.

information network

The collective term that refers to all of an organization's information resource servers, desktops, network configuration, applications, and organizational structure.

information network directory (IND)

The internal model Control Center uses to represent the organization's information network. The IND can be thought of as a repository that maps to the data that is enrolled from an LDAP directory or LDIF files, about physical entities in the network such as users and groups. Physically, the IND is stored in the same database as the Activity Journal—either Control Center's embedded database, or an external or PostgreSQL, Oracle, or SQL Server database.

management database

One of the three databases used by Control Center. The management database stores data about authorized users and the operation of the Administrator console.

management server

The Control Center component that centralizes management of all system components.

monitors

Predefined criteria that identify the policy enforcement activities to be tracked. Together with alerts, monitors can provide continuous coverage of key policy enforcements in an enterprise, as well as a notification system that warns administrators when action is required.

object components

One of the two general categories of components; the other is Activity Journal. Object components provide the subjects and objects in policies.

obligations

Instructions that must be performed along with the authorization decision. Built-in obligations include instructions to make a log entry in the Activity Journal, display a notification to the subject, and send

an email notification to a specified recipient. Custom obligations are defined by users and are executed either by the PDP (.exe files for desktops) or the PEP. For example, an obligation might instruct the PEP to notify users why they are not allowed to perform certain operations, and the message itself might be included in the obligation. In other cases, obligations might instruct the PEP to record the allowed user action in an audit database; encrypt a value before allowing the user to store that value; or inject a security filter in a SQL query to restrict the rows being returned, before allowing the user to run the query against a given database table. Policies can include one or more of each of these obligation types. Log entries are required for all On Deny enforcements.

PAP

See Policy Administration Point.

PDP

See Policy Decision Point.

PEP

See Policy Enforcement Point.

PIP

See Policy Information Point.

policy

A rule designed to control information use in an organization. Policies consist of components combined with logical operators, variables such as obligations, and time-based contexts.

Policy Adapter

The interface to the specific application or system you want to enforce. Each enforcer consists of two logical components, a Policy Adapter and Policy Controller. Each of the available predefined enforcers includes a different Policy Adapter, specific to the Windows Desktop, SharePoint Server, Linux system, Microsoft SharePoint, and so on. If you need to enforce policies on other systems—such as proprietary CRM applications—you need to create a PEP.

Policy Administration Point

The Control Center server. The PAP has these key components:

- Policy management service: The service that provides the interface used to author and manage policies.
- ICENet: A server component that communicates with the Cloud PDP to send policies and receive user activity records.
- Reports, Monitoring and Alerts: The interface used to view and track user activity.
- Database repositories: Used to store policies and audit records.

policy bundle

The set of policies and required component definitions that are defined and active for each enforcer running in the system. Each enforcer has only one bundle stored locally at any one time, and it contains all the information it requires for all policies currently deployed to it. Every heartbeat interval, each enforcer contacts the Control Center inquiring if there are any new or updated policies that should be deployed to that enforcer; if there are, the Control Center sends a new bundle that overwrites the one currently deployed there.

policy component

See components.

Policy Decision Point (PDP)

The entity that evaluates applicable policies and computes authorization decisions.

Policy Enforcer Configuration

A tab in the Administrator console main window, where policy enforcer profiles are defined.

policy enforcer profiles

A named set of configuration settings. These are defined and assigned to policy enforcers in the Administrator console on the Policy Enforcer Configuration tab.

policy enforcer status

One of sub-tabs available on the Administrator console Status tab. This tab shows the status of all enforcers in the network, and provides filters to select and view categories of enforcers.

policy enforcers

The generic term for Control Center clients that monitor resource access and enforce policies at the point of use. Each policy enforcer consists of a policy enforcement point (PEP), a Policy Controller, and a Policy Adapter.

NextLabs policy enforcers enforce policies on specific platforms and systems, such as Windows desktops, Microsoft Office Communicator, Microsoft Outlook, or Linux systems. In addition, SDKs enable users to create policy enforcers that can enforce policies on in-house applications or third-party applications for which NextLabs enforcers are not yet available.

Policy Enforcement Point (PEP)

The software component that performs access control by making authorization requests to PDPs (Policy Decision Points) and enforcing decisions.

Policy Information Point (PIP)

The component that is the source of user, resource, and environment attributes. A PEP (Policy Enforcement Point) or PDP (Policy Decision Point) interacts with the PIP to obtain attributes. In an enterprise deployment, a typical PIP would be an LDAP server, HR database, or other data repository with license keys.

Policy Management Server

A server component of the Control Center. The Policy Management Server is responsible for policy management including policy creation and editing, deployment and status tracking, and lifecycle management.

policy map

The optimized policy packages that are deployed to each policy enforcer and perform real-time evaluation and enforcement even when disconnected from the network.

Policy Master

A database maintained by the Control Center, where the Policy Management Server creates and stores definitions and deployment information for policy objects. The Policy Master can be located either on the internal PostgreSQL database, or on an external one.

policy modeling

The process of defining the metadata to represent actors (Subjects) requesting authorization for a given set of objects (Resources).

policy object

A generic term sometimes used to refer to policies and policy components.

policy-on-demand

Policy-on-demand requests, also known as discretionary policies, are ad-hoc policies or requests used to include attributes and values that are not part of deployed policies in policy requests. The REST API extends the XACML 3.0 specification to support policy-on-demand. See REST API for external PDPs.

profile

See policy enforcer profiles.

Property Manager

A system utility that provides simple way to view and customize the properties, such as LDAP attributes, of enrolled users, hosts and applications.

report query

The group of filter settings available in the Reporter console that control how report content is extracted from the Activity Journal.

Report Server

A major component of Control Center that is responsible for storing, managing, and serving data on end-user activity and policy enforcement. This information is stored in the Activity Journal database. The Report Server is the back end of the Reporter console.

report settings

The group of settings available in the Reporter console that control how reports are formatted.

Reporter console

One of the user interfaces in Control Center. The Reporter console is a web-based application used to create reports as well as monitor and track user activity. It is the front end of the Report Server.

reports

The result of a query run on information stored in the Activity Journal, showing details related to information access and use and policy enforcement in the network. Reports can be displayed in detail (grid) format or as bar charts.

resource components

Components that can be included in policy definitions. Resources, such as documents or servers, are the grammatical object of actions in the ACPL language.

resource types

The policy model templates that include information about attributes, actions, and obligations available to components and policies.

roles

Named sets of permission settings that govern how a user can access Control Center applications and edit components and policies. Predefined roles include Business Analyst, Policy Administrator, Policy Analyst, Report Administrator, and System Administrator. There are default settings for each predefined role, but they can be customized as needed. Roles are defined on the Roles subtab of the Users And Roles tab.

scheduled

An intermediate state that objects can occupy, in the standard object model. Components and policies can be deployed immediately or at a specified time in the future. In the latter case, the object's state is Scheduled for Deployment.

shared reports

Reports that are available to multiple users. When reports are saved, they can be marked as shared.

Shared reports appear on a separate tab in the Reporter console. See Saving reports.

states

The current status of a policy or a component. There are seven possible states: Editing, Submitted for Deployment, Deployed, Deactivated, and Deleted. An object's state determines the actions that can be performed on it.

System Administrator

One of the roles available in the Administrator console. See roles.

System Status

The upper left pane on the Status tab in the Administrator console main window, displaying the number of enforcers that have failed to connect within the last 24 hours, and the number to which the latest version of a policy or component has not yet been deployed.

Index

Special Characters

- _ wildcard character [222](#)
- * wildcard character
 - in advanced conditions [182](#)
- % wildcard character [222](#)

A

- abstraction [167](#)
- Action field [217](#)
- Active Directory
 - and load balancing [43](#)
- Activity Journal
 - and Reporter [209](#)
 - configuring log queue manager [145](#)
- ad hoc policies [288](#)
- ad.default.def (definition file)
 - example of [101](#)
- ad.sample.default.conn (connection file)
 - example of [49, 92](#)
- ad.sample.default.def (definition file) [54](#)
- add command (Property Mgr) [101](#)
- Administrative Password (profiles) [117](#)
- Aggregator property [240](#)
- Aggregators property [237](#)
- Alert property [237](#)
- alerts
 - analyzing [243](#)
 - filtering [241](#)
 - sorting [241](#)
 - viewing
 - by monitors [244](#)
 - by tag [243](#)
 - by time [245](#)
- aliases.txt [86, 89](#)
- Allow
 - filtering by [217](#)
- anomalies in report data [235](#)
- app.sample.default.def (definition file) [54, 64, 84](#)
- app.sample.default.def (definition file) example of [84](#)
- AppDiscovery [83](#)
- Application ID [43](#)
- application-id
 - in Azure AD connection file [52](#)
- application-key
 - in Azure AD connection file [52](#)
- applications
 - discovering and enrolling [83](#)
 - enrolling different versions [85](#)
- archiving data tables
 - automated feature [128](#)
- attributes
 - custom
 - See custom properties [98](#)
- Audit Log Upload Frequency [117](#)
- audits
 - targeted [120](#)

- using enforcer profiles [120](#)
- authentication
 - config settings [137](#)
- Authentication key [43](#)
- Azure Active Directory application [42, 42](#)
- azure-oauth-authority
 - in Azure AD connection file [52](#)

C

- C SDK
 - files [322](#)
- case sensitivity
 - computer and user properties
 - custom [101](#)
 - default [99](#)
 - in report filtering [221](#)
 - of Enrollment Mgr commands [45](#)
 - of Property Mgr commands [100](#)
- portal content properties
 - custom [104](#)
 - default [103](#)
- certificates
 - for SharePoint servers [95](#)
- character sets [228](#)
- cloning
 - policies [186](#)
- collation values
 - for non-Latin characters [228](#)
- command line arguments
 - Enrollment Manager [45](#)
- compiling sample enforcer (C) [320](#)
- components
 - cloning [186](#)
 - computers
 - custom properties [101, 101](#)
 - exporting [201](#)
 - portal
 - default properties [103](#)
 - user
 - custom properties [101, 101](#)
- computer components
 - defining custom properties [101, 101](#)
 - properties of [99, 99](#)
- configuration
 - Reporter [140](#)
- configuration files
 - encrypting passwords [144](#)
- configuration.xml
 - <UserRepositoryConfiguration> section [137](#)
- connection files
 - description of [48](#)
 - for SharePoint groups [92](#)
- connection pools
 - adjusting max size [124](#)
 - minimum size [124](#)
- connections
 - required for enrolling [40](#)

copying
 components 186
 policies 186
 creating
 monitors 237
 policies 349
 reports 217
 subpolicies 184
 tags 237
 crypt.jar 97
 CSampleconnectorDlg 320
 custom attributes, *See* custom properties
 custom enforcers
 installing 327
 required DLLs 327
 Custom Journaling Options 119
 custom properties
 CustomAttributes section in config file 140
 of portal content components 104
 of users
 computers
 & groups 101
 hosts and groups 101
 users
 computers
 & groups 101

D

DABSLocation 114
 dashboard
 about 212
 and Activity Journal 16, 212
 dashboard data 213
 database
 and connection pools 122
 column collations 228
 description of 121
 management
 auto-archiving 128
 synchronizing tables 126
 truncating archive data 130
 unarchiving data 129
 managing size 131
 optimizing report generation 131
 types supported 121
 utilities
 DB Init 133
 truncate 130
 Vacuum DB 132
 Vacuum DB utility 132
 weekly backup 131
 Date Mode field 228
 Date Selection field 228
 DB Init utility 133
 deactivating 197
 default
 properties
 of computer components 99
 of user components 99
 DefaultFrom (config setting) 140
 DefaultTo (config setting) 140

defining 217
 defining queries
 by actions 224
 by policy 221
 by users 221
 definition files
 for LDAP directories 54
 for LDIF files
 applications 84
 users
 hosts and groups 54
 must rename templates 54, 58
 delete command (Enrollment Mgr)
 after failed enrollment 39
 delete command (Property Mgr) 103
 deleting
 hosts from profiles 120
 Deny
 filtering by 217
 dependency checking 195
 deployment
 definition 195
 Description field 228, 237
 desktop enforcers
 stopping 111
 Desktop Enforcers
 reconfiguring 114
 uninstalling 113
 determining logging volume of 131
 dFS 89
 dialog class (of sample enforcer) 320
 Display Columns field 217
 Distributed File System (Microsoft) 89
 DLLs
 required 327
 document activity
 audits
 setting options 117
 duration
 specifying 239
 values 239
 Duration field 237

E

email
 configuration settings 140
 notification
 contents of 175
 setting the From field 175
 email notification
 configuration settings 140
 on sync failures 73
 EnableADDirChgReplications 50, 52
 encrypt utility 97
 encrypting passwords 144
 enforcer profiles
 assigning to enforcers 114
 enforcers
 permissions for service accounts 113
 types of 114

enrolling
 applications 83
 file shares
 Linux servers 88
 Windows hosts 86
 filter files for 56
 from multiple domains 37, 77
 location sites 89
 SharePoint groups 91
enrollment
 definition 37
 enrollment log 152
 failures 39, 79
 from AD
 about load balancing 43
 manual 37
 prerequisites 43
 summary of different types 40
 what to enroll 37
Enrollment Manager
 connections required 40
 delete command
 after failed enrollment 39
 description of 40
 enrolling
 applications 83
 SharePoint groups 91
 users and hosts 63
 list command 77
 summary of CLI command arguments 45
 sync command 73
 user permission required 40
enrollment_name parameter 45
Equals operator 222, 224
exporting policies and components 201
external domain authentication
 configuring 138

F

File Server Enforcers
 reconfiguring 114
 stopping 111
 uninstalling 113
file shares
 enrolling
 Linux 88
 Windows 86
FileSystemLogConfiguration 145
filter examples 224
filter files 56
filtering
 by action 224
 by policy 221
 by users 221
filters
 in AD connection file 50
 setting properties 57
Firefox browser
 TLS 1.0 required 209
From field (notification email)
 configuring 140, 175

G

global audit
 setting activity audit level for 117
 setting audit level 117
Granting permission 42, 42, 43
Group By property 237, 240
groups
 SharePoint
 enrolling 91

H

heartbeat
 enforcers
 changing default value 117
 of enforcers 116
 of Policy Server components 140
 policy condition 175
 policy enforcers
 monitoring 114
heartbeat rate
 resetting 327
HeartbeatRate (config setting) 140
Help
 for mkpassword utility 144
 for Property Manager utility 100
Help link 212
Hide
 in enforcer status display 114
hosts
 of policy enforcers
 determining 114

I

ICENet Servers
 associating with a profile 117
 configuring log queue 145
 initial registration 107
 install wizard vs. profile 107
 load balancing 109
 moving enforcers among 114
 reassigning profiles to 109
implied actions 235
Import Locations
 enrolling
 sites 89
 syntax 89
In operator 222, 224
Information Network Directory
 and custom properties 101
Initialize (API) 327
installing
 C SDK 321
 custom enforcers 327
 Java SDK 322
 Policy Controller 257, 321
 sample enforcer (C) 321
IsPagingEnabled 50, 52

J

java.security.krb5.kdc (config setting) 138
java.security.krb5.realm (config setting) 138
java.sql.SQLException 124

K

Keytool utility 95

L

Last Heartbeat
 of policy enforcers 114
Last Policy Update
 policy enforcers 114
LDAP
 filter files 56
LDAP directory
 definition files 54
LDIF
 enrolling applications 83
LDIF definition files
 for applications 84
 for users
 hosts and groups 54
ldif.sample.default.def (definition file)
 example of 54
Level property 237
Like operator 222, 224
Linux
 enrollments for 38
Linux file servers
 enrolling file shares 88
list command (Enrollment Mgr) 77
list command (Property Mgr) 103
load balancing
 and enrolling from AD 43
 ICENet Servers 109, 109
location sites
 definition of 89
 enrolling 89
locations file
 required format 89
logging
 enrollment log 152
logging levels
 of enforcers 151
logging.properties file 151
login.password (config setting) 137
Logout link 211
logs
 configuring queue 145
 setting upload frequency 117
Lookup tool 221

M

MachineList file 86, 86, 86
Max Results field 217
MaxPoolSize (config setting) 122
Message property 237

MessageHandlers (config settings)
 description of 140
 with sync operations 73
Microsoft Distributed File System 89
mkpassword.bat 144
monitor policies
 obligations for 187
monitoring
 permissions 236
monitors
 about 209
 creating 237
 deactivating 240
 deleting 240

N

Name field (Create Monitor dialog) 237
Name field (Save dialog) 228
Network Information Directory
 and Enrollment Manager 40
New application registration 42, 42
non-Latin characters 228
Not Equals operator 222, 224
notification email
 contents of 175

O

object components
 properties of 99
obligations
 for monitor policies 187
Other Criteria field 217

P

password
 default for enforcer profiles 117
 for stopping enforcers 117
 for uninstalling enforcers 117
 in LDAP directory connection file 50
 in portal connection file 92
 to uninstall enforcers 113
passwords
 encrypting 97, 144
 enrollment utilities 97
 for stopping enforcers 117
 for uninstalling enforcers 117
 of enforcer profiles 117

Paste

 limited obligation support 217

permissions
 for enforcer service accounts 113

policies
 cloning 186
 deactivating 197
 exporting 201
 modifying 197
 monitor
 obligations for 187
 submitting 195

policies on demand 288
policy components
 deactivating 197
 modifying 197
 submitting 195
Policy Controller
 components of 260
 installing 257, 321
Policy Controller for Java 271
Policy Criteria field 217, 222
Policy Decision field 217
Policy Enforcer Status 114
policy enforcers
 reconfiguring 114
Policy Name field 217, 221
policy objects
 exporting 201
Policy Server
 starting and stopping 350
Policy Up to Date
 monitoring 114
portal content components
 custom properties 104
 default properties of 103
portals
 in connection file 92
PostgreSQL
 vacuum DB utility 132
Profile Name
 of policy enforcers 114
profiles
 deleting 120
 moving between ICENet servers 109
 setting password 117
 using in audits 120
properties
 deleting 103
 listing 103
 of computer components 99
 of portal content components
 custom 104
 of user components
 default 99
Property Manager
 add command 101
 delete command 103
 list command 103
 summary of CLI command arguments 100
 summary of CLI commands for 100
Push Deployment feature
 enabling 117

Q

queries
 defining
 by actions 224
 by policy 221
 by users 221
QueueManagerUploadSize 145

R

RDP
 in policy context 175
registration
 of ICENet Servers 107
remote connection
 policies based on 175
report period
 and Activity Journal 220
 and time zones 220
 specifying 220
Report Type field 217
Reporter
 configuration 140
reports
 anomalies in 235
 drilling down 227
 examples
 Group by Day chart 234
 Group by Policy chart 231
 Group by Resource chart 233
 Group by User chart 232
 table 225, 230
 running 225
 saving 228
 viewing 227
 with non-Latin characters 228
Resource Criteria field 217, 222
Resource Path Discovery
 description of 86
 MachineList file 86
 output file 86
 using with dFS 89
Resource Type field 217
Resource.h 320
REST API
 Request examples 289
 request format 289
 Request URL 287

S

Samba Directory Mapping utility 88
sample enforcer
 files
 SampleEnforcer.cpp 319
 SampleEnforcer.vcproj 319
Sample Enforcer
 files 322
sample enforcer (C)
 compiling 320
 files
 Resource.h 320
 SampleEnforcerDlg.cpp 320
 Stdafx.cpp 320
 Stdafx.h 320
 overview 319
 user interface 320
sample enforcer(C)
 files
 SampleEnforcer.ico 319

SampleEnforcer.rc 319
SampleEnforcerDlg.h 320

SampleEnforcer.cpp 319
SampleEnforcer.ico 319
SampleEnforcer.rc 319
SampleEnforcer.vcproj 319
SampleEnforcerDlg.cpp 320
SampleEnforcerDlg.h 320
saving reports 228
ScheduledSyncInterval
 for AD enrollments 50, 59
 for Azure AD enrollments 52
 for portal enrollments 92

ScheduledSyncTime
 for AD enrollments 50, 59
 for Azure AD enrollments 52
 for portal enrollments 92

selectivefilter.sample.properties file 57

Share With option 228

shared keys
 for SharePoint servers 95

SharePoint
 certificates for 95
 enrolling groups 91

Show field 217

site.sample.default.def (definition file) 54

sites
 description of 89

smbDirMapping utility 88

Sort By field 217

sp.sample.default.conn 92

sp.sample.default.conn (connection file) 48

sp.sample.default.def (definition file) 54, 54

SQL scripts 129

starting
 policy enforcers 112
 Policy Server 350

status
 of policy enforcers
 determining 114

Status Summary 114

Stdafx.cpp 320

Stdafx.h 320

stopping
 policy enforcers 111
 Policy Server 350

Submit 195

subpolicies
 about 184
 creating 184

synchronizing data tables
 configuring 126

synchronizing enrollments
 automatically 48, 77
 failure notifications 73
 from multiple domains 77
 manually 73
 vs. updating 67

T

Tags
 creating 237, 240
tamper prevention
enforcers
 setting password 117

targeted audits 120

ThreadPoolMaximumSize 145

time
 context in policies 175
 time zones 175

time zones 220

TLS 1.0
 required with Firefox 209

To Field 217

To field (notification email)
 configuring 140

truncating archive tables
 automated feature 130

trusted domains
 configuring 139, 140

type
 of policy enforcers 114

U

unarchiving data tables 129

uninstalling
 Policy Enforcers 113

updating enrollments
 from LDAP directory 69
 from LDIF file 70
 vs. synchronizing 67

URL 117

user components
 default properties 99
 defining custom properties 101, 101

User Criteria 217

User Criteria field 222

User field 221

User Principal Name 99

user repository
 config settings 137

useSSL (config setting) 137, 138

utilities
 AppDiscovery 83
 DBInit 133
 Import Locations 89
 Keytool 95
 Samba Directory Mapping 88

V

Vacuum DB utility 132

verbose option (Enrollment Manager) 45

version
 of enrolled applications 85
 of policies
 using 205

W

wildcard

_ (underscore) [222](#)

* (asterisk)

 in advanced conditions [182](#)

% (percent symbol) [222](#)

X

XACML [289](#)

Z

ZIP archives

 contents of [322](#)

