IBM Security Privileged Identity Manager
Version 2.0.2

Reference Guide

IBM Security Privileged Identity Manager
Version 2.0.2

Reference Guide

Note
Before using this information and the product it supports, read the information in Notices.

Edition notice

Note: This edition applies to Version 2.0.2 of *IBM Security Privileged Identity Manager* **(product number 5725-H30) and to all subsequent releases and modifications until otherwise indicated in new editions.**

Contents

Figures vii

Tables ix

Chapter 1. Policies for ESSO Agent . . . 1

Chapter 2. Web services API 5

Chapter 3. Shared access JavaScript APIs 9

Chapter 4. AccessAgent shared access APIs 11

CheckOut 11
CheckIn 13

Chapter 5. JavaScript object reference 15

How to read the reference pages 15
Activity. 17
Activity.auditEvent() 18
Activity.description 19
Activity.duedate 19
Activity.getSubProcesses() 19
Activity.guid 20
Activity.id 20
Activity.index. 20
Activity.name. 21
Activity.participant 21
Activity.resultDetail. 21
Activity.resultSummary 21
Activity.setResult() 22
Activity.started 22
Activity.state 22
Activity.subtype 23
Activity.type 23
AttributeChangeOperation 24
AttributeChangeOperation.attr 24
AttributeChangeOperation.op 24
AttributeChangeOperation.values[] 24
ContainerSearch 25
ContainerSearch.searchByFilter() 25
ContainerSearch.searchByURI() 26
Context. 26
Context.getAccountParameter() 28
Context.getActivityResult() 28
Context.getActivityResultById(). 28
Context.getLoopCount() 29
Context.getLoopCountByID() 29
Context.getProcessType() 30
Context.getRequestee() 30
Context.getService() 30
Context.isAccountDataChanged() 30
Credential 31
Credential.getAccessMode() 32
Credential.getCheckoutDuration() 32
Credential.getNotifyOption() 33
Credential.getNotificationRecipient() 33
Credential.isCheckoutSearchEnable() 33
Credential.isNotifyOnly() 34
Credential.isPasswordViewable() 34
Credential.isResetPasswordAtCheckin() 34
Delegate 35
DirectoryObject 36
DirectoryObject.addProperty() 36
DirectoryObject.dn 37
DirectoryObject.getChanges() 38
DirectoryObject.getProperty() 38
DirectoryObject.getPropertyAsDate() 39
DirectoryObject.getPropertyAsString() 40
DirectoryObject.getPropertyNames() 40
DirectoryObject.name 40
DirectoryObject.profileName. 41
DirectoryObject.removeProperty(name) 41
DirectoryObject.removeProperty(name,value) . . 41
DirectoryObject.setProperty() 42
EmailContext 43
Enrole 45
Enrole.generatePassword() 46
Enrole.getAttributeValue() 46
Enrole.getAttributeValues() 47
Enrole.localize() 47
Enrole.log() 48
Enrole.logError(). 48
Enrole.logInfo() 49
Enrole.logWarning() 50
Enrole.toGeneralizedTime() 50
Enrole.toMilliseconds() 50
Enrole.traceMax() 51
Enrole.traceMid() 51
Enrole.traceMin() 52
Error 53
Error.setMessage() 53
Error.getMessage() 54
Error.setErrorCode() 54
Error.getErrorCode() 54
Participant. 54
Participant.implementation 55
Participant.name. 56
Participant.type 56
ParticipantType 56
Person 58
Person.getAllAssignmentAttributes() 59
Person.getRoleAssignmentData() 60
Person.getRoleAssignmentData(String roleAssignedDN) 60
Person.getRoles() 61
Person.getNewRoles() 61
Person.getRemovedRoles() 62
Person.isInRole(). 62
Person.removeRole() 63

Person.removeRoleAssignmentData() 63
Person.updateRoleAssignmentData() 64
PersonSearch 64
PersonSearch.searchByFilter() 65
PersonSearch.searchByURI() 65
PostOffice 66
PostOffice.getAllEmailMessages() 66
PostOffice.getEmailAddress() 67
PostOffice.getPersonByEmailAddress() 67
PostOffice.getTopic() 67
Process 68
Process.auditEvent() 70
Process.comment 70
Process.description 70
Process.getActivity() 70
Process.getParent() 71
Process.getRootProcess() 71
Process.getRootRequesterName() 72
Process.guid 72
Process.getSubProcesses() 72
Process.id 73
Process.name 73
Process.parentId 73
Process.requesteeDN 73
Process.requestorDN 74
Process.requesteeName 74
Process.requestorName 74
Process.requestorType 75
Process.resultDetail 75
Process.resultSummary 75
Process.setRequesteeData() 75
Process.setResult() 76
Process.setSubjectData() 76
Process.started 77
Process.state 77
Process.subject 77
Process.type 78
ProcessData 78
ProcessData.get() 78
ProcessData.set() 79
Reminder 79
Role 80
Role.getAssignmentAttributes() 81
Role.getAllAssignmentAttributes() 81
Role.getOwner() 82
Role.setAssignmentAttributes() 82
RoleAssignmentAttribute 82
RoleAssignmentAttribute.getName() 83
RoleAssignmentAttribute.getRoleName() . . . 83
RoleAssignmentAttribute.getRoleDN 84
RoleAssignmentObject 85
RoleAssignmentObject.getAssignedRoleDN() . . 86
RoleAssignmentObject.getDefinedRoleDN() . . . 86
RoleAssignmentObject.addProperty() 86
RoleAssignmentObject.getChanges() 87
RoleAssignmentObject.getProperty() 88
RoleAssignmentObject.getPropertyNames() . . . 88
RoleAssignmentObject.removeProperty() . . . 88
RoleAssignmentObject.setProperty() 89
RoleSearch 89
RoleSearch.searchByName() 90
RoleSearch.searchByURI() 90
Service 91
ServiceSearch 91
ServiceSearch.searchByFilter() 91
ServiceSearch.searchByName() 92
ServiceSearch.searchByURI() 93
ServiceSearch.searchForClosestToPerson() . . . 93

Chapter 6. Application identity commands 95
install-certificate 95
register-first-instance 96
register-additional-instance 98
get-credential 100
configure-services 101
register-service-manager 102
discover-services 103

Chapter 7. Virtual appliance commands 105
Current mode commands for the virtual appliance 105
Global commands 107
Cleaning core dump files 109
Enabling trace for the virtual appliance services 112

Chapter 8. Dynamic tags in mail templates 115
Mail templates 120
Generic workflow messages 120
Organization management default messages . . . 125
Shared or Application identity management default messages 129

Chapter 9. Sample virtual appliance configuration response file 133

Chapter 10. Schema reference 135
Auditing schema tables 135
AUDIT_EVENT table 135
IBM Security Privileged Identity Manager authentication 136
Person management 137
Delegate authority 139
Policy management 140
ACI management 143
Access request management 145
Manual activity events 149
Lifecycle rule events 157
Account management 158
Database views 161
Container management 165
Organization role management 166
Group management 168
Service management 170
Reconciliation 171
Entitlement workflow management 172
Entity operation management 173
System configuration 174
Runtime events 176

Self-password change 176
Credential management 177
Credential Pool management 178
Credential Lease management 179
Shared Access Policy management 182
Application ID management 183

Chapter 11. Scenarios 187
Creating an access request workflow 187
Privileged user: Logging on to a managed resource 188
Logging on with PuTTY or SecureCRT 188
Logging on with the Microsoft Remote Desktop Connection (RDP) client 189
Logging on with IBM Personal Communications 189
Logging on with SQL Server Management Studio 190
Logging on with DB2 Data Studio 191
Logging on with the VMware vSphere Client 192
Logging on with SoftLayer 192
More examples that can trigger check-out and check-in automation 193
Integration with IBM Security Identity Manager 194
Integration with SoftLayer 198
Integration with IBM QRadar Security Intelligence Platform 199
Configuring the log sources 200
Enabling `syslog` on the IBM Security Privileged Identity Manager virtual appliance 200

Notices 203

Index 209

Figures

Tables

1. JavaScript objects 15
2. Syntax and example of using JavaScript code to replace message content. 115
3. Syntax and examples of using a RE tag to replace message content. 116
4. Syntax and example of using tags to replace message content. 117
5. Syntax and examples of ITIMURL. 117
6. Escape characters 118
7. Auditing schema tables 135
8. AUDIT_EVENT table 135
9. Column values in the AUDIT_EVENT table 136
10. AUDIT_MGMT_TARGET table 137
11. Values for columns in the AUDIT_EVENT table 137
12. AUDIT_MGMT_DELEGATE table 139
13. Values for columns in the AUDIT_EVENT table 140
14. Values for columns in the AUDIT_EVENT table 141
15. AUDIT_MGMT_TARGET table 144
16. Values for columns in the AUDIT_EVENT table 144
17. **AUDIT_MGMT_ACCESS_REQUEST** table for access request management 145
18. **AccessRequest** values for the **AUDIT_MGMT_OBLIGATION** table 147
19. **AccessRequest** values for the **AUDIT_MGMT_OBLIGATION_ATTRIB** table. . . . 147
20. **AccessRequest** values for the **AUDIT_MGMT_OBLIGATION_RESOURCE** table . . . 148
21. **AUDIT_MGMT_MESSAGE** table for access request management. 148
22. AUDIT_EVENT table for access request management. 148
23. Create manual activity values for the **AUDIT_MGMT_ACTIVITY** table 150
24. Create manual activity values for the **AUDIT_MGMT_PARTICIPANT** table 153
25. AUDIT_EVENT table for the create manual activity event 154
26. Escalate manual activity values for the **AUDIT_MGMT_ACTIVITY** table 155
27. Escalate manual activity values for the **AUDIT_MGMT_ACTIVITY** table 155
28. Escalate manual activity event values for the **AUDIT_MGMT_PARTICIPANT** table 156
29. AUDIT_EVENT table for the escalate manual activity event 157
30. AUDIT_EVENT table for lifecycle rule events 157
31. AUDIT_MGMT_PROVISIONING table 158
32. Values for columns in the AUDIT_EVENT table 158
33. V_PIM_CICO_HISTORY view 161
34. Resource property mapping. 161
35. V_PIM_CICO_HISTORY_DB_RSRC view 162
36. V_PIM_CRED_INFO view 162
37. V_PIM_CRED_INFO_DB_RSRC view 163
38. V_PIM_CRED_DETAILS view 164
39. V_PIM_CRED_DETAILS_DB_RSRC view 165
40. Values for columns in the AUDIT_EVENT table 166
41. AUDIT_MGMT_TARGET table 167
42. Values for columns in the AUDIT_EVENT table 167
43. AUDIT_MGMT_TARGET table 168
44. Values for columns in the AUDIT_EVENT table 169
45. AUDIT_MGNT_TARGET table 170
46. Values for columns in the AUDIT_EVENT table 171
47. Values for columns in the AUDIT_EVENT table 172
48. Values for columns in the AUDIT_EVENT table 172
49. Values for columns in the AUDIT_EVENT table 173
50. Values for columns in the AUDIT_EVENT table 174
51. Value of the entity_name column table 175
52. Values for columns in the AUDIT_EVENT table 176
53. Values for columns in the AUDIT_EVENT table 177
54. Values for columns in the AUDIT_EVENT table 177
55. Values for columns in the AUDIT_EVENT table 179
56. AUDIT_MGMT_LEASE table 180
57. AUDIT_MGMT_TARGET table 180
58. Values for columns in the AUDIT_EVENT table 181
59. Values for columns in the AUDIT_EVENT table 182
60. Values for columns in the AUDIT_EVENT table 183
61. Example audit log entry when an application instance is registered successfully 183
62. Example audit log entry when an application instance retrieves a credential successfully . . 184
63. Example audit entry when an application instance is unable to retrieve a credential due to an invalid token. 184
64. Example audit entry when an application instance is unable to retrieve a credential because the application fingerprint does not match 184
65. More events that can trigger automated check-out or check-in behavior. 193
66. Types of users to be on-boarded 195
67. Create an ISPIM administrative domain and ISPIM account 195
68. Create a shared access policy 195
69. Assign Privileged Administrator as access owner 196
70. Check out the administrative account 196
71. Grant access to the Pinnacle Server 197

Chapter 1. Policies for ESSO Agent

Know the privileged identity management policies, where to find the policies, their descriptions, and their default values.

Use privileged identity management policies to configure the authentication service, check-in and check-out, and session recording options.

pid_pim_cico_svc_url: IBM® Security Privileged Identity Manager URL

IMS Entry	IBM Security Privileged Identity Manager URL
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Service URL for IBM Security Privileged Identity Manager shared access service. For example: `https://hostname/itim/services/WSSharedAccessService`
Registry	
Type	String
Values	
Scope	System
Note	Refreshed on sync.

pid_pim_itim_auth_svc_id: IBM Security Privileged Identity Manager Authentication Service ID

IMS Entry	IBM Security Privileged Identity Manager Authentication Service ID
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Authentication service ID for identifying the IBM Security Privileged Identity Manager credential that is stored in the user's wallet.
Registry	
Type	String
Values	
Scope	System
Note	Refreshed on sync.

pid_recorder_enabled: Enable session recording?

IMS Entry	Enable session recording?
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Whether to record sessions created using shared access identities.
Registry	
Type	Boolean

pid_recorder_enabled: Enable session recording?

Values	• No • Yes (default, if session recording feature is activated)
Scope	System
Note	Refreshed on sync.

pid_recorder_server: Privileged Session Recorder Server URL

IMS Entry	Privileged Session Recorder Server URL
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	URL of the Privileged Session Recorder Server. For example: `https://hostname/recorder/collector`
Registry	
Type	
Values	String
Scope	System
Note	Refreshed on sync.

pid_recorder_image_capture_option: Session recording image capture option

IMS Entry	Session recording image capture option
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Color depth of the images to capture for session recording.
Registry	
Type	Non-negative integer
Values	• #0: Full Color • #1: Grayscale (default value)
Scope	System
Note	Refreshed on sync.

pid_recorder_keyboard_capture_option: Session recording keyboard capture option

IMS Entry	Session recording keyboard capture option
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Specifies keyboard events to capture during a screen-based session recording.
Registry	
Type	Non-negative integer

pid_recorder_keyboard_capture_option: Session recording keyboard capture option

Values	• #0: Log all key inputs (default value) • #1: Log only special keys • #2: Do not log any key inputs
Scope	System
Note	Refreshed on sync. This option does not apply to text-based recordings.

pid_collector_comm_fail_action: Action to take when the client computer cannot connect to the server

IMS Entry	Action to take when the client computer cannot connect to the server
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Specifies the action to take on the client computer when the Privileged Session Recorder Server is not reachable after a maximum number of retries.
Registry	
Type	Non-negative integer
Values	• #0: Take no action. (default) • #1: Block user inputs. • #2: Close application.
Scope	System
Note	Refreshed on sync.

pid_collector_comm_fail_limit: Maximum number of failed connection attempts to the server

IMS Entry	Maximum number of failed connection attempts to the server
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Specifies the maximum number of connection attempts before determining that the Privileged Session Recorder Server is not reachable.
Registry	
Type	Non-negative integer
Values	3 (default value)
Scope	System
Note	Refreshed on sync.

pid_collector_comm_fail_retry_interval: Time interval between connection retries to the server

IMS Entry	Time interval between connection retries to the server
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Time interval, in seconds, between attempts to connect to the Privileged Session Recorder Server.
Registry	
Type	Non-negative integer
Values	15 (default value)
Scope	System
Note	Refreshed on sync.

pid_pim_optional_justification

IMS Entry	Allows empty justification on credential check out.
Location	**AccessAdmin** > **System** > **System policies** > **IBM Security Privileged Identity Manager Configuration Policies**
Description	Allows privileged users to check out credentials without entering a justification.
Type	Boolean
Values	• Yes • No (default)
Scope	System
Note	Refreshed on sync.

Chapter 2. Web services API

This API consists of multiple web services, which are grouped by function. The services are listed alphabetically except the `WSSessionService`. This service is listed first since it is the first service that is called by any application. The session object that is returned by its login method is used as a parameter in all subsequent services.

For more information about each web service that you can use to extend the IBM Security Privileged Identity Manager solution, see the web services technote (http://www-01.ibm.com/support/docview.wss?uid=swg21691862).

WSSessionService

The `WSSessionService` web service provides authentication, session creation, and password challenge authentication. A client calls `WSSessionService` before you start any other web services. `WSSessionService` returns a session (handle) object that must be passed to the other web service calls to maintain a threaded conversation. The service provides the following operations:

- Login.
- Logout.

You can also use the `WSUnauthService` web service for other operations.

WSAccessService

The `WSAccessService` web service provides the following operations:

- Create a user access.
- Retrieve existing user access of a person.
- Remove user access.
- Search access entitlements available to a person.

The service provides following operations:

- Create and modify accesses.
- Do access searches.

WSOrganizationalContainerService

The `WSOrganizationalContainerService` web service provides IBM Security Privileged Identity Manager organization tree traversal and retrieval methods.

WSPasswordService

The `WSPasswordService` web service provides password management functions. The service provides the following operations:

- Validates the password as per the password policy rules.
- Enables change or generate password.

WSPersonService

The `WSPersonService` web service provides person-object related methods. The service provides the following operations:

- Create, modify, suspend, restore, delete, and other simple person operations.
- Retrieve the services to which a person is entitled in IBM Security Privileged Identity Manager or accounts.
- Do person searches.
- Retrieve the person object of the Principal.

WSRequestService

The `WSRequestService` web service provides the IBM Security Privileged Identity Manager request related functions. The service provides the following operations:

- Search for completed requests.
- Retrieve pending requests.
- Retrieve the request object that is based on the process ID or request ID.

WSRoleService

The `WSRoleService` web service provides role-based capabilities in the IBM Security Privileged Identity Manager. The service provides the following operations:

- Create and modify roles.
- Do role searches.
- Manage role hierarchy.

WSSearchDataService

The `WSSearchDataService` web service provides functions to search various IBM Security Privileged Identity Manager directory objects. The search method does not enforce the IBM Security Privileged Identity Manager ACIs, but a valid IBM Security Privileged Identity Manager session is required to call these methods. The service provides the following operations:

- Search for persons from `root` container.
- Search for persons that are having an IBM Security Privileged Identity Manager account.
- Search for the possible delegates within IBM Security Privileged Identity Manager for the logged-in user.
- Retrieve the searchable attributes of an entity in IBM Security Privileged Identity Manager.
- Retrieve common searchable attributes for the IBM Security Privileged Identity Manager entity.

WSServiceService

The `WSServiceService` web service provides IBM Security Privileged Identity Manager-based managed services (end-point configuration) functions. The service provides the following operations:

- Retrieve support data. For example, group data for UNIX, Linux, or Microsoft Windows services.
- Determine whether a password is required when provisioning on a service.

- Retrieve services that are configured on IBM Security Privileged Identity Manager.

WSSharedAccessService

The `WSSharedAccessService` web service provides many functions for shared access. The web service clients must call the login method before it calls any other web services. The service provides the following operations:

- Retrieve authorized shared accesses.
- Retrieve the credentials.
- Check in or checkout credentials.

WSSystemUserService

The `WSSystemUserService` web service provides the functions that are related to system users. The service provides the following operations:

- Manage delegates, that is, add, modify, or delete delegates.
- Retrieve all the system roles.
- Configure challenge response.
- Search for system users who have an IBM Security Privileged Identity Manager account.

WSToDoService

The `WSToDoService` web service provides the functions to manage the different activities available in IBM Security Privileged Identity Manager. The service provides the following operations:

- Approve or reject activities.
- Retrieve or Submit Request for information activity details.
- Retrieve the pending activities of the logged-in user.

WSUnauthService

The `WSUnauthService` web service provides an interface for all the web service APIs that do not require the IBM Security Privileged Identity Manager authentication. The service provides the following operations:

- Version information.
- Reset password by using the challenge responses.
- Password policies.

Related information:

IBM Security Privileged Identity Manager web services
This document describes the IBM Security Privileged Identity Manager web services that can be used to extend the IBM Security Privileged Identity Manager solution.

Chapter 3. Shared access JavaScript APIs

Shared access provides the `CredentialModelExtension` API to support access to a Credential object in various shared access-related workflow operations.

You can use the access to customize operations, including `addCredentialToVault`, `checkin`, and `checkout`.

For more information, see "Credential" on page 31.

Chapter 4. AccessAgent shared access APIs

Use the AccessAgent shared access APIs to extend single sign-on automation support for the check-in and check-out of shared credentials with AccessProfiles.

CheckOut

Use `CheckOut` method to check out a credentials with the `PIMSlnHelper` DLL. The check-out widget is an example of an implementation that uses the `CheckOut` method.

```
HRESULT CheckOut(
    [in] ISERuntime* RuntimeObj,
    [in] BSTR ItimSvcUrl,
    [in] BSTR ItimAuthSvcId,
    [in] BSTR PrivCredBag,
    [in] VARIANT_BOOL IsPrivCredBagLocal,
    [in] BSTR ApplicationName,
    [in] VARIANT_BOOL ServiceLowerCaseConventionEnabled,
    [in] VARIANT_BOOL ReAuthPasscodeEnabled,
    [in] VARIANT_BOOL CheckInAllBeforeCheckOutEnabled,
    [in] BSTR RoleSelectionDlgParentHwndSignature,
    [in] VARIANT_BOOL SilentModeEnabled,
    [in, defaultvalue("true")] VARIANT_BOOL IsRegistrationEnabled,
[in, defaultvalue("")] BSTR RecordingId,
                       [out, retval] int* pRet);

   [out,retval] int* pRet);
```

Parameters

RuntimeObj
Run time object obtained from the scripting host.

ItimSvcUrl
URL of the IBM Security Identity Manager service. For example: `https://`*`pimva.example.com`*`/itim/services/WSSharedAccessService`.

ItimAuthSvcId
Authentication service ID of IBM Security Identity Manager. The user Wallet must contain the IBM Security Identity Manager credential.

PrivCredBag
Privileged credential bag stores:

- Checked-out privileged credentials.
- Application managed resource authentication service ID.

IsPrivCredBagLocal
Specify whether to use local bag for the privileged credential bag.

ItimTokenBag
This parameter is not used. It is included for compatibility with an earlier version.

IsItimTokenBagLocal
Specify whether to use local bag for IBM Security Identity Manager token bag.

CheckInAllBeforeCheckOutEnabled
Specify whether to reauthenticate user credentials before you check out.

ReAuthPasscodeEnabled
Specify whether to check in all credentials before checkout.

RoleSelectionDlgParentHwndSignature
Signature of the role selection dialog box parent window. If the parameter is an empty string, the role selection dialog box parent window is NULL.

SilentModeEnabled
If this parameter is true, no dialogs and prompts are displayed.

IsRegistrationEnabled
If this parameter is true, the background process automatically checks in the shared credential. It occurs when the process fails to check in the credential, for example, a user exits the program in an unexpected way.

BSTR RecordingId
Specifies the Privileged Session Recorder console recording ID.

Example VBScript action

```
' Perform CheckOut

Dim cicomgr
Dim pc
Dim result
Dim reauth_needed

Const PrivCredBag = "CICO_injection_bag"

Set pc = runtime.getPropertiesContainer()
RoleSelectionDlgParentHwndSignature = pc.GetPropValue("RoleSelectionDlgParentHwndSignature")
isim_url = pc.GetPropValue("WSPATH")
isim_auth_service = pc.GetPropValue("isim_auth_service")

ApplicationName = pc.GetAccDataItem(PrivCredBag, "ApplicationName")
RecordingId = pc.GetPropValue("recording_session_guid")

reauth_needed_str = pc.GetPropValue("reauth_needed")
if reauth_needed_str = "0" then
   reauth_needed = False
else
   reauth_needed = True
end if
Set cicomgr = CreateObject("PIMSlnHelper.CICOMgr")
result = cicomgr.CheckOut(runtime _
      , isim_url _
      , isim_auth_service _
      , PrivCredBag  _
      , true  _
      , ApplicationName _
      , true _
      , reauth_needed _
      , false _
      , RoleSelectionDlgParentHwndSignature _
      , false _
      , True _
      , RecordingId _
      )

' save the result value so that we can show the error description
pc.SetPropValue "checkout_result", result

if result = 0 then

      ' succesful checkout
      pc.SetPropValue "checkout_done","1"
      pc.SetAccDataItem PrivCredBag, "checkout_done", "1"

elseif result = 5 then

      ' if ISIM credentials not found in wallet
      pc.SetPropValue "checkout_done", "-1"
      pc.SetAccDataItem PrivCredBag, "checkout_done", "-1"

else
```

```
      ' default error handling
      pc.SetPropValue "checkout_done", "0"
      pc.SetAccDataItem PrivCredBag, "checkout_done", "0"
end if
```

CheckIn

Use `CheckIn` to check in shared access credentials into the credential vault.

```
HRESULT CheckIn(
[in] ISERuntime* RuntimeObj,
[in] BSTR PrivCredBag,
[in] VARIANT_BOOL IsPrivCredBagLocal,
[out,retval] int* pRet);
```

Parameters

RuntimeObj
Runtime object obtained from the scripting host.

PrivCredBag
Privileged credential bag that contains the checked out credentials. It stores the:

- Privileged Application (For example: PuTTY, PCOMM, RDP or vSphere)
- Authentication service ID
- Service URI or managed service endpoint
- LeaseDN
- Expiration Time
- Selected RoleID

IsPrivCredBagLocal
Boolean. Specify whether the bag is local or global.

Example

Create a `PIMSlnHelper` object and to check in the credentials.

```
Set cicomgr = CreateObject("PIMSlnHelper.CICOMgr")
result = cicomgr.CheckIn(runtime , PrivCredBag , true)
```

Chapter 5. JavaScript object reference

There are a number of IBM Security Privileged Identity Manager specific objects available for use.

The reference section is arranged alphabetically.

Table 1. JavaScript objects

Script Object	Object Name	Object Type
AttributesExtension (deprecated)	**ATTRIBUTES**	**Map**
EmailContextExtension	**EmailContext**	**EmailContext**
EnroleExtension	**Enrole** **error**	**Enrole** **Error**
LoopCountExtension	**loopcount**	**int**
PersonPlacementRulesExtension	**entry**	**Map**
PostOfficeExtension	**PostOffice**	**PostOffice**
AccountModelExtension	**Account constructor** **AccountSearch constructor**	**Account** **AccountSearch**
CredentialModelExtension	**Credential**	**Credential**
OrganizationModelExtension	**ContainerSearch constructor**	**ContainerSearch**
PersonModelExtension	**Person constructor** **ExtendedPerson constructor** **PersonSearch constructor**	**Person** **ExtendedPerson** **PersonSearch**
RoleModelExtension	**Role constructor** **RoleSearch constructor**	**Role** **RoleSearch**
ServiceModelExtension	**Service constructor** **ServiceSearch**	**Service** **ServiceSearch**
ReminderExtension	**reminderCtx**	**Reminder**
ServiceExtension	**service**	**DirectoryObject**
SubjectExtension	**subject**	**Person** **Note:** For Orphan Adoption Rule JavaScript, the subject is a Map, which contains the account attributes returned from reconciliation. The entries in the map are referred by the name of the account attributes, which might vary based on the service type.
WorkflowExtension	**process** **activity** **Participant constructor** **ParticipantType** **$RelevantDataName**	**Activity** **Participant** **ParticipantType** **ProcessData**Process

How to read the reference pages

This section explains the structure of each reference item.

Title and Description

Every reference entry begins with a title and a one line description. The entries are alphabetized by title. The one-line description gives a quick summary of the item documented in the entry.

Availability

The IBM Security Privileged Identity Manager JavaScript extensions change over time. Unless otherwise noted, anything available in one version of the IBM Security Privileged Identity Manager extensions is also available in later versions. This section also specifies whether an existing item was enhanced with a later version of the extensions and when an item is deprecated. Deprecated items are no longer supported and can be removed from future versions of the IBM Security Privileged Identity Manager extensions. Do not use deprecated items in new IBM Security Privileged Identity Manager JavaScript code.

Provided by

At installation, IBM Security Privileged Identity Manager provides this initial set of registered extensions:

- **`EnroleExtension`**
- **`PostOfficeExtension`**
- **`PersonPlacementRulesExtension`**
- **`WorkflowExtension`**
- **`ReminderExtension`**
- **`ServiceExtension`**
- **`SubjectExtension`**
- **`AttributesExtension`**
- **`LoopCountExtension`**
- **`EmailContextExtension`**
- **`Model extensions package`**

Inherits From

JavaScript classes can inherit properties and methods from other classes. When it occurs, an Inherits From section appears in the reference entry. The inherited fields and methods are in the listed superclasses. For example, the subject object inherits all of its fields and properties from the **`DirectoryObject`** class.

Synopsis

This section is a synopsis of how to use the object, method, property, or function.

Arguments

If the reference page describes a function or method that has arguments, the Synopsis is followed by an Arguments subsection that describes the arguments to the function or method. For some objects, the Synopsis section is replaced by a Constructor section which is also followed by an Arguments subsection.

Returns

If a function or a method has a return value, the Arguments subsection is followed by a Returns subsection that explains the return value of the function, method or constructor.

Properties

If the reference page documents an object, the Properties section lists the properties the object supports and provides short explanations of each.

Methods
The reference page for an object that defines methods includes a Methods section.

Description
Most reference entities contain a Description section, which is a basic description of whatever is documented. For some simple methods, the Arguments and Returns sections document the method sufficiently by themselves, so the Description section is omitted.

Usage This section describes common techniques for using the item, or it contains cautionary information.

Activity

Activity is used to reference any activity in a IBM Security Privileged Identity Manager workflow.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
The activity JavaScript object in the `WorkflowExtension` returns an Activity object that represents the current workflow activity. The workflow activity can be used in the context of a workflow activity PostScript, or in a transition script, to reference the current activity. For a transition script, this object represents the activity whose completion has lead to the evaluation of the transition script.

`Process.getActivity()` can return any Activity object in the context of a workflow process. For more information, see the description of this method.

Activity Result Summary Code

APPROVED
Approved process summary code. Result code is `AA`.

ESCALATED
Escalated process summary code. Result code is `ES`.

FAILED
Failed process summary code. Result code is `SF`.

PARTICIPANT_RESOLVE_FAILED
Participant resolved failure process summary code. Result code is `PF`.

PENDING
Pending process summary code. Result code is `PE`.

REJECTED
Rejected process summary code. Result code is `AR`.

SUBMITTED
Submitted process summary code. Result code is `RS`.

SUCCESS
Success process summary code. Result code is `SS`.

TIMEOUT
Time out process summary code. Result code is `ST`.

WARNING
Warning process summary code. Result code is `SW`.

Properties

description
Describes the purpose of the activity given when defined in the workflow designer.

duedate
Indicates the time in milliseconds by when the activity is due.

id Assigned by the workflow designer to uniquely identify the workflow activity within the workflow engine.

index Index of the instance of the activity.

name Label given this activity when defined in the workflow designer.

participant
The activity participant, as defined in the workflow designer.

resultDetail
An application-specific string that provides more detail about the result of the activity.

resultSummary
An application-specific string that represents the summary result of the activity.

started
Indicates when the activity started.

state Code that represents the current state of the activity.

subtype
Code that further categorizes the activity beyond the type of the activity, such as approval or request for information.

type Code that categorizes the activity given when defined in the workflow designer, such as manual or application.

Methods

auditEvent()
Create an event in the audit trail specific to the activity.

setResult()
Change the result member of the activity in the current activity.

Description
This entity represents the current workflow activity that is being run. Within the context of a workflow transition script, this entity represents the activity whose completion has lead to the evaluation of the transition script. No constructor is available to create this object in any IBM Security Privileged Identity Manager context.

Activity.auditEvent()

The method creates an event in the audit trail.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity`.auditEvent(event)`

Arguments

event String representing the event to be audited.

Description

This method creates an event in the audit trail specific to the activity. The function takes in one parameter that can be any JavaScript object that can be translated into a String for storage. In the audit trail, the event is automatically time stamped.

Usage

```
activity.auditEvent("Task completed");
```

Activity.description

The field provides information about the purpose of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

activity.description

Description

This read-only field is a String that describes the purpose of the activity given when defined in the workflow designer.

Usage

```
x = activity.description;
```

Activity.duedate

The field represents the time in milliseconds by when the activity is due.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

activity.duedate

Description

This read-only field is a long number of milliseconds by when this activity is due.

Usage

```
x = activity.duedate;
```

Activity.getSubProcesses()

The method returns the subordinate processes (if any) of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
activity.getSubProcesses()
```

Returns

The subordinate processes. If there are no subordinate processes, an empty array is returned.

Description

This method returns the subordinate processes (if any) of this activity.

Usage

```
var out = "subprocesses of the activity: \n";

var subProcesses = activity.getSubProcesses();
for (var i = 0; i < subProcesses.length; i++) {
  out += subProcesses[i].id + " type: " + subProcesses[i].type + " resultSummary: " +
  subProcesses[i].resultSummary + "\n";
}

activity.auditEvent(out);
```

Activity.guid

The generated unique identifier assigned to the activity at runtime.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.guid

Description
This read-only field is a String of the generated unique identifier for the workflow activity within the workflow engine.

Usage `x = activity.guid;`

Activity.id

The field is the unique identifier assigned to the activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.id

Description
This read-only field is a String assigned by the workflow designer to uniquely identify the workflow activity within the workflow engine.

Usage `x = activity.id;`

Activity.index

The field is an index of the instance of the activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.index

Description
This field is a read-only and a number. If there is more than one instance of this activity, such as in the case where the activity of the ID is called multiple times in a loop in the workflow process, the value starts at one. If there is only one instance of this activity, the index value is zero.

Usage `x = activity.index;`

Activity.name

The field is the label that is assigned to the activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.name

Description
This read-only field is a String assigned by the workflow designer to label this activity.

Usage `x = activity.name;`

Activity.participant

The field represents the activity participant.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.participant

Description
This read-only field is a Participant that represents the activity participant. Not all activities have a participant. If there is no participant associated with the activity, this member is empty.

Usage `x = activity.participant;`

Activity.resultDetail

You can get the details about the result of the activity with this field.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.resultDetail

Description
This read-only field is an application-specific string that provides more detail about the result of the activity.

Usage `x = activity.resultDetail;`

Activity.resultSummary

The field helps you view the summary of the result of the activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
activity.resultSummary

Summary
This read-only field is an application-specific string that provides a summary of the result of the activity. It can represent a success or failure.

Usage `x = activity.resultSummary;`

Activity.setResult()

The method changes the result member of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
activity.setResult(summary)
activity.setResult(summary, detail)
```

Arguments

summary
String code that represents the result summary.

detail String representing the result details.

Description

This method changes the result member of the activity in the current activity. It is supported for current activities in the current workflow process. The result is composed by an application-specific summary code, and optional more detailed application-specific description. The summary code can indicate a success or failure. This summary code is stored as the `resultSummary` member locally and updated in the relevant data in the workflow engine. The detail is stored as the `resultDetail` member locally and updated in the relevant data in the workflow engine.

Usage

```
activity.setResult(activity.FAILED);
activity.setResult(activity.FAILED, "Unable to connect to resource");
```

Activity.started

The field represents the date that indicates when the activity started.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
activity.started
```

Description

This read-only field is a string that represents the date that indicates when the activity started.

Usage

```
x = activity.started;
```

Activity.state

The field represents the current state of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
activity.state
```

Description

This read-only field is a code string that represents the current state of the activity. The state can have the following values:

- **R** for running

- **I** for not started
- **T** for terminated
- **A** for aborted
- **S** for suspended
- **C** for completed
- **B** for bypassed

Usage

```
if (activity.state == "S") {
  ...
}
```

Activity.subtype

The field represents the subtype of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

activity.subtype

Description

This read-only field is a code string that further categorizes the activity beyond the type of the activity, such as approval or request for information. This is defined in the workflow designer. Not all activities have a subtype. If there is no subtype associated with the activity, this member is empty. The currently supported subtypes are:

- **AP** for approval
- **RI** for request for input
- **WO** for work order

Usage

```
x = activity.subtype;
```

Activity.type

The field represents the type of the activity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

activity.type

Description

This read-only field is code string that categorizes the activity given when defined in the workflow designer, such as manual or application. The currently supported types are:

- **S** for subprocess
- **L** for loop
- **A** for application
- **R** for route
- **M** for manual
- **O** for operation

Usage

```
x = activity.type;
```

AttributeChangeOperation

The object represents an entity about the attribute change operation.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`AttributeChangeOperation` objects are returned from the method **`DirectoryObject.getChanges()`** and are therefore not provided by any specific extension.

Properties

attr Name of the attribute that is being changed.

op An integer that identifies the type of change that is being made.

values[]
An array of objects that must be either added, removed, or replaced.

Description
This entity represents the changes made to a IBM Security Privileged Identity Manager object.

AttributeChangeOperation.attr

Represents the name of an attribute that is being changed.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
attributeChangeOperation.attr

Description
Value is the attribute that is being changed.

Usage
```
x = attributeChangeOperation.attr;
```

AttributeChangeOperation.op

The field represents the type of change that is being made.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
attributeChangeOperation.op

Description
This read-only field is a number that identifies the type of change that is being made. The values are:

- **1** for add
- **2** for replace
- **3** for remove

Usage
```
x = attributeChangeOperation.op;
```

AttributeChangeOperation.values[]

The field represents the name of attribute that is being changed.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
attributeChangeOperation.values[]

Description
This read-only field is an array of objects that must be added, removed, or replaced.

Usage
```
x = attributeChangeOperation.values[1];
```

ContainerSearch

The object represents the search for an organizational container.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
com.ibm.itim.script.extensions.model.OrganizationModelExtension

Constructor
new ContainerSearch()

Returns
The newly created and initialized container search object.

Methods

searchByFilter()
Search for a container with a filter.

searchByURI()
Search for an organizational container by URI within a parent organizational container.

Description
Implements the IBM Security Privileged Identity Manager **OrganizationalContainerSearch** class.

ContainerSearch.searchByFilter()

The method represents the search for a container with a filter.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
containerSearch.searchByFilter(*profileName*, *filter*, *scope*)

Arguments

profileName
The String name of the organizational container profile to use.

filter LDAP search filter String that defines the criteria for returned containers to meet. The filter must be in the format defined by RFC2254.

scope Optional Int search scope. Use 1 for One Level Scope and 2 for SubTree Scope. One Level Scope is the default scope.

Returns

An array of **DirectoryObjects** representing the results of the search.

Description

This method searches for a container with a filter.

Usage

```
var locationContainer = new ContainerSearch();
// use subtree scope
var thisLocation = locationContainer.searchByFilter("Location",
 "(l=Raleigh)", 2);

// use default one level scope
var otherLocation = locationContainer.searchByFilter("Location",
 "(l=Raleigh)");
```

ContainerSearch.searchByURI()

The method finds an organizational container by URI in a parent organizational container.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
ContainerSearch.searchByURI(containerDN, uri)
```

Arguments

Container DN
String representing the distinguished name of the parent organizational container.

uri String representing the URI of the organizational container.

Returns

A **DirectoryObject** representing the container.

Description

Given the distinguished name of the parent organizational container and the container URI, this method finds the container. If the container is not found, this function returns null. If more than one container is found, this function throws a scripting exception.

Usage

```
var container = (new ContainerSearch()).searchByURI(parentContainer.dn,
   uri);
if (container != null) {
Enrole.log("script", "Found " + container.getProperty("ou") );}
```

Context

The object represents the context of the currently running workflow process (for example, requestor or subject). Only used for entitlement workflows.

Note: This object type is deprecated. Use workflow JavaScript objects, such as **Process**, **Activity**, and **Relevant Data**.

Some account-specific functions of the context JavaScript extension, including **getService()**, **isAccountDataChanged()**, and **getAccountParameter()** cannot be applicable to operation workflows that are not account related. The context JavaScript extension is not suggested for custom workflows.

Availability

IBM Security Privileged Identity Manager 1.0

Provided by

`com.ibm.itim.workflow.script.WorkflowExtension`

Context Constants

APPROVED

This constant is used to describe the result of an activity. The member applies only to Approval types of activities.

Usage

```
if (context.getActivityResult() == context.APPROVED) {...
```

REJECTED

This constant is used to describe the result of an activity. This member applies only to Approval types of activities.

Usage

```
if (context.getActivityResult() == context.REJECTED) {...
```

NEWACCOUNT

This constant is used to identify the type of request that triggers the custom workflow run time.

Usage

```
if (context.getProcessType() ==
  context.NEWACCOUNT) {...
```

ACCOUNTDATACHANGE

This constant is used to identify the type of request that triggers the custom workflow in run time.

Usage

```
if (context.getProcessType() ==
  context.ACCOUNTDATACHANGE) {...
```

Methods

getAccountParameter()

Returns the value of an account attribute.

getActivityResult()

Returns the activity result for the current activity.

getActivityResultByID()

Returns the activity result for a specific activity.

getLoopCount()

Returns the loop count for the current loop activity.

getLoopCountByID()

Returns the current loop count for a specific loop activity.

getProcessType()

Returns the type of the request that triggers the custom workflow process.

getRequestee()

Returns the requestee associated with the request as a Person object.

getService()

Returns the target service as a Service entity object.

isAccountDataChanged()
Identifies whether a specific account attribute was changed in the request that triggers the custom workflow process.

Description
The context of the currently running workflow process (for example, requestor or subject) is represented within the JavaScript as an object named context.

Context.getAccountParameter()

The method returns the value of an account attribute.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getAccountParameter(`*`String attributeName`*`)`

Arguments

attributeName
String representing the attribute name.

Returns
String value of an account attribute.

Description
This member function returns the value of an account attribute as a string.

Usage
```
parameter=context.getAccountParameter("group");
```

Context.getActivityResult()

The method returns the activity result for the current activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getActivityResult()`

Returns
String

Description
This member function returns the activity result for the current activity. The function returns APPROVED or REJECTED. If this function is used to specify a transition condition, the function refers to the activity from which the transition is coming.

Usage
```
if (context.getActivityResult() == context.APPROVED) {...
```

Context.getActivityResultById()

The method returns the activity result for a specific activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getActivityResultById(`*`String activityDefinitionID`*`)`

Arguments

activityDefinitionID
String ID of the activity definition.

Returns
String

Description
This member function returns the activity result for a specific activity. The function returns APPROVED or REJECTED.

Usage
```
if (context.getActivityResultByID("1234567890") == context.APPROVED)
{...
```

Context.getLoopCount()

The method returns the loop count for the current loop activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
```
getLoopCount()
```

Returns
Integer of loop count.

Description
This member function returns the loop count for the current loop activity. If this function is called before a loop is started, the loop count is 0. If this activity is called while the loop activity is in process, the loop count is the number of times the loop ran. If this function is called after the loop is completed, the loop count is the total number of times the loop is defined to run.

Usage
```
currentiteration = context.getLoopCount();
```

Context.getLoopCountByID()

The method returns the current loop count for a specific loop activity.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
```
context.getLoopCountByID(String activityDefinitionID)
```

Arguments

activityDefinitionID
ID of the activity definition.

Returns
Integer

Description
This member function returns the current loop count for a specific loop activity. If this function is called before the loop is started, the loop count is 0. If this function is called while the loop activity is in process, the loop count is the number of times the loop ran. If this function is called after the loop is completed, the loop count is the total number of times the loop is defined to run.

Usage
```
currentiteration = context.getLoopCount("1234567890");
```

Context.getProcessType()

The method returns the type of the request that triggers the custom workflow process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getProcessType()`

Returns
String

Description
This member function returns the type of the request that triggers the custom workflow process. The function returns `NEWACCOUNT` or `ACCOUNTDATACHANGE`.

Usage `if (context.getProcessType() == context.NEWACCOUNT) {...`

Context.getRequestee()

The method returns the requestee associated with the request as a person object.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getRequestee();`

Returns
A `DirectoryObject` that represents a Person.

Description
This member function returns the requestee associated with the request as a Person object. The requestee is the user who owns the associated, provisioned account.

Usage `requestee = context.getRequestee();`

Context.getService()

The method returns the target service as a service entity object.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`context.getService()`

Returns
`DirectoryObject`

Description
This member function returns the target service as a Service entity object. The service entity is the service associated with the provisioned account.

Usage `service = context.getService();`

Context.isAccountDataChanged()

The method identifies whether a specific account attribute was changed in the request that triggers the custom workflow process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`isAccountDataChanged(String attributeName)`

Description
This member function identifies whether a specific account attribute was changed in the request that triggers the custom workflow process. If the request that triggers the custom workflow is `NEWACCOUNT` and the attribute is in the new account parameters, this function returns TRUE. Otherwise, this function returns FALSE. If the request that triggers the custom workflow is `ACCOUNTDATACHANGE` and the specified attribute is changed, this function returns TRUE. Otherwise, this function returns FALSE.

Usage `if (context.isAccountDataChanged("group")) {...`

Credential

Credentials are associated with a shared access module operation, such as `addCredentialToVault`, `checkin`, or `checkout`.

Availability
IBM Security Privileged Identity Manager 1.0

Inherits from
`DirectoryObject`

Provided by
`com.ibm.itim.script.extensions.model.CredentialModelExtension`

Access mode

`EXCLUSIVE`
Indicates that an authorized user must access the credential through the checkout process.

`NON_EXCLUSIVE`
Indicates that an authorized user can access the credential without the checkout process.

`NON_SHARED`
Indicates that the credential is not intended for sharing.

Notification option

`NOTIFY_ONLY`
When the credential lease expires, a notification email is sent.

`NOTIFY_AND_CHECKIN`
When the credential lease expires, the credential is checked in automatically, and a notification email is sent.

Constructor
`new Credential(dn)`

Returns
The newly created Credential object that represents the credential with the specified DN, which is a String.

Methods

getAccessMode()
Returns an integer constant to represent the access mode, which can be `EXCLUSIVE`, `NON_EXCLUSIVE`, or `NON_SHARED`.

getCheckoutDuration()
Returns the maximum checkout time in hours.

getNotificationRecipient()
Returns the Participant object.

getNotifyOption()
Returns integer constant `NOTIFY_ONLY`, or `NOTIFY_AND_CHECKIN`.

isCheckoutSearchEnable
Returns true if the credential is enabled for search during checkout; returns false, otherwise.

isNotifyOnly()
Returns true if the system is configured to send only a notification when a lease is expired; returns false, otherwise.

isPasswordViewable()
Returns true if the credential password can be displayed to an authorized user; returns false, otherwise.

isResetPasswordAtCheckin()
Returns true if the credential password needs to be reset during the checkin process; returns false, otherwise.

Credential.getAccessMode()

The method returns the access mode of the credential.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
```
Credential.getAccessMode()
```

Returns
Integer

Description
This function returns `EXCLUSIVE`, `NON_EXCLUSIVE`, or `NON_SHARED`.

Usage
```
var accessMode = credential.getAccessMode();
if (accessMode == Credential.EXCLUSIVE) {
 ...;
}
```

Credential.getCheckoutDuration()

The method returns the maximum checkout time for the credential in hours.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
```
Credential.getCheckoutDuration()
```

Returns
Integer

Description
This function returns an integer value in hours.

Usage `var checkoutDuration = credential.getCheckoutDuration();`

Credential.getNotifyOption()

The method returns the notification option when a credential lease is expired.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`Credential.getNotifyOption()`

Returns
Integer

Description
This function returns `NOTIFY_ONLY` or `NOTIFY_AND_CHECKIN`.

Usage

```
var notifyOption = credential.getNotifyOption();
if (notifyOption == Credential.NOTIFY_ONLY) {
 ...;
}
```

Credential.getNotificationRecipient()

The method returns the notification recipient when a credential lease is expired.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`Credential.getNotificationRecipient()`

Returns
Participant

Description
This function returns Participant object to whom the lease expiration email is sent.

Note: The person who checked out the credential always gets a notification when the lease is expired.

Usage `var participant = credential.getNotificationRecipient();`

Credential.isCheckoutSearchEnable()

The method returns whether the credential is enabled for a checkout search.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`Credential.isCheckoutSearchEnable()`

Returns
Boolean

Description
This function returns true if the credential is enabled for a checkout search; returns false otherwise.

Usage

```
var isSearchable = credential.isCheckoutSearchEnable();
if (isSearchable) {
 ...;
}
```

Credential.isNotifyOnly()

The method returns whether the system must send only a notification email when a credential lease is expired or not.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
Credential.isNotifyOnly()
```

Returns

Boolean

Description

This function returns true if the notification option is `NOTIFY_ONLY`; returns false if the notification option is `NOTIFY_AND_CHECKIN`.

Usage

```
var isNotifyOnly = credential.isNotifyOnly();
if (isNotifyOnly) {
 ...;
}
```

Credential.isPasswordViewable()

The method returns whether the credential password can be displayed to an authorized user or not.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
Credential.isPasswordViewable()
```

Returns

Boolean

Description

This function returns true if the credential password can be displayed to an authorized user; returns false, otherwise.

Usage

```
var isDisplayPwd = credential.isPasswordViewable();
if (isDisplayPwd) {
...;
}
```

Credential.isResetPasswordAtCheckin()

The method returns whether to reset the credential password during the checkin process or not.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
Credential.isResetPasswordAtCheckin()
```

Returns

Boolean

Description

This function returns true if the credential password needs to be reset during the checkin process; returns false, otherwise.

Usage

```
var isResetPwd = credential.isResetPasswordAtCheckin();
if (isResetPwd) {
 ...;
}
```

Delegate

The object provides the `Delegate JavaScript` object for use in the JavaScript environment of delegation notification. The `Delegate JavaScript` object and their use is described in this section.

Delegate

The `Delegate` object contains all the information associated with the current delegation operation.

Availability

IBM Security Privileged Identity Manager 1.0

Provided by

`com.ibm.itim.script.extensions.DelegateExtension`

Methods

Delegate.getDelegator()

Returns the `DirectoryObject` that represents a system user such as the IBM Security Privileged Identity Manager account, whose activities are delegated.

Delegate.getDelegatee()

Returns the `DirectoryObject` that represents a system user such as the IBM Security Privileged Identity Manager account, who is selected to be the delegate for the activities of the delegator.

Delegate.getStartDate()

Returns a `Date` that contains the date and time when the delegation starts.

Delegate.getEndDate()

Returns a `Date` that contains the date and time when the delegation ends.

Delegate.getRequester()

Returns the `DirectoryObject` that represents a system user such as the IBM Security Privileged Identity Manager account, who initiated the delegation.

Description

The `Delegate` object is available in the context of a delegation notification. The object retrieves the delegation information in the delegation notification template. The model script extensions are also available in the delegation notification context.

DirectoryObject

The object represents any IBM Security Privileged Identity Manager directory object or entity.

Availability
IBM Security Privileged Identity Manager 1.0

Constructor
There is no specific constructor for this object. Specific constructors for Account, Person, Role, and Service return `DirectoryObject`.

For example, `new Service()` returns a `DirectoryObject`.

Properties

dn String representing the distinguished name of the entity.

name String representing the logical name of the entity.

profileName
String representing the profile name of the entity.

Methods

addProperty()
Changes the value of the specified property, or adds the specified property if it does not exist. For multivalued objects, `addProperty()` adds the values to the specified property in the directory object and does not replace them.

getChanges()
Returns the changes made to the entity.

getProperty()
Returns the values of the property specified by the given name.

getPropertyNames()
Returns a list of properties (attributes and relationships).

removeProperty()
Removes the specified property.

setProperty()
Changes the value of the specified property, or adds the specified property if it does not exist.

getPropertyAsDate()
Returns the value of the specified property as a Date.

getPropertyAsString()
Returns the value of the specified property as a String.

Description
This Object represents a IBM Security Privileged Identity Manager entity in the JavaScript environment. Each IBM Security Privileged Identity Manager entity is wrapped in one of these object classes.

DirectoryObject.addProperty()

The method adds or updates the value for the specified property.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis

directoryObject.addProperty(*name*, *value*)

Arguments

name String representing the name of the property to be created or modified.

value The value to add to the property.

Description

This method changes the value of the specified property or adds the specified property if it does not exist. This change is made locally to the script environment, not to the data store. The value can be a single value object or an array of objects. For multivalued objects, `addProperty()` adds the values to the specified property in the directory object and does not replace them. The value type (syntax) of object must be compatible with the syntax of the specified property. This method is available for the following data types:

- `void addProperty(String name, Collection value);`
- `void addProperty(String name, Date value);`
- `void addProperty(String name, Map value);`
- `void addProperty(String name, boolean value);`
- `void addProperty(String name, byte value);`
- `void addProperty(String name, String value);`
- `void addProperty(String name, number value);`
- `void addProperty(String name, char value);`

Usage

```
directoryObject.addProperty("eruid", "jdoe");
```

The `getProperty` method returns a Java™ array of objects that is stored in a JavaScript `JavaArray` object. Unlike a standard JavaScript array, `JavaArray` objects are used to access members of a Java array. Because Java arrays cannot be resized, the size of a `JavaArray` object cannot be changed. Also, `JavaArray` objects are typed. Setting a `JavaArray` element to the wrong type throws a JavaScript error.

In IBM Security Privileged Identity Manager, a `JavaArray` object cannot be passed directly back into a `addProperty` method. The `JavaArray` array might be converted into a standard JavaScript array as follows:

```
jsAliases = new Array();
myPerson = person.get();
aliases = myPerson.getProperty("eraliases");
for (i=0; i < aliases.length; i++) {
jsAliases[i] = aliases[i];
}
jsAliases[aliases.length] = "myNewAlias";
myPerson.addProperty("eraliases", jsAliases);
person.set(myPerson);
```

DirectoryObject.dn

The field represents the distinguished name of the object.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

directoryObject.dn

Description

This read-only field is a string that provides the distinguished name of the object. If the object holds information that was not created, there is no value.

Usage `x = directoryObject.dn;`

DirectoryObject.getChanges()

The method returns the changes made to the entity.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

`directoryObject.getChanges()`

Returns

An array of change objects. If there are no changes, an empty array is returned. Each element in the array is an `AttributeChangeOperation`.

Description

This method returns the changes made to the entity. These changes are represented by change objects with the following members:

attr String name of the attribute that is being changed.

op An integer that identifies the type of change that is being made. The enumerated values are 1 for add, 2 for replace, and 3 for remove.

values An array of objects that must be either added, removed, or replaced.

The changes are returned as an array of these change objects. If there are no changes, an empty array is returned.

Usage

```
changes = directoryObject.getChanges();
for (i = 0; i < changes.length; i++) {
  name = changes[i].attr;
  if (changes[i].op == 1) {
    ...
  } else if (changes[i].op == 2) {
    ...
  } else {
    ...
  }
};
```

DirectoryObject.getProperty()

The method returns the values of the property specified by the given name.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

`directoryObject.getProperty(name)`

Arguments

name String representing the name of the property to return.

Returns

Either a String or a `DirectoryObject`. The type of object returned depends on the property obtained. If the specified property does not exist, an empty array is returned.

Description

This method returns the values of the property specified by the given name. The type of object returned depends on the property obtained. If the specified property does not exist, an empty array is returned.

The property name can be either an attribute name or a relationship name. For an attribute name, the return is a `String[]`; for a relationship name, an array of `DirectoryObjects` is returned. If an attribute and a relationship have the same name, then the attribute is returned. For example, an Account entity has both an owner attribute and an owner relationship.

Usage When operating on an account, for example, the user ID property can return a String, where the owner property can return another entity (`DirectoryObject`). The owner entity can then be operated on with the `getProperty()` member to obtain information about it.

```
userids = directoryObject.getProperty("eruid");
if (userids.length > 0)
    userid = userids[0];
owner = directoryObject.getProperty("owner");
if (owner.length > 0)
    ownerName = owner.getProperty("name")[0];
```

Note: These statements assume there is at least one value returned. If no values are returned, an array indexing violation occurs.

The `getProperty` method returns a Java array of objects that is stored in a JavaScript `JavaArray` object. Unlike a standard JavaScript array, `JavaArray` objects are used to access members of a Java array. Since Java arrays cannot be resized, the size of a `JavaArray` object cannot be changed. Also, `JavaArray` objects are typed. Setting a `JavaArray` element to the wrong type throws a JavaScript error.

DirectoryObject.getPropertyAsDate()

The method returns the value of the property specified by the given name as a date object.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

`directoryObject``.getPropertyAsDate(name)`

Arguments

name String representing the name of the property to return.

Returns

A `Date` object. If the specified property does not exist, current date is returned.

Description

This method returns the value of the property specified by the given name as a date object. If the specified property does not exist, current date is returned.

Usage

```
var createDate = directotyObject. getPropertyAsDate("ercreatedate");
```

DirectoryObject.getPropertyAsString()

The method returns the value of the property specified by the given name as a string.

Availability
IBM Security Privileged Identity Manager 1.0.

Synopsis
directoryObject.getPropertyAsString(name)

Arguments

name String representing the name of the property to return.

Returns
A `String` object. If the specified property does not exist, empty is returned. If the specified property has multiple values, only the first value is returned.

Description
This method returns the value of the property specified by the given name as a `String` object. If the specified property does not exist, empty string is returned. If the specified property has multiple values, only the first value is returned.

Usage

```
var name = directotyObject.getPropertyAsString("erservicename");
```

DirectoryObject.getPropertyNames()

The method returns a list of properties, such as attributes and relationships.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
directoryObject.getPropertyNames()

Returns
An array of Strings.

Description
This method returns a list of properties as an array of Strings. A property can be either an attribute or a relationship.

Usage `properties = directoryObject.getPropertyNames();`

DirectoryObject.name

The field represents the logical name of the object.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
directoryObject.name

Description
This read-only field is a string that provides the logical name of the object, represented as a String. The physical attribute used as the name can be different for each type of object.

Usage `x = directoryObject.name;`

DirectoryObject.profileName

The field returns the object profile name.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`directoryObject.profileName()`

Description
This read-only field is a string that provides the profile name of the object, represented as a String.

Usage

```
x = directoryObject.profileName;
```

DirectoryObject.removeProperty(name)

The method removes the property specified by the given name.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`directoryObject.removeProperty(name)`

Arguments

name String representing the name of the property to remove.

Description
This method removes the specified property. This change is made locally to the script environment, not to the data store. The property name can be either an attribute name or a relationship name.

Usage `directoryObject.removeProperty("eruid");`

DirectoryObject.removeProperty(name,value)

The method removes the value from the specified property.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`directoryObject.removeProperty(name,value)`

Arguments

name String representing the name of the property to be modified.

value The value to remove from the property.

Description
This method removes the specified value from property if it exists. This change is made locally to the script environment, not to the data store. The value can be a single value object or an array of objects. For multivalued objects, removeProperty(name,value) removes the values from the specified property in the directory object. The object type of the value (syntax) must be compatible with the syntax of the specified property. This method is available for the following data types:

- void removeProperty(String name, Collection value);
- void removeProperty(String name, Date value);
- void removeProperty(String name, Map value);
- void removeProperty(String name, boolean value);
- void removeProperty(String name, byte value);
- void removeProperty(String name, String value);
- void removeProperty(String name, Number value);

Usage

```
var directoryObject = Entity.get();
directoryObject.removeProperty("eraliases", "jdoe");
Entity.set(directoryObject);
```

DirectoryObject.setProperty()

The method sets the value of the specified property.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

directoryObject.setProperty(*name*, *value*)

Arguments

name String representing the name of the property to be created or modified.

value The value to set the property to.

Description

This method changes the value of the specified property, or adds the specified property if it does not exist. This change is made locally to the script environment, not to the data store. The value can be a single value object or an array of objects. The value type (syntax) of object must be compatible with the syntax of the specified property. This method is available for the following data types:

- `void setProperty(String name, Collection value);`
- `void setProperty(String name, Date value);`
- `void setProperty(String name, Map value);`
- `void setProperty(String name, boolean value);`
- `void setProperty(String name, byte value);`
- `void setProperty(String name, String value);`
- `void setProperty(String name, number value);`
- `void setProperty(String name, char value);`

Usage *directoryObject*.setProperty("eruid", "jdoe");

The `getProperty` method returns a Java array of objects that is stored in a JavaScript `JavaArray` object. Unlike a standard JavaScript array, `JavaArray` objects are used to access members of a Java array. Since Java arrays cannot be resized, the size of a `JavaArray` object cannot be changed. Also, `JavaArray` objects are typed. Setting a `JavaArray` element to the wrong type throws a JavaScript error.

In IBM Security Privileged Identity Manager, a `JavaArray` object cannot be passed directly back into a `setProperty` method. The `JavaArray` array into a standard JavaScript array as follows:

```
jsAliases = new Array();
myPerson = person.get();
aliases = myPerson.getProperty("eraliases");
for (i=0; i < aliases.length; i++) {
    jsAliases[i] = aliases[i];
}
jsAliases[aliases.length] = "myNewAlias";

myPerson.setProperty("eraliases", jsAliases);
person.set(myPerson);
```

EmailContext

The object provides access to contextual information specific to a type of notification that is sent.

Some methods for accessing information change are based upon the listed notification types. (The Reminder/Approval/RFI/WorkOrder/ComplianceAlert Notification does not support this.)

- Activity Timeout Template
- Change Account Template
- Compliance Template
- New Account Template
- New Password Template
- Process Completion Template
- Process Timeout Template
- Restore Account Template
- Suspend Account Template

Availability

IBM Security Privileged Identity Manager 1.0

Provided by

`com.ibm.itim.workflow.script.EmailContextExtension`

Synopsis

Call methods documented in this section as an `EmailContext` object. For example:

```
notificationActivity=EmailContext.getActivity();
owner=EmailContext.getAccountOwnerName()
```

Common methods

These methods are available for all types of notifications:

getActivity()

Returns information about the most recent running activity. (Returns the `ActivityInfoOC` Java Object. To get the activity information in JavaScript object, use the object, 'activity'.

getActivity(java.lang.String actDefID)

Returns information about the activity with the specified definition ID. (Returns the `ActivityInfoOC` Java Object.) This obtains information by using the `Process.$dataName.get()workflow` process. To get the activity information in JavaScript object, use '`process.getActivity(java.lang.String actDefID)`'.

getParentProcess()

Returns information about the parent process of the currently

running process. (Returns the `ProcessInfoOC` Java object.) To get the process information of the parent process in JavaScript object, use ‘`process.getParent()`'.

getProcess()
Returns the information about the currently running process. (Returns the `ProcessInfoOC` Java object.) To get the process information of the parent process in JavaScript object, use the object, ‘process'.

getRootProcess()
Returns information about the root process of the current running process. (Returns the `ProcessInfoOC` Java object.) To get the process information of the parent process in JavaScript object, use `'process.getRootProcess ()')`.

Account notification methods
These methods are available for all types of account notifications:

getAccountOwnerName()
Returns the account owner name for the account.

getAccountServiceName()
Returns the account service name for the account.

getAccountServiceProfileName()
Returns the account service profile name for the account.

getAccountUserId()
Returns the account user ID for the account.

hasNewAccess()
Returns true if the account has new access and false otherwise.

hasRemovedAccess()
Returns true if the account removed access and false otherwise.

getAccountNewAccessAsString()
Returns String that contains list of new access separated by commas.

getAccountNewAccessList()
Returns Array of String that contains the new access.

getAccountRemovedAccessAsString()
Returns a string that contains the list of removed access separated by commas.

getAccountRemovedAccessList()
Returns Array of String that contains the list of removed access.

Account Suspend/Deprovisioning Notification Methods:
These methods are only available for all types of account suspend/deprovision notifications:

getAction()
Returns the action taken against the service (resource) itself.

getReason()
Returns a descriptive reason for the deprovision.

Account New/Modify/Restore Notification Methods:
These methods are only available for all types of notifications for new, modified, and restored accounts:

showPassword()
Returns whether to display the password when the user is notified of their new account.

getAccountPassword()
Returns the account password for the account. .

getPasswordExpirePeriod()
Returns the password delivery expiration period.

getPasswordRetrievalUrl()
Returns the password delivery URL in order to retrieve the password with the accounts shared secret.

getTransactionId()
Returns the password delivery transaction ID for picking up the password created for this account.

Account Password Change Notification Methods:
These methods are available for all types of account password change notifications:

getAccountPassword()
Returns the account password for the account.

getPasswordExpirePeriod()
Returns the password delivery expiration period.

getPasswordRetrievalUrl()
Returns the password delivery URL in order to retrieve the password with the accounts shared secret.

getTransactionId()
Returns the password delivery transaction ID for picking up the password created for this account.

Enrole

The object contains the general methods.

Availability

- All JavaScript contexts
- IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.EnroleExtension`

Methods

generatePassword()
Generates a password for a specific service.

getAttributeValue()
Get a single value attribute value.

getAttributeValues()
Get a multi-valued attribute value.

localize()
Localized message specified in `<Message>` XML format.

log() Logs a message to the IBM Security Privileged Identity Manager log at `ERROR` level.

logError()
Logs the specified text to the IBM Security Privileged Identity Manager message log (`msg.log`) at `ERROR` level.

logInfo()
Logs the specified text to the IBM Security Privileged Identity Manager message log (`msg.log`) at `INFO` level.

logWarning()
Logs the specified text to the IBM Security Privileged Identity Manager message log (`msg.log`) at `WARN` level.

toGeneralizedTime()
Converts a time or date to generalized time format.

toMilleseconds()
Converts a String in generalized time format to an integer value in milliseconds.

traceMax()
Logs the specified text to the IBM Security Privileged Identity Manager trace log (`trace.log`) at `DEBUG_MAX` level.

traceMid()
Logs the specified text to the IBM Security Privileged Identity Manager trace log (`trace.log`) at `DEBUG_MID` level.

traceMin()
Logs the specified text to the IBM Security Privileged Identity Manager trace log (`trace.log`) at `DEBUG_MIN` level.

Description
Provides some common utilities for use in many different scripting contexts.

Enrole.generatePassword()

The method generates a new valid password for an account.

Availability
`generatePassword()` requires a service to work, so `generatePassword()` is only available when the `ServiceExtension` is used.

Synopsis
```
Enrole.generatePassword()
```

Returns
A String that is a valid password for the Service `DirectoryObject` stored in the "service" variable.

Description
This method generates a new valid password for a service.

Enrole.getAttributeValue()

The method retrieves the attribute's value.

Availability
Deprecated as of IBM Security Privileged Identity Manager 1.0. Replace with `DirectoryObject.getProperty()`

Synopsis
```
Enrole.getAttributeValue(name, defaultValue)
```

Arguments

name String representing the name of the property to return.

defaultValue
Default value to return if there is no value to return.

Returns
An Object. The type of object returned depends on the property obtained. If the specified property does not exist, the default value is returned.

Description
This method retrieves the value of the specified property.

Enrole.getAttributeValues()

The method retrieves a multi-valued attribute value.

Availability
Deprecated as of IBM Security Privileged Identity Manager 1.0. Replace with `DirectoryObject.getProperty()`

Synopsis
`Enrole.getAttributeValues(`*name*`)`

Arguments

name String representing the name of the property to return.

Returns
An array of objects. The type of object returned depends on the property obtained. If the specified property does not exist, an empty array is returned.

Description
This method retrieves the value of the specified property.

Enrole.localize()

The method localizes a message specified in <Message> XML format.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`Enrole.localize(String xmlMsg, String localStr)`

Arguments

xmlMsg
A message specified in XML.

localStr
A String that represents the locale to be used for globalization.

Returns
AA localized message.

Description
This method globalizes an XML message to the specified locale.

Enrole.log()

The method logs messages to the IBM Security Privileged Identity Manager message log (`msg.log`).

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

`Enrole.log(`*`category`*`, `*`message`*`);`

Arguments

category

The category of the log entry, entered as a String. The category argument can be used or it can be left empty, but the argument must not be null.

message

The message to be logged, entered as a String.

Description

Logs a message to the IBM Security Privileged Identity Manager log at error level.

Usage

```
var roleDN = ..;(DN of role)
var role = new Role(roleDN);

//  Put next statement on one line

Enrole.log("script", "The role name is
 "+ role.getProperty("errolename")[0]);
```

Use the following new methods in IBM Security Privileged Identity Manager Version 1.0 to provide greater adaptability, control, or flexibility over the `Enrole.log()` method:

- logError()
- logInfo()
- logWarning()
- traceMax()
- traceMid()
- traceMin()

Enrole.logError()

The method logs text messages to the IBM Security Privileged Identity Manager message log (`msg.log`) with a message severity level of `ERROR`.

Availability

IBM Security Privileged Identity Manager Version 1.0

Synopsis

`Enrole.logError(`*`component`*`, `*`method`*`, `*`message`*`);`

Arguments

component

The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the `enRoleLogging.properties` file.

method

The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message

The string to represent the contents of the message log to be written to the log file.

Description

Writes an error message to the IBM Security Privileged Identity Manager message log (`msg.log`).

Usage An example to write a `msg.log` message at `ERROR` level with the component name `com.ibm.myExtension` and the method name `postScriptOfAccountCreate`:

```
var userName = "Joe";
 // below is a single line
 Enrole.logError("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording error message after unsuccessful account creation for user "
    + userName + ".");
```

Enrole.logInfo()

The method logs text messages to the IBM Security Privileged Identity Manager message log (`msg.log`) with a message severity level of `INFO`.

Availability

IBM Security Privileged Identity Manager Version 1.0

Synopsis

`Enrole.logInfo(`*`component`*`,` *`method`*`,` *`message`*`);`

Arguments

component

The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the `enRoleLogging.properties` file.

method

The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message

The string to represent the contents of the message log to be written to the log file.

Description

Writes an error message to the IBM Security Privileged Identity Manager message log (`msg.log`).

Usage An example to write a `msg.log` message at `INFO` level with the component name `com.ibm.myExtension` and the method name `postScriptOfAccountCreate`:

```
var userName = "Joe";
 // below is a single line
 Enrole.logInfo("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording information message after account creation for user " + userName + ".");
```

Enrole.logWarning()

The method logs text messages to the IBM Security Privileged Identity Manager message log (`msg.log`) with a message severity level of `WARN`.

Availability

IBM Security Privileged Identity Manager Version 1.0

Synopsis

`Enrole.logWarning(`*`component`*`, `*`method`*`, `*`message`*`);`

Arguments

component
The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the `enRoleLogging.properties` file.

method
The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message
The string to represent the contents of the message log to be written to the log file.

Description

Writes a warning message to the IBM Security Privileged Identity Manager message log (`msg.log`).

Usage An example to write a `msg.log` message at `WARN` level with the component name `com.ibm.myExtension` and the method name `postScriptOfAccountCreate`:

```
var userName = "Joe";
 // below is a single line
 Enrole.logWarning("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording warning message after account creation for user " + userName + ".");
```

Enrole.toGeneralizedTime()

The method converts a time or date to generalized time format.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

`Enrole.toGeneralizedTime(`*`time`*`)`

Arguments

time Integer time in milliseconds or a Date object.

Description

This method converts a time or date to generalized time format. Can be used in either Identity Policies or in default entitlements.

Usage `genTime = Enrole.toGeneralizedTime(seconds);`

Enrole.toMilliseconds()

The method converts a string in generalized time format to an integer value in milliseconds.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
Enrole.toMilliseconds(*genTime*)

Arguments

genTime
String in generalized time format.

Description
This method converts a String in generalized time format to an integer value in milliseconds.

Usage
```
seconds = Enrole.toMilliseconds(genTime);
```

Enrole.traceMax()

The method logs text messages to the IBM Security Privileged Identity Manager trace log (trace.log) with a message severity level of DEBUG_MAX.

Availability
IBM Security Privileged Identity Manager Version 1.0

Synopsis
Enrole.traceMax(*component*, *method*, *message*);

Arguments

component
The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the enRoleLogging.properties file.

method
The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message
The string to represent the contents of the trace message to be written to the log file.

Description
Writes a DEBUG_MAX message to the IBM Security Privileged Identity Manager trace log (trace.log).

Usage
An example to write a trace.log message at DEBUG_MAX level with the component name com.ibm.myExtension and the method name postScriptOfAccountCreate:

```
var userName = "Joe";
 // below is a single line
 Enrole.traceMax("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording DEBUG_MAX trace message after account creation for user " + userName + ".");
```

Enrole.traceMid()

Logs text messages to the IBM Security Privileged Identity Manager trace log (trace.log) with a message severity level of DEBUG_MID.

Availability
IBM Security Privileged Identity Manager Version 1.0

Synopsis

```
Enrole.traceMid((component, method, message);
```

Arguments

component
The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the `enRoleLogging.properties` file.

method
The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message
The string to represent the contents of the trace message to be written to the log file.

Description

Writes a `DEBUG_MID` message to the IBM Security Privileged Identity Manager trace log (`trace.log`).

Usage An example to write a `trace.log` message at `DEBUG_MID` level with the component name `com.ibm.myExtension` and the method name `postScriptOfAccountCreate`:

```
var userName = "Joe";
 // below is a single line
 Enrole.traceMid("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording DEBUG_MID trace message after account creation for user " + userName + ".");
```

Enrole.traceMin()

The method logs text messages to the IBM Security Privileged Identity Manager trace log (`trace.log`) with a message severity level of `DEBUG_MIN`.

Availability

IBM Security Privileged Identity Manager Version 1.0

Synopsis

```
Enrole.traceMin((component, method, message);
```

Arguments

component
The component of the log entry, entered as a String. The component can be any string. Logging can be controlled for components by setting specific log levels in the `enRoleLogging.properties` file.

method
The string to display in the "Method" record of the message log. Useful to point where in the script the message originated.

message
The string to represent the contents of the trace message to be written to the log file.

Description

Writes a `DEBUG_MIN` message to the IBM Security Privileged Identity Manager trace log (`trace.log`).

Usage An example to write a `trace.log` message at `DEBUG_MIN` level with the component name `com.ibm.myExtension` and the method name `postScriptOfAccountCreate`:

```
var userName = "Joe";
 // below is a single line
 Enrole.traceMin("com.ibm.myExtension","postScriptOfAccountCreate",
"Recording DEBUG_MIN trace message after account creation for user " + userName + ".");
```

Error

This object contains a script error description to notify the calling code of an exceptional runtime condition.

When an error is returned from a script evaluation, it is converted to a Java exception and thrown from the script evaluator class.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.EnroleExtension`

Methods

setMessage()
Sets the message for the error.

getMessage()
Retrieves the error message for the error.

setErrorCode()
Sets the error code for the error.

getErrorCode()
Retrieves the error code for the error.

Usage

```
var sn = subject.getProperty("sn");
if(sn == null || sn.length == 0) {
   error.setMessage("sn was missing");
   return error;
} else {
  return sn[0];
}
```

Error.setMessage()

The method sets the message for the error.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`error.setMessage(`*`String msg`*`)`

Arguments

msg String representing the message to be set.

Description
This method sets the text for an error message. The function takes in one String parameter.

Usage `error.setMessage("sn was missing");`

Error.getMessage()

The method retrieves the message set for an error.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`error.getMessage()`

Returns
String message for an error.

Description
This method retrieves the text of an error message.

Usage `messageValue = error.getMessage();`

Error.setErrorCode()

The method sets the error code for the error.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`error.setErrorCode(int code)`

Arguments

code Integer representing the error code.

Description
This method sets the error code for an error message. The function takes in one **Int** parameter.

Usage `error.setErrorCode(1);`

Error.getErrorCode()

The method retrieves the error code set for an error.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`error.getErrorCode()`

Returns
Integer value for an error code.

Description
This method retrieves the error code of an error message.

Usage `errorCodeValue = error.getErrorCode();`

Participant

Workflow participant entity, which specifies an activity participant. In a mail node, this entity specifies the mail recipient.

Participant applies only to manual activity types, including `Approval`, `RFI`, `WorkOrder`, and `Mail`.

The participant of an activity can be specified during workflow design as Custom Defined Participant. In this case, the Participant JavaScript object can be used to construct the appropriate participant based on the process context.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.workflow.script.WorkflowExtension`

Constructor
`new Participant(type, dn)`

Arguments

type Code that categorizes the participant type.

dn Optional DN of a specific entity.

Returns
The newly created and initialized participant object.

Properties

implementation
This property contains JavaScript that returns participant when the participant type is Custom.

name Identifies the participant.

type Code that categorizes the participant type.

Description
The participant specifies an activity participant. Participant applies only to manual activity types, including `Approval`, `RFI`, `Work Order` and `Mail` activities. The participant of an activity or recipient of a mail activity can be specified during workflow design as Custom Defined Participant. In this case, the Participant JavaScript object can be used to construct the appropriate participant based on the process context.

Usage

```
//assume person is one of the relevant data in the workflow
//process for the target user involved
if( person.get().getProperty("title")[0]=="Manager" )
  return new Participant(ParticipantType.SYSTEM_ADMIN);
else
  return new Participant(ParticipantType.SUPERVISOR);


//assume person is one of the relevant data in the workflow
//process for the target user involved
if( person.get().getProperty("title")[0]=="Manager")
  return new Participant(ParticipantType.USER, person.get().dn);
else
  ...
```

Participant.implementation

The field represents the custom defined participant.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
participant.implementation

Description
This read-only field is a string that provides the custom-defined participant, which contains the JavaScript code to return the participant.

Usage `x = participant.implementation;`

Participant.name

The field represents the DN of the participant.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`participant.name`

Description
This read-only field is a Distinguished Name that identifies the participant. It is only applicable to participant types of `ROLE` and `USER`.

Usage `x = participant.name;`

Participant.type

the field represents the code that categorizes the participant type.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`participant.type`

Description
This read-only field is a string that represents a code that categorizes the participant type.

Usage `x = participant.type;`

ParticipantType

An entity that represents the workflow participant type constants.

Availability
IBM Security Privileged Identity Manager 1.0.

Provided by
`com.ibm.itim.workflow.script.WorkflowExtension`

Properties

DOMAIN_ADMIN
Participant type for the domain administrator of the organizational container. It is associated with the Subject account service (as specified by the Subject context in the workflow properties window).

```
participant = new Participant(ParticipantType.DOMAIN_ADMIN);
```

REQUESTOR
Participant type for the person that initiated the request. If a person initiates a change request for a person that triggers policy enforcement, the participant is the person that requests the change. For data loads, the participant is the system user. By setting the following property in `$ISPIM_HOME/data/enRole.properties` to

true, an approval request that has the requester as the participant is automatically approved by the system:

```
participant = new Participant(ParticipantType.REQUESTOR);
```

REQUESTEE
Participant type for the person designated as the requestee in the owner field of the relevant data.

```
participant = new Participant(ParticipantType.REQUESTEE);
```

ROLE Participant type for a specific organizational role. All user members of the role and its child roles are notified and are eligible to respond, the first response triggers the workflow to continue. In other words, specifying a role cannot be used to require multiple participants to approve the request.

```
participant = new Participant(ParticipantType.ROLE, roleDN);
```

ROLE_OWNER
Participant type for the owner of the role (if specified). The Role is resolved based on the owners specified in the `OrgRole` listed as an input parameter for the operational workflow operation. If there is no `OrgRole` specified as an input parameter in the workflow, the participant is not resolved.

```
participant = new Participant(ParticipantType.ROLE_OWNER);
```

SERVICE_OWNER
Participant type for the owner of the service (if specified). The Service is resolved based on the account object from the workflow relevant data that is marked as "Subject" in the properties window.

```
participant = new Participant(ParticipantType.SERVICE_OWNER);
```

SOD_POLICY_OWNER
Participant type for the owners of the separation of duty policy (if specified). The owners are resolved based on the `SeparationOfDutyRuleViolation` object from the workflow relevant data that is marked as "Subject" in the properties window. If there is no `SeparationOfDutyRuleViolation` specified as the Subject of the workflow, the participant is not resolved.

The SOD_POLICY_OWNER participant type is used only in the `approveSoDViolation` global operation.

```
participant = new Participant(ParticipantType.SOD_POLICY_OWNER);
```

SPONSOR
Participant type for the person designated as the sponsor with the sponsor relationship for the requestee (as marked in relevant data).

```
participant = new Participant(ParticipantType.SPONSOR);
```

SUPERVISOR
Participant type for the supervisor or manager of the requestee. If none is specified for the requestee, then the supervisor designated on the organizational container of the requestee becomes the participant. If no supervisor is specified for the organizational container of the requestee, then the next level up is checked for a supervisor. The search continues up the tree until the top of the organization is reached. If no supervisor is found, the participant is unresolved.

```
participant = new Participant(ParticipantType.SUPERVISOR);
```

SYSTEM_ADMIN
Participant type for a member of the IBM Security Privileged Identity Manager System Administrator group.

```
participant = new Participant(ParticipantType.SYSTEM_ADMIN);
```

USER Participant type for a specific person to respond to the request. The person must have a IBM Security Privileged Identity Manager account.

```
participant = new Participant(ParticipantType.USER, userDN);
```

SYSTEM GROUP
Participant type for a specific system group. Though all members of the group are notified, and all are eligible to respond, the first response triggers the workflow to continue. Specifying a group cannot be used to require multiple participants to approve the request.

```
participant = new Participant(ParticipantType.GROUP, groupDN);
```

Description
This entity represents the workflow participant type constants.

Person

The object represents the person entity.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.model.PersonModelExtension`

Inherits From
`DirectoryObject`

Constructors

new Person(String dn)
Arguments:

dn Optional DN of a specific entity.

new Person(DirectoryObject directoryObject)
Arguments:

directoryObject
`DirectoryObject` to be contained in the person

new Person(DirectoryObjectEntity directoryObjectEntity)
Arguments:

directoryObjectEntity
`DirectoryObjectEntity` to be contained in the person

Methods

getAllAssignmentAttributes()
Returns an array of the `RoleAssignmentAttribute` objects that are defined in all of authorized roles for this person. The authorized roles consist of both the direct roles for this person and also all of the parent roles of the direct roles.

getRoleAssignmentData()
Returns all role assignment data for the person.

getRoleAssignmentData(String roleAssignedDN)
Returns all role assignment data for the person for the specified role.

getRoles()
Returns an array of DirectoryObjects, each representing a role.

getNewRoles()
Returns an array of newly added roles for the person.

getRemovedRoles()
Returns an array of removed roles for the person.

isInRole(String roleName)
Determines whether the person belongs to the role. Returns Boolean.

removeRole()
Removes the person from the specified role.

removeRoleAssignmentData(String roleAssignedDN)
Removes all role assignment data for the person from the specified role.

updateRoleAssignmentData(RoleAssignmentObject[] roleAssignmentObject)
Updates a person with the role assignment attribute value changes that are defined in the set of `RoleAssignmentObjects`.

Person.getAllAssignmentAttributes()

The method returns an array of the `RoleAssignmentAttribute` objects that are defined for all of authorized roles for this person. The authorized roles consist of both the direct roles for this person and also all the parent roles of the direct roles.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
person`.getAllAssignmentAttributes()`

Arguments
None

Description
This method is defined on the Person object. It returns an array of the `RoleAssignmentAttribute` objects that are defined in all of authorized roles for this person. The authorized roles consist of both the direct roles for this person and also all the parent roles of the direct roles. The method returns an empty array if no assignment attribute exists. `RoleAssignmentAttribute` objects contains role assignment attribute name, role name, and role DN.

Usage

```
//The script is used in a workflow, in which Entity is a person object.
var person = Entity.get();

//get assignment attributes of the person
var attributeList = person.getAllAssignmentAttributes();
if (attributeList.length == 0) {
    Enrole.log("script", "No assignment attribute for this role: "
        + role.name);
    return;
}
```

```
// print out the role assignment attribute name.
for (var i=0; i < attributeList.length; i++) {
    var roleAtr = attributeList[i];
    Enrole.log("script","attribute name-----: "+ roleAtr.getName());
}
```

Person.getRoleAssignmentData()

The method returns all the role assignment data for the person, as an array of `RoleAssignmentObject` objects that contain the role assignment values, defined Role DN and assigned Role DN.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

person`.getRoleAssignmentData()`

Arguments

none

Description

This method is defined on the Person object. It returns an array of `RoleAssignmentObject` objects, containing the role assignment values, defined Role DN, and assigned Role DN. The method returns an empty array if no assignment data exists.

Usage

```
//The script is used in a workflow, in which Entity is a person object.
var person = Entity.get();
 var assignmentObjects = person.getRoleAssignmentData();
 if (assignmentObjects.length == 0) {
  Enrole.log("script", "There is no assignment values for " + person.name);
  return;
 }
 var str = "The number of role assignment objects returned from
      person.getRoleAssignmentData(): " +
     assignmentObjects.length + "\n";
 for(var i=0; i<assignmentObjects.length; i++) {
  var obj = assignmentObjects[i];
  str += obj.toString() + "\n";
 }
 Enrole.log("script", "The assignment attribute data for person:"+
      person.name+" is:"+ str);
```

Person.getRoleAssignmentData(String roleAssignedDN)

The method returns all the role assignment data for the person. The data is an array of `RoleAssignmentObject` objects that contain the role assignment values, defined Role DN, and assigned Role DN for the specified assigned role.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

person`.getRoleAssignmentData(String roleAssignedDN)`

Arguments

roleAssignedDN

The distinguished name of the assigned role

Description

This method is defined on the Person object. It returns an array of `RoleAssignmentObject` objects, containing the role assignment values,

defined Role DN, and assigned Role DN for a specified assigned role. The method returns an empty array if no assignment data exists.

Usage

```
//The script is used in a workflow, in which Entity is a person object.
var person = Entity.get();
var roleDNs = person.getProperty("erroles");
if(roleDNs.length == 0) {
 Enrole.log("script", person.name + " does not have any role");
 return;
}
// Get role assignment data for the first role.
var roleDN = roleDNs[0];
var role = new Role(roleDN);
var assignmentObjects = person.getRoleAssignmentData(roleDNs[0]);
if (assignmentObjects.length == 0) {
 Enrole.log("script", person.name + " does not have any assignment
   objects for role: + role.name);
 return;
}
var str = "The number of role assignment objects returned from
    person.getRoleAssignmentData() for "
    + role.name + " :" + assignmentObjects.length + "\n";
for(var i=0; i<assignmentObjects.length; i++) {
 var obj = assignmentObjects[i];
 str += obj.toString() + "\n";
}
Enrole.log("script", str);
```

Person.getRoles()

The method returns roles assigned to a Person.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
person.getRoles()
```

Description

This method defined on the Person object returns an array of roles that the person belongs to. The return type is an array of entities, which are instances of role directory entity objects. The properties available on the Entity Objects are `name` and `description`.

Usage

```
// logs the names of all roles that a person belongs to
var per = person.get();
var rolesArray = per.getRoles();
if(rolesArray.length>0){
   Enrole.log("script", per.getProperty("cn")[0] +
      " belongs to following roles: ");

   for( var i=0; i<rolesArray.length;i++) {
      Enrole.log("script",
         rolesArray[i].getProperty("errolename")[0]);
   }
} else {
   Enrole.log("script", per.getProperty("cn")[0] +
      "does not belong to any roles");
}
```

Person.getNewRoles()

The method returns an array of newly added static roles for a Person.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
person.getNewRoles()

Description
This method defined on the person object returns an array of new static roles associated with the person. The return type is an array of DirectoryObjects,

Note: The person object is often a runtime object in memory, and these new static roles were not added to the directory.

Usage

```
var newRoles = per.getNewRoles();
```

Person.getRemovedRoles()

The method returns an array of removed static roles for the Person.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
person.getRemovedRoles()

Description
This method defined on the person object returns an array of static roles from which the person was removed. The return type is an array of DirectoryObjects.

Note: The person object is often a runtime object in memory, and these static roles were not removed from the directory.

Usage

```
var removedRoles = per.getRemovedRoles();
```

Person.isInRole()

The method evaluates whether a Person belongs to a role.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
person.isInRole(*roleName*)

Arguments

roleName
The name of the role to check.

Description
Given a person object and the name of the role, determine whether the person belongs to the role. If the role is not uniquely determined by the roleName parameter or if the person cannot be found, then return an error object.

Usage

```
// Check whether the person is in the role Manager and log a
// message
var per=person.get();
```

```
if(!per.isInRole("Manager")) {
   Enrole.log("script",per.getProperty("cn")[0] +
      "does not belong to role Manager");
} else {
   Enrole.log("script",per.getProperty("cn")[0] +
      "belong to role Manager");
}
```

Person.removeRole()

The method removes the person from the specified role.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

person.removeRole(*role*)

Arguments

role Role object that represents the role from which the person is removed.

Description

Removes the person from the role.

Note: This operation removes only the role from the Person object in run time, and it does not remove the role from the directory.

Usage

```
//Remove the first role in the Person object
var roles = person.getRoles();
if (roles.length > 0) {
   person.removeRole(roles[0]);
}
```

Person.removeRoleAssignmentData()

The method removes all role assignment data of the person for an array of assigned Roles. It does not directly change data in the data source, but removes from memory the data inside the person object.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

person.removeRoleAssignmentData(String [] roleAssignedDNs)

Arguments

roleAssignedDNs
An array of distinguished names of the assigned role.

Description

This method is defined on the Person object. It removes all role assignment data of the person for an array of assigned roles.

Usage

```
//The script is used in a workflow, in which Entity is a person object.
var person = Entity.get();
var roleDNs = person.getProperty("erroles");
if(roleDNs.length == 0) {
 Enrole.log("script", person.name + " does not have any roles");
 return;
}
```

```
//remove the role assignment attribute.
person.removeRoleAssignmentData(roleDNs);
```

Person.updateRoleAssignmentData()

The method updates a person with the role assignment attribute value changes that are defined in the set of `RoleAssignmentObjects`. It does not directly change data in the data source, but updates (in memory) the data inside the person object.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

person`.updateRoleAssignmentData(RoleAssignmentObject [] roleAssignmentObject)`

Arguments

roleAssignmentObject
A list of `roleAssignmentObjects` that contains the role assignment attribute value change set to be applied.

Description

This method is defined on the Person object. It updates a person with the role assignment attribute value changes that are defined in the set of `RoleAssignmentObjects`.

Usage

```
//The script is used in a workflow, in which Entity is a person object.
var person = Entity.get();
var roleDNs = person.getProperty("erroles");
if(roleDNs.length == 0) {
 Enrole.log("script", person.name + " does not have any role");
 return;
}

//construct a new RoleAssignmentObject
var assignmentObj = new RoleAssignmentObject(roleDNs[0], roleDNs[0]);
assignmentObj.addProperty("attr_3", ["newv1", "newv2"]);
person.updateRoleAssignmentData([assignmentObj]);
```

PersonSearch

The object searches for a person.

Availability

IBM Security Privileged Identity Manager 1.0
Provisioning Policy context
Service Selection Policy context

Provided by

`com.ibm.itim.script.extensions.model.PersonModelExtension`

Constructor

`new PersonSearch()`

Returns

The newly created and initialized person search object.

Methods

searchByFilter()
Search for a person by a filter.

searchByURI()
Search for a person by URI in an organizational container.

Description
The entity implements the IBM Security Privileged Identity Manager `PersonSearch` class. The API Javadoc for this class is in the following directory:

`$ISPIM_HOME/extensions/version_number/api/com/ibm/itim/dataservices/model/domain/`

PersonSearch.searchByFilter()

The method searches for a person by a filter.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`personSearch.searchByFilter(profileName, filter, scope)`

Arguments

profileName
The name of the person profile to use.

filter LDAP search filter that defines the criteria for returned containers to meet. The filter must be in the format defined by RFC2254.

scope Optional search scope. Use **1** for One Level Scope and **2** for SubTree Scope. One Level Scope is the default scope.

Returns
An array of `DirectoryObjects` representing the results of the search.

Description
This method searches for a person by a filter.

Usage

```
var personSearch = new PersonSearch();
var searchResult1 = personSearch.searchByFilter("Person",
  "(sn=Smith)", 2);

// use default one level scope
var searchResult2 = personSearch.searchByFilter("Person",
    "(sn=Smith)");
```

PersonSearch.searchByURI()

The method finds a person by URI within an organizational container.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`PersonSearch.searchByURI(containerDN, uri)`

Arguments

Container DN
String representing the distinguished name of the parent organizational container.

uri String representing the URI of the person.

Returns

A Person object.

Description

Given the distinguished name of the parent organizational container and the person URI, this method finds the person. If the person is not found, this function returns null. If more than one persons found, this function throws a scripting exception.

Usage

```
var person= (new PersonSearch()).searchByURI(container.dn, uri);
if (person != null) {
Enrole.log("script", "Found " + person.getProperty("cn") );}
```

PostOffice

The object post office object that consolidates notifications.

Availability

IBM Security Privileged Identity Manager 1.0

Provided by

`com.ibm.itim.mail.postoffice.script.PostOfficeExtension`

Methods

getAllEmailMessages()

Obtains the Subject, Text Body, and HTML Body of each individual message contained in an aggregate message.

getEmailAddress()

Contains the email address that is the destination of the aggregate email message.

getPersonByEmailAddress()

Returns the Person that corresponds to the email address specified.

getTopic()

Returns the topic of the aggregated email message.

The `getAllEmailMessages()` extension allows access to the `NotificationMessage` object. Do not call the `getHtmlMessage()` method from a template. This call returns an XHTML version of the notification text. It is not possible to embed XML documents, so a call to this method results in a template execution failure. Use the text body of the original notifications by calling `getMessage()` instead.

PostOffice.getAllEmailMessages()

The message returns an array of `NotificationMessage` objects.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
PostOffice.getAllEmailMessages()
```

Description

This JavaScript extension returns an array of `NotificationMessage` objects for obtaining the Subject, Text Body, and HTML Body of each message in an aggregate message.

Usage An example of how to iterate through the returned array in JavaScript is as follows:

```
Here are the email text bodies fetched using the JavaScript extension:
<JS>
    var msgListIterator =
      PostOffice.getAllEmailMessages().iterator();
    var returnString = "<br />";
    while (msgListIterator.hasNext()) {
        returnString = returnString +
        msgListIterator.next().getMessage() + "<br />";
    }
    return returnString;
</JS>
```

PostOffice.getEmailAddress()

The method returns email address of aggregate email destination.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`PostOffice.getEmailAddress()`

Description
This JavaScript extension returns a String containing the email address that is the destination of the aggregate email message.

Usage `destinationAddress = PostOffice.getEmailAddress();`

PostOffice.getPersonByEmailAddress()

The method returns the Person object that corresponds to this email address.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`PostOffice.getPersonByEmailAddress(`*`String email`*`)`

Description
This JavaScript extension returns the Person object that corresponds to the email address specified.

Usage `targetPerson = PostOffice.getPersonByEmailAddress()`

Examples:

```
targetPerson = PostOffice.getPersonByEmailAddress("user@itim.com");
targetPerson =
 PostOffice.getPersonByEmailAddress(PostOffice.getEmailAddress());
```

PostOffice.getTopic()

The method returns the topic string of the aggregate email.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
`PostOffice.getTopic()`

Description
This JavaScript extension returns a string containing the topic of the aggregated email message.

Usage `topicString = PostOffice.getTopic();`

Process

Represents the IBM Security Privileged Identity Manager workflow process.

Availability

IBM Security Privileged Identity Manager 1.0

Provided by

The Process JavaScript Object in the `WorkflowExtension` returns a Process object. The object represents the current workflow process. The parent processes of the current workflow can be returned by calling `Process.getParent()` recursively, and the parent process is also a Process object.

Properties

Note: Custom result codes are supported in the workflow designer for approval activities.

APPROVED
Approved process summary code. Result code is `AA`.

ESCALATED
Escalated process summary code. Result code is `ES`.

FAILED
Failed process summary code. Result code is `SF`.

PARTICIPANT_RESOLVE_FAILED
Participant resolved failure process summary code. Result code is `PF`.

PENDING
Pending process summary code. Result code is `PE`.

REJECTED
Rejected process summary code. Result code is `AR`.

SUBMITTED
Submitted process summary code. Result code is `RS`.

SUCCESS
Success process summary code. Result code is `SS`.

TIMEOUT
Time out process summary code. Result code is ST.

WARNING
Warning process summary code. Result code is `SW`.

comment
Provides additional information about the process given when defined in the workflow designer.

description
Describes the purpose of the process given when defined in the workflow designer.

id Assigned by the workflow designer to uniquely identify the workflow process within the workflow engine.

name Label given this activity when defined in the workflow designer.

parentId
Uniquely identifies the parent process (if any) that started this process.

requesteeDN
Uniquely identifies the requestee if the requestee is a user in the IBM Security Privileged Identity Manager data store.

requesteeName
Name of the process requestee.

requestorName
The name of the process requestor if the requestor is a user.

requestorType
Categorize the requestor

resultDetail
An application-specific string that provides more detail about the result of the process.

resultSummary
An application-specific string that represents the summary result of the process.

started
Indicates when the process started.

state Code that represents the current state of the process.

subject
Describes the object that is the focal point of the workflow process.

type Code that categorizes the process given when defined in the workflow designer.

Methods

auditEvent()
Create an event in the audit trail specific to the activity.

getActivity()
Returns an activity with the ID and index.

getParent()
Get the parent process (if any) that started this process.

getRootProcess()
Returns the JavaScript Process object that contains information about the root process.

getRootRequesterName()
Returns String of requester name of the root process.

setRequesteeData()
Change the requestee data for the current process.

setResult()
Change the result member of the activity in the current activity.

setSubjectData()
Change the subject data for the current process.

Description
This entity represents the current workflow process is running.

Process.auditEvent()

The method creates an event in the audit trail.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.auditEvent(*event*)

Arguments

event String representing the event to be audited.

Description
This method creates an event in the audit trail specific to the process. The function takes in one parameter that can be any JavaScript object that can be translated into a string for storage. In the audit trail, the event is automatically time stamped.

Usage
```
process.auditEvent("Task completed");
```

Process.comment

The field provides additional information about the process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.comment

Description
This read-only field is a string that provides additional information about the process given when defined in the workflow designer.

Usage
```
x = process.comment;
```

Process.description

The field represents the purpose of the process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.description

Description
This read-only field is a string that describes the purpose of the process when defined in the workflow designer.

Usage
```
x = process.description;
```

Process.getActivity()

The method returns an activity with the ID and index.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.getActivity(*id*, *index*)

Arguments

id Activity ID assigned by the workflow designer.

index Optionally identifies specific activity if there is more than one activity with the ID.

Returns

The associated Activity.

Description

This method returns an activity with the ID and index in the event that there is more than one activity with the ID. This might occur if the activity of the given ID is called multiple times in a loop in the workflow process. If there is no activity with the ID and index, this function returns null. If the optional index is not specified and if there is more than one activity with the ID, the first activity with the ID is returned.

Usage

```
theFirstActivity = process.getActivity("id1", 3);
theActivityName = theFirstActivity.name;

theSecondActivity = process.getActivity("id2");
theActivityName = theSecondActivity.name;
```

Process.getParent()

The method returns the parent process (if any) that started this process.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

process.getParent()

Returns

The parent Process. If there is no parent, a null is returned.

Description

This method returns the parent process (if any) that started this process.

Usage

```
parent = process.getParent();
parentName = parent.name;
```

Process.getRootProcess()

The method returns the root process (if any) that started this process.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
process.getRootProcess()
```

Returns

The root process. If there is no root process, a null is returned.

Description

This method returns the root process (if any) of this process.

Usage

```
root = process.getRootProcess();
rootName = root.name;
```

Process.getRootRequesterName()

The method returns the root requester name.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.getRootRequesterName()

Description
This method returns the root requester name of the workflow process initiator.

Usage `rootRequester = process.getRootRequesterName();`

Process.guid

The generated unique identifier assigned to the process at runtime.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.guid

Description
This read-only field is a String of the generated unique identifier for the workflow process in the workflow engine.

Usage `x = process.guid;`

Process.getSubProcesses()

The method returns the subordinate processes (if any) of the process.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
process.getSubProcesses()
```

Returns

The subordinate processes. If there are no subordinate processes, an empty array is returned.

Description

This method returns the subordinate processes (if any) of this process.

Usage

```
var out = "subprocesses of the process: \n";

function traverse(p, prefix) {
  var subProcesses = p.getSubProcesses();
  prefix += "/" + p.name;
```

```
  out += prefix + ": " + p.id + " type: " + p.type + " resultSummary: " + p.resultSummary + "\n";
  for (var i = 0; i < subProcesses.length; i++) {
      traverse(subProcesses[i], prefix);
  }
}

traverse(process, "");
activity.auditEvent(out);
```

Process.id

The generated unique identifier assigned to the process at runtime.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.id

Description
This read-only field is a string of the generated unique identifier for the workflow process in the workflow engine.

Usage `x = process.id;`

Process.name

The label assigned to the process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.name

Description
This read-only field is a string assigned by the workflow designer to label this process.

Usage `x = process.name;`

Process.parentId

The field uniquely identifies the parent process that started this process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.parentId

Description
This read-only field is a string representation of the long integer that uniquely identifies the parent process (if any) that started this process.

Usage `x = process.parentId;`

Process.requesteeDN

The field uniquely identifies the requestee if the requestee is a user in the IBM Security Privileged Identity Manager data store.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.requesteeDN

Description

This read-only field is a string that uniquely identifies the requestee if the requestee is a user in the IBM Security Privileged Identity Manager data store. Not all requestees are users (that is, the process can act on a policy, not a user directly), so this member can be empty.

Usage

```
x = process.requesteeDN;
```

Process.requestorDN

The field specifies the distinguished name of the process requester, if the requester is a user in the IBM Security Privileged Identity Manager data store.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
process.requestorDN
```

Description

This read-only field is a string that represents the distinguished name of the process requester. This string is displayed only if the requester is a user in the IBM Security Privileged Identity Manager data store. Not all requesters are users (that is, the process can act on a policy, not a user directly), so this member can be empty.

Usage

```
if (process.requestorType == "U")
x = process.requestorDN;
```

Process.requesteeName

The field represents the name of the process requestee as a string.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
process.requesteeName
```

Description

This read-only field is a string that provides the name the requestee if the requestee is a user in the IBM Security Privileged Identity Manager data store. Not all requestees are users (that is, the process can act on a policy, not a user directly), so this member can be empty.

Usage

```
x = process.requesteeName;
```

Process.requestorName

The field represents the name of the process requester if the requester is a user.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

```
process.requestorName
```

Description

This read-only field is a string that represents the name of the process requester if the requester is a user.

Usage

```
if (process.requestorType == "U")
  x = process.requestorName;
```

Process.requestorType

The field categorize the requestor.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.requestorType

Description
This read-only field is a string that categorizes the requestor. The potential categories, or types, are:

- **U** for user
- **S** for the workflow engine
- **P** for the system

Usage

```
x = process.requestorType;
if (x == "U")
  ...
else if (x == "S")
  ...
else if (x == "P")
  ...
```

Process.resultDetail

The field details about the result of the process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process.resultDetail

Description
This read-only field is an application-specific string that provides more detail about the result of the process.

Usage `x = process.resultDetail;`

Process.resultSummary

The field represents the summary of the result of the process.

Availability
IBM Security Privileged Identity Manager 1.0

Description
This read-only field is an application-specific string that provides a summary of the result of the process.

Usage `x = process.resultSummary;`

Process.setRequesteeData()

The method changes the requestee data for the current process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis

process.setRequesteeData(*person*)

Arguments

person

DirectoryObject representing the new requestee.

Description

This method changes the requestee data for the current process. It is not supported for a process that is not the current process. It not only updates the current process in the script, but also in the workflow engine. The requesteeData argument contains a person distinguished name or a collection of strings from which the requestee data can be extracted.

Usage *process*.setRequesteeData(*person*);

Process.setResult()

The method changes the result member of the process.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

process.setResult(*summary*, *detail*)

Arguments

summary

String code that represents the result summary.

detail String representing the result details.

Description

This method changes the result member of the process in the current process. It is supported for current activities in the current workflow process. The result is composed by an application-specific summary code, and optional more detailed application-specific description. The summary code can indicate a success or failure. This summary code is stored as the resultSummary member locally and updated in the relevant data in the workflow engine. The detail is stored as the resultDetail member locally and updated in the relevant data in the workflow engine.

Usage

```
process.setResult(process.FAILED, "Unable to connect to resource");
```

Process.setSubjectData()

The method changes the subject data for the current process.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

process.setSubjectData(*person*)

Arguments

person

DirectoryObject representing the new subject.

Description

This method changes the subject data for the current process. It is not supported for a process that is not the current process. It not only updates

the current process in the script, but also in the workflow engine. The `subjectData` argument contains a person distinguished name or a collection of strings from which the subject data can be extracted.

Usage *process*`.setSubjectData(`*person*`);`

Process.started

The field represents the JavaScript date that indicates when the process started.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process`.started`

Description
This read-only field is code string that represents the JavaScript Date that indicates when the process started.

Usage

```
x = process.started;
```

Process.state

The field represents the current state of the process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process`.state`

Description
This read-only field is code string that represents the current state of the process. The state can have the following values:

- `R` for running
- `I` for not started
- `T` for terminated
- `A` for aborted
- `S` for suspended
- `C` for completed
- `B` for bypassed

Usage

```
if (process.state == "S") {
  ...
}
```

Process.subject

The field represents the object that is the focal point of the workflow process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis
process`.subject`

Description
This read-only field is code string that describes the object that is the focal

point of the workflow process. This string can be an identity in the system, an account, a policy, or another object.

Usage `x = process.subject;`

Process.type

The field represents the type of process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis

`process.type`

Description
This read-only field is code string that categorizes the process when defined in the workflow designer.

Usage `x = process.type;`

ProcessData

The object represents the workflow process data entity.

Availability

IBM Security Privileged Identity Manager 1.0
Workflow context

Provided by
com.ibm.itim.workflow.script.WorkflowExtension

Methods

get() Returns a JavaScript object that represents the value of the relevant data item.

set() Changes the value of the relevant data item.

Description
Each workflow process has a set of relevant data, or process specific parameters, which can be read or changed from within a workflow script. The name and syntax of these parameters, or relevant data items, are defined in the workflow designer, and are typically specific to the workflow process purpose. For example, when adding a user, an object that holds all the attributes of the new user can be a relevant data item. However, when deleting a user, the only needed relevant data item can be the distinguished name of the user to delete.

Each relevant data item will be represented in the workflow script as a variable with the same relevant data ID as defined in the workflow designer.

ProcessData.get()

The method changes the subject data for the current process.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis

`processData.get()`

Returns

Returns a JavaScript object that represents the value of the relevant data item.

Description

This method returns a JavaScript object that represents the value of the relevant data item. There is a variable present for each relevant data item in the context of script. For performance reasons, the values are not retrieved from the workflow engine until the script specifically requests the values with this call. The returned JavaScript object is in the same syntax as defined in the workflow designer.

Usage `dn = subjectDN.get();`

ProcessData.set()

The method changes the value of the relevant data item.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

processData`.set(value)`

Arguments

value Value to use to update the relevant data item.

Description

This method changes the value of the relevant data item. It not only updates the relevant data item in the script, but also in the workflow engine. The new value is a parameter to the function. The new value must be compatible with the syntax of the relevant data item as defined in the workflow designer. For example, if the relevant data item is an integer, the value cat would not be a valid parameter to this function.

Usage `processData.set("engineering");`

Reminder

An activity to-do item reminder informs the participant that the IBM Security Privileged Identity Manager requires user action.

Availability

IBM Security Privileged Identity Manager 1.0
Reminder context

Provided by

`com.ibm.itim.script.extensions.ReminderExtension`

Methods

Reminder.getOriginalSubject()

This method returns the subject of the original notification sent when the work item was first assigned.

Reminder.getXhtmlBody()

This method returns the XHTML body of the original notification sent when the work item was first assigned.

Reminder.getTextBody()
This method returns the text body of the original notification sent when the work item was first assigned.

Reminder.getRemindersSent()
This method returns the number of reminders previously sent.

Reminder.getEscalationTime()
This method returns a string that contains the date and time when the work item is escalated unless acted upon.

Reminder.getEscalationDate()
This method returns a Date containing the date and time when the work item is escalated unless acted upon.

Description
An activity to-do item reminder informs the participant that IBM Security Privileged Identity Manager requires user action.

Role

The object represents the role associated with a provisioning operation.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.model.RoleModelExtension`

Constructor
new `Role(dn)`

Returns
A new Role object that represents the Role with the given DN.

Methods

getAssignmentAttributes()
Returns an array of assignment attribute names. Returns an empty array if no assignment attribute exists.

getAllAssignmentAttributes()
Returns an array of `RoleAssignmentAttribute` objects containing assignment attribute name, role name, and role DN. Returns an empty array if no assignment attribute exists. Returns the role assignment attributes of the whole role hierarchy.

getOwner()
Returns an array of `DirectoryObjects` that represent any Person that has an Owner relationship with this role.

setAssignmentAttributes()
Sets role assignment attributes of the role.

Inherits from
`DirectoryObject`

Synopsis
```
role.dn;
```

Description
The role object is available in the context of a provisioning policy.

Note: For more information on role assignment attributes, see Defining assignment attributes when creating a role.

Role.getAssignmentAttributes()

The method returns an array of assignment attribute names. Returns an empty array if no assignment attribute exists.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

```
Role.getAssignmentAttributes()
```

Arguments

None

Description

This method is defined on the Role object and returns an array of assignment attribute names. The method returns an empty array if no assignment attribute exists.

Usage

```
var role = new Role(roleDN);

//get assignment attributes of the role
var attributeList = role.getAssignmentAttributes();
if (attributeList.length == 0) {
    Enrole.log("script", "No assignment attribute for this role: "
        + role.name);
    return;
}

// print out role assignment attribute name.
for (var i=0; i < attributeList.length; i++) {
    var attrName = attributeList[i];
    Enrole.log("script","attribute name-----: "+ attrName);
}
```

Role.getAllAssignmentAttributes()

The method returns an array of `RoleAssignmentAttribute` objects that contain the assignment attribute name, role name, and role DN. Returns an empty array if no assignment attribute exists. Returns the role assignment attributes of the whole role hierarchy.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

```
Role.getAllAssignmentAttributes()
```

Arguments

None

Description

This method is defined on the Role object and returns an array of `RoleAssignmentAttribute` objects. The array contains the assignment of the attribute name, role name, and role DN of the role. The method returns an empty array if no assignment attribute exists. It returns the role assignment attributes of the whole role hierarchy.

Usage

```
var role = new Role(roleDN);
//get assignment attributes of the role
var attributeList = role.getAllAssignmentAttributes();
if (attributeList.length == 0) {
    Enrole.log("script", "No assignment attribute for this role: "
        + role.name);
    return;
}

// print out role assignment attribute name.
for (var i=0; i < attributeList.length; i++) {
    var roleAtr = attributeList[i];
    Enrole.log("script","attribute name-----: "+ roleAtr.getName());
}
```

Role.getOwner()

The method returns an array of `DirectoryObjects` that represents any Person that has an Owner relationship with this role.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

`Role.getOwner()`

Returns

Array of `DirectoryObjects` that represents the owners of this Role or null if there are no owners.

Usage `var owners = role.getOwner();`

Role.setAssignmentAttributes()

The method sets role assignment attributes of the role.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

`Role.setAssignmentAttributes(String[] attributeNames)`

Arguments

attributeNames

The array of assignment attribute names of the role. If an empty array is specified, all assignment attributes for the role are removed.

Description

This method is defined on the Role object and sets the role assignment attributes for a role.

Usage

```
var roleDN = roles[0];
var role = new Role(roleDN);
var roleAtr = new Array();
roleAtr[0] = "creditlimit";
//set assignment attribute names
role.setAssignmentAttributes(roleAtr);
```

RoleAssignmentAttribute

The object represents the role assignment attribute associated with a role.

Availability
IBM Security Privileged Identity Manager 1.0

Methods

getName()
Returns the attribute name associated with the role assignment attribute object.

getRoleName()
Returns the name of the role. Returns an empty string if there is no name associated with the role assignment attribute object.

getRoleDN()
Returns the DN of the role. Returns an empty string if there is no DN associated with the role assignment attribute object.

Description
The `RoleAssignmentAttribute` object associated with the role assignment attribute.

RoleAssignmentAttribute.getName()

The method returns the name of the assignment attribute.

Availability
IBM Security Privileged Identity Manager 1.0.

Synopsis
```
RoleAssignmentAttribute.getName()
```

Arguments
None

Returns
The name of the assignment attribute.

Description
Returns the name of the assignment attribute that is defined on the role.

Usage
```
var role = new Role(roleDN);
    //get assignment attributes of the role
    var attributeList = role.getAllAssignmentAttributes();
    if (attributeList.length == 0) {
        Enrole.log("script", "No assignment attribute for this role: "
            + role.name);
        return;
    }

    // print out role assignment attribute name.
    for (var i=0; i < attributeList.length; i++) {
        var roleAtr = attributeList[i];
        Enrole.log("script","attribute name-----: "+ roleAtr.getName());
    }
```

RoleAssignmentAttribute.getRoleName()

The method returns the name of the role that has the assignment attribute defined.

Availability
IBM Security Privileged Identity Manager 1.0.

Synopsis
```
RoleAssignmentAttribute.getRoleName()
```

Arguments

None

Returns

The name of the role that has the assignment attribute defined.

Description

Returns the name of the role that has the assignment attribute defined.

Usage

```
var role = new Role(roleDN);
    //get assignment attributes of the role
    var attributeList = role.getAllAssignmentAttributes();
    if (attributeList.length == 0) {
        Enrole.log("script", "No assignment attribute for this role: "
            + role.name);
        return;
    }

    // print out all role names.
    for (var i=0; i < attributeList.length; i++) {
        var roleAtr = attributeList[i];
        Enrole.log("script","role name-----: "+ roleAtr.getRoleName());
    }
```

RoleAssignmentAttribute.getRoleDN

The method returns the distinguished name of the role that defines the assignment attributes.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

```
RoleAssignmentAttribute.getRoleDN()
```

Arguments

None

Returns

The distinguished name of the role that defines the assignment attributes.

Description

Returns the distinguished name of the role that defines the assignment attributes.

Usage

```
var role = new Role(roleDN);
    //get assignment attributes of the role
    var attributeList = role.getAllAssignmentAttributes();
    if (attributeList.length == 0) {
        Enrole.log("script", "No assignment attribute for this role: "
            + role.name);
        return;
    }

    // print out the distinguished name of the role that defines
    // assignment attributes.
    for (var i=0; i < attributeList.length; i++) {
        var roleAtr = attributeList[i];
        Enrole.log("script","define role DN-----: "+ roleAtr.getRoleDN());
    }
```

RoleAssignmentObject

The `RoleAssignmentObject` class is a DataObject class for role assignment data.

This class holds the assignment data that are associated with the defined role and the assigned role. The defined role is the role that holds a list of assignment attributes. The assigned role is the role to which the person is assigned.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.model.RAObjectModelExtension`

Constructors

new RoleAssignmentObject(RoleAssignmentObject assignmentObject)
Arguments:

assignmentObject
`RoleAssignmentObject` that is wrapped inside the `RoleAssignmentObject`.

new RoleAssignmentObject(String assignedRoleDN, String definedRoleDN)
Arguments:

assignedRoleDN
The String format of the distinguished name for the assigned role.

definedRoleDN
The String format of the distinguished name for the defined role.

Methods

addProperty()
Adds the values for specified assignment attribute.

getAssignedRoleDN()
Returns the distinguished name string for the role to which the person is assigned.

getDefinedRoleDN()
Returns the distinguished name string for the role in which the assignment attribute is defined.

getChanges()
Returns the changes made to this `RoleAssignmentObject`.

getProperty()
Returns the values of the property specified by the assignment attribute name.

getPropertyNames()
Returns a list of role assignment attribute names.

removeProperty()
Removes the values for the specified assignment attribute name.

setProperty()
Sets the values for a specified assignment attribute.

Description

RoleAssignmentObject contains the role assignment data, including the assigned role DN, the defined role DN and attribute values.

RoleAssignmentObject.getAssignedRoleDN()

The method returns the distinguished name string for the role to which a person is assigned.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

roleAssignmentObject.getAssignedRoleDN()

Arguments

None

Returns

The distinguished name string for the role to which a person is assigned.

Description

This method returns the distinguished name string for the role to which a person is assigned.

Usage

```
var assignedRoleDN = "globalid=111";
var definedRoleDN = "globalid=222";
var assignmentObj = new RoleAssignmentObject(assignedRoleDN, definedRoleDN);

var assignedRoleDN2 =  assignmentObj.getAssignedRoleDN();
```

RoleAssignmentObject.getDefinedRoleDN()

The method returns the distinguished name string for the role in which the assignment attribute is defined.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

roleAssignmentObject.getDefinedRoleDN()

Arguments

None

Returns

Returns the distinguished name string for the role in which the assignment attribute is defined.

Description

This method returns the distinguished name string for the role to which the person is assigned.

Usage

```
var assignedRoleDN = "globalid=111";
var definedRoleDN = "globalid=222";
var assignmentObj = new RoleAssignmentObject(assignedRoleDN, definedRoleDN);

var definedRoleDN2 =  assignmentObj.getDefinedRoleDN();
```

RoleAssignmentObject.addProperty()

Use this method to add the values for specified assignment attribute.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

RoleAssignmentObject.addProperty(*name*, *value*)

Arguments

name String representing the name of the assignment attribute to be added.

value The value to be added.

Description

This method changes the value of the specified assignment attribute or adds the specified assignment attribute if it does not exist. This change is made locally to the script environment, not to the data store.

Usage

```
// Create assignment object with assigned role dn and defined role dn.
var assignmentObj = new RoleAssignmentObject("eruid=1111,dc=com",
    "eruid=2222,dc=com");
// Add some assignment attribute with values.
assignmentObj.addProperty("attr1", ["attr1val1","attr2val1"]);
assignmentObj.addProperty("attr2", ["attr2val1"]);
assignmentObj.addProperty("attr2", ["attr2val2"]);
```

RoleAssignmentObject.getChanges()

The method returns the changes made to an entity.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

RoleAssignmentObject.getChanges()

Returns

An array of change objects. If there are no changes, an empty array is returned. Each element in the array is an AttributeChangeOperation.

Description

This method returns the changes made to the entity. These changes are represented by change objects with the following members:

attr String name of the attribute that is being changed.

op An integer that identifies the type of change that is being made. The enumerated values are 1 for add, 2 for replace, and 3 for remove.

values An array of objects that can be either added, removed, or replaced.

The changes are returned as an array of these change objects. If there are no changes, an empty array is returned.

Usage

```
changes = assignmentObject.getChanges();
for (i = 0; i < changes.length; i++) {
  name = changes[i].attr;
  if (changes[i].op == 1) {
    ...
  } else if (changes[i].op == 2) {
    ...
```

```
  } else {
    ...
  }
};
```

RoleAssignmentObject.getProperty()

The method returns the values of the assignment attribute specified by the given name.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

RoleAssignmentObject.getProperty(*name*)

Arguments

name String representing the name of the assignment attribute to return.

Returns

The array of strings that represents the values for an assignment attribute. If the specified assignment attribute does not exist, an empty array is returned.

Description

This method returns the values of the assignment attribute specified by the given name. If the specified assignment attribute does not exist, an empty array is returned.

Usage

```
// create assignment object with assigned role dn and defined role dn.
var assignmentObj = new RoleAssignmentObject("eruid=1111,dc=com",
    "eruid=2222,dc=com");
assignmentObj.addProperty("attr1", ["attr1val1", "attr1val2"]);

// get assignment attribute values for attr1.
var attrValues = assignmentObj.getProperty("attr1");
var attrValuesStr = "";
for (var j=0; j<attrValues.length; j++) {
  attrValuesStr += attrValues[j] + ", ";
Enrole.log("script", "The values for attr1:" + attrValuesStr);
```

RoleAssignmentObject.getPropertyNames()

The method returns a list of assignment attributes.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

RoleAssignmentObject.getPropertyNames()

Returns

An array of strings.

Description

This method returns a list of assignment attributes as an array of strings.

Usage `properties = RoleAssignmentObject.getPropertyNames();`

RoleAssignmentObject.removeProperty()

The method removes the assignment attribute specified by the given name.

Availability
IBM Security Privileged Identity Manager 1.0.

Synopsis
RoleAssignmentObject.removeProperty(*name*)

Arguments

name String representing the name of the assignment attribute to remove.

Description
This method removes the specified assignment attribute. This change is made locally to the script environment, not to the data store.

Usage *RoleAssignmentObject*.removeProperty("assignmentAttr1");

RoleAssignmentObject.setProperty()

The method sets the value of the specified assignment attribute.

Availability
IBM Security Privileged Identity Manager 1.0.

Synopsis
RoleAssignmentObject.setProperty(*name*, *value*)

Arguments

name String representing the name of the assignment attribute to be created or modified.

value Specifies the value to which the assignment attribute is set.

Description
This method changes the value of the specified assignment attribute, or adds the specified assignment attribute if it does not exist. This change is made locally to the script environment, not to the data store.

Usage *RoleAssignmentObject*.setProperty("attr1",["val1","val2"]);

RoleSearch

The object searches for a role.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
com.ibm.itim.script.extensions.model.RoleModelExtension

Constructor
new RoleSearch()

Returns
The newly created and initialized role search object.

Methods

searchByName()
Search for a role by name.

searchByURI()
Search for a role by URI within an organizational container.

RoleSearch.searchByName()

The method searches for a role by a name.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

RoleSearch.searchByName(*name*)

Arguments

name The role name to use as the basis for the search.

Returns

Array of DirectoryObjects that represents a role.

Description

Given the name of a role, locate the Role entity. Will return null if there is not exactly one matching role.

Usage

```
// Given the name of a role, see if it exists and log its
// description
var roles = (new RoleSearch()).searchByName("testRole");
if (roles.length >= 1) {
  if (roles[0].getProperty("errolename")[0] == "testRole") {
   Enrole.log("script", "The Role "+ roles[0].getProperty("errolename")[0] +
    "has Description :" + roles[0].getProperty("description")[0]);
  }
}
```

RoleSearch.searchByURI()

The method finds a role by URI in an organizational container.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

```
RoleSearch.searchByURI(containerDN, uri)
```

Arguments

Container DN

String representing the distinguished name of the organizational container.

uri String representing the URI of the role.

Returns

A Role object

Description

Given the distinguished name of the organizational container and the role URI, this method finds the container. If the role is not found, this function returns null. If more than one role is found, this function throws a scripting exception.

Usage

```
var role = (new RoleSearch()).searchByURI(container.dn, uri);
if (role != null) {
Enrole.log("script", "Found " + role.getProperty("errolename") );}
```

Service

The object represents the service associated with a provisioning operation.

Availability
IBM Security Privileged Identity Manager 1.0

Provided by
`com.ibm.itim.script.extensions.model.ServiceModelExtension`

Constructor
`new Service(dn)`

Returns
A new Service object that represents the Service with the DN.

Inherits From
`DirectoryObject`

Synopsis
```
service.dn;
```

Description
The service object is available in the context of a Provisioning Policy and Service Selection Policy.

ServiceSearch

Use the object to provide searching capability for IBM Security Privileged Identity Manager services.

Availability

IBM Security Privileged Identity Manager 1.0
Provisioning Policy context
Service Selection Policy context

Provided by
`com.ibm.itim.script.extensions.model.ServiceModelExtension`

Methods

searchByFilter()
Search for a service by a filter.

searchByName()
Search for a service by a name.

searchByURI()
Search for a service by URI in an organizational container.

searchForClosestToPerson()
Search for the closest Service to a person.

Description
This object is used to provide searching capability for IBM Security Privileged Identity Manager services.

ServiceSearch.searchByFilter()

The method searches for a service by a filter.

Availability
IBM Security Privileged Identity Manager 1.0

Synopsis

ServiceSearch.searchByFilter(*filter*, *scope*)

Arguments

filter LDAP search filter that defines the criteria for returned containers to meet. The filter must be in the format defined by RFC2254.

scope Optional search scope. Use 1 for One Level Scope and 2 for SubTree Scope. One Level Scope is the default scope.

Returns

An array of `DirectoryObjects` representing the results of the search.

Description

This method searches for a service by a filter.

Usage

```
searchResult1 =
 ServiceSearch.searchByFilter("(erntlocalservername=*srv)", 2);

// use default one level scope, put statement on one line

searchResult2 =
 ServiceSearch.searchByFilter("(erntlocalservername=*srv)");
```

ServiceSearch.searchByName()

The method searches for a service by name.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

ServiceSearch.searchByName(*name*, *profileName*, *scope*)

Arguments

name The service name, provided as a string, to use as the basis for the search.

profileName

Optional profile name, provided as a string. The profile name of the service to use as the basis for the search.

scope Optional search scope, provided as an int. Use **1** for One Level Scope and **2** for Scope. One Level Scope is the default scope. When you use this method in workflow JavaScripts, set the scope parameter to SubTree because the logical search context is limited to the tenant above the default organization. In this context, setting the scope to One Level Scope returns empty results during a search because there are no services at the tenant level.

Returns

An array of `DirectoryObjects` representing the results of the search.

Description

This method searches for a service by a name.

Usage

```
searchResult1 = ServiceSearch.searchByName("US NT Service", 2);

// use default one level scope
searchResult2 = ServiceSearch.searchByName("US NT Service");
```

ServiceSearch.searchByURI()

The method finds a service by URI in an organizational container.

Availability

IBM Security Privileged Identity Manager 1.0.

Synopsis

```
ServiceSearch.searchByURI(containerDN, uri)
```

Arguments

Container DN
String representing the distinguished name of the organizational container.

uri String representing the URI of the service.

Returns

A Service object

Description

Given the distinguished name of the organizational container and the service URI, this method finds the service. If the service is not found, this function returns null. If more than one service is found, this function throws a scripting exception.

Usage

```
var service = (new ServiceSearch()).searchByURI(container.dn, uri);
if (service != null) {
Enrole.log("script", "Found " + service.getProperty("erservicename") );}
```

ServiceSearch.searchForClosestToPerson()

The method searches for a service closest to a person.

Availability

IBM Security Privileged Identity Manager 1.0

Synopsis

ServiceSearch.searchForClosestToPerson(*person*, *profileName*)

Arguments

person
The `DirectoryObject` representing a person to use as the basis for the search.

profileName
Optional service profile name.

Returns

An array of `DirectoryObjects` representing the results of the search.

Description

This method searches for a service closest to a person.

Usage

```
//Search for AIX service closest to the person.
searchResult1 = ServiceSearch.searchForClosestToPerson(subject,
    "PosixAixProfile");

//Search for any service closest to the person.
searchResult2 = ServiceSearch.searchForClosestToPerson(subject);
```

Chapter 6. Application identity commands

Use the application identity command-line tools to register application instances, capture application fingerprints, reconfigure application services, and get credentials from the server.

In a command prompt, type:

```
java -jar ibmappid.jar
```

Note: Use this command for an interactive prompt.

or

```
java -jar ibmappid.jar <command> [options]
```

where

`<command>` is the command that you want to run.

`[options]` are the optional switches for the command.

Note: To display help with a specific command or option, specify the `-?` parameter. For example:`<command> [options] -?`

install-certificate

Installs the certificate on the workstation if its not already installed.

Syntax

```
install-certificate [options]
```

Options

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-s,--server *<server url>*
URL of the IBM Security Privileged Identity Manager Server. For example: `pimhost:9443`

-v,--verbose
Verbose output displays additional information.

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Examples

Example: Installs the certificate on the workstation without prompts.
`install-certificate -s pimhost:9443 -x`

Example: Installs the certificate on the workstation interactively.
`install-certificate -s pimhost:9443`

register-first-instance

Registers the first application instance to capture the application fingerprint.

The user must be a member of the Privileged Administrator group.

Syntax

register-first-instance [options]

Options

-a,--application-name <*application name*>
Name of the application.

-b,--binary path <*path*>
Path to the JAR file that contains the class that uses `AppIDManager`.

For example: `c:\hrapp\hrapp.jar`

This parameter applies to Java applications. (**--application-type 1**)

-d,--instance description <*description*>
The application instance description.

Default: empty

-g,--group-id <*id*>
Group name for separating two application instances that have the same fingerprint. For example, if there are scripts for tuning databases and scripts for backing up databases on the same host, you can categorize the scripts into two application instances. You might categorize the scripts by assigning group labels such as `db_tuning_scripts` and `db_backup_scripts`.

Default: empty

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-l,--class-name <*name*>
Full name of the class using `AppIDManager`.

This parameter applies to Java applications (**--application-type 1**).

-n,--instance-name <*application instance name*>
Name of the application instance.

-o,--os-user <*path*>
The operating system user name, that the application instance will run under.

For example: `test\user1`

Default: current user

Note: If you are registering an application instance for a different user, or a user that belongs to a domain, the value you provide during registration must match the output of the `whoami` command for that user.

For example, you are registering an instance for a user `User1` who is part of domain `test.example.com`.

Run `whoami` while logged in as `User1`.

If `whoami` returns `test\user1`, this is the value that you must use for the operating system user.

-p,--password *<password>*
The login password.

-s,--server *<server url>*
URL of the IBM Security Privileged Identity Manager Server. For example: `pimhost:9443`

-t,--application-type *<application type>*
The application type. Enter one of the following options:

- `1` for Java applications.
- `2` for scripts.
- `3` for data sources.

-u,--username *<user name>*
The login user name.

-v,--verbose
Verbose output displays additional information.

-w,--workspace *<path>*

Path of the workspace to store SSL certificates and tokens.

Default: parent folder of `ibmappid.jar`

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Examples

Example: Register a Java application that integrates with the App ID SDK

```
register-first-instance --server pimhost:9443 --username valerie --password secret
  --application-type 1 --application-name HRApp --instance-name hrapp@host3
  --binary-path c:\hrapp\hrapp.jar --class-name example.hrapp
  --instance-description "HR Application on Host3"
  --workspace ./workspace -x
```

Example: Register a script that uses the App ID command-line tool.

```
register-first-instance --server pimhost:9443 --username valerie --password secret
  --application-type 2 --application-name db_backup_script
  --instance-name db_backup_script@host3
  --instance-description "Backup Script on Host3"
  --workspace ./workspace -x
```

Example: Register a data source for a Java EE application that uses an application identity

```
register-first-instance --server pimhost:9443 --username valerie --password secret
  --application-type 3 --application-name ds_hrdb
  --instance-name ds_hrdb@host3 --instance-description "HR DB on Host3"
  --workspace ./workspace -x
```

register-additional-instance

Registers additional instances of an application.

The user must be a member of the Privileged Administrator group.

Syntax

register-additional-instance [options]

Options

-a,--application-name <*application name*>
Name of the application.

-b,--binary path <*path*>
Path to the JAR file that contains the class that uses `AppIDManager`.

For example: `c:\hrapp\hrapp.jar`

This parameter applies to Java applications. (**--application-type 1**)

-d,--instance description <*description*>
The application instance description.

Default: empty

-g,--group-id <*id*>
Group name for separating two application instances that have the same fingerprint. For example, if there are scripts for tuning databases and scripts for backing up databases on the same host, you can categorize the scripts into two application instances. You might categorize the scripts by assigning group labels such as `db_tuning_scripts` and `db_backup_scripts`.

Default: empty

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-l,--class-name <*name*>
Full name of the class using `AppIDManager`.

This parameter applies to Java applications (**--application-type 1**).

-n,--instance-name <*application instance name*>
Name of the application instance.

-o,--os-user <*path*>
The operating system user name, that the application instance will run under.

For example: `test\user1`

Default: current user

Note: If you are registering an application instance for a different user, or a user that belongs to a domain, the value you provide during registration must match the output of the `whoami` command for that user.

For example, you are registering an instance for a user `User1` who is part of domain `test.example.com`.

Run `whoami` while logged in as `User1`.

If `whoami` returns `test\user1`, this is the value that you must use for the operating system user.

-p,--password *<password>*
The login password.

-s,--server *<server url>*
URL of the IBM Security Privileged Identity Manager Server. For example: `pimhost:9443`

-t,--application-type *<application type>*
The application type. Enter one of the following options:

- `1` for Java applications.
- `2` for scripts.
- `3` for data sources.

-u,--username *<user name>*
The login user name.

-v,--verbose
Verbose output displays additional information.

-w,--workspace *<path>*

Path of the workspace to store SSL certificates and tokens.

Default: parent folder of `ibmappid.jar`

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Example: Registers a new application instance and prompts for additional information about the application instance

```
register-additional-instance --server pimhost:9443 --username valerie --password secret
```

Example: Registers a Java application instance that is hosted on a new workstation

```
register-additional-instance --server pimhost:9443 --username valerie --password secret
  --application-type 1 --application-name HRApp --instance-name hrapp@host3
  --binary-path c:\hrapp\hrapp.jar --class-name example.hrapp
  --instance-description "HR Application on Host4"
  --workspace ./workspace --silent
```

Example: Registers a script application instance that is hosted on a new workstation

```
register-additional-instance --server pimhost:9443 --username valerie --password secret
  --application-type 2 --application-name HRApp
  --instance-name db_backup_script@host4
  --instance-description "HRApp DB Backup Script on Host4"
  --workspace ./workspace --silent
```

Example: Register a data source application instance for a Java EE application that uses an application identity

```
register-additional-instance --server pimhost:9443 --username valerie --password secret
  --application-type 3 --application-name HRApp
  --instance-name ds_hrdb@host4 --instance-description "HR DB on Host4"
  --workspace ./workspace --silent
```

get-credential

Gets the specified credentials from the credential vault.

The user must be a member of the Privileged Administrator group. The application instance must be registered.

Syntax

get-credential [options]

Options

-c,--credential-username <*name*>

The user name of the credential to be retrieved.

Default: empty (any credential is accepted)

-g,--group-id <*id*>
Group name for separating two application instances that have the same fingerprint. For example, if there are scripts for tuning databases and scripts for backing up databases on the same host, you can categorize the scripts into two application instances. You might categorize the scripts by assigning group labels such as `db_tuning_scripts` and `db_backup_scripts`.

Default: empty

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-n,--instance-name <*application instance name*>
Name of the application instance.

-r,--resource-uid <*resourceuri*>
Resource UID or alias from which IBM Security Privileged Identity Manager can get credentials.

-s,--server <*server url*>
URL of the IBM Security Privileged Identity Manager Server. For example: `pimhost:9443`

-v,--verbose
Verbose output displays additional information.

-w,--workspace <*path*>

Path of the workspace to store SSL certificates and tokens.

Default: parent folder of `ibmappid.jar`

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?

Shows help for a specific command or option. For example: `<command> -?`

Example

Retrieves credentials from the vault for the hrapp_host3 application instance without any prompts.

```
get-credential -s pimhost:9443 -r ldap.example.com -n hrapp_host3 -w default
  -c default -x
```

Output:

```
rootuser
secretpassword
```

configure-services

Applies the configuration to the services.

Syntax

configure-services [options]

Options

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-n,--service-manager <*service manager*>
Name of the service management agent.

-s,--server <*server url*>
URL of the IBM Security Privileged Identity Manager Server. For example: pimhost:9443

-v,--verbose
Verbose output displays additional information.

-w,--workspace <*path*>

Path of the workspace to store SSL certificates and tokens.

Default: parent folder of ibmappid.jar

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Examples

Example: Reconfigures application services silently.

```
configure-service --server pimhost:9443
  --service-manager HRAppService --workspace C:\appid --install-certificate -x
```

Example: Reconfigures application services silently. Options specified in short form.

```
configure-service -s pimhost:9443
  -n HRAppService -w C:\appid -i -x
```

register-service-manager

Registers a new service management agent. The service management agent manages services for a group of endpoints.

Syntax

register-service-manager [options]

Options

-i,--install-certificate
Installs the certificate, if the certificate is not already installed.

-n,--service-manager <*service manager*>
Name of the service management agent.

-o,--os-user <*path*>
The operating system user name, that the application instance will run under.

For example: `test\user1`

Default: current user

Note: If you are registering an application instance for a different user, or a user that belongs to a domain, the value you provide during registration must match the output of the `whoami` command for that user.

For example, you are registering an instance for a user `User1` who is part of domain `test.example.com`.

Run `whoami` while logged in as `User1`.

If `whoami` returns `test\user1`, this is the value that you must use for the operating system user.

-p,--password <*password*>
The login password.

-s,--server <*server url*>
URL of the IBM Security Privileged Identity Manager Server. For example: `pimhost:9443`

-u,--username <*user name*>
The login user name.

-v,--verbose
Verbose output displays additional information.

-w,--workspace <*path*>

Path of the workspace to store SSL certificates and tokens.

Default: parent folder of `ibmappid.jar`

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Examples

Example: Registers a service management agent on the workstation silently. Installs a certificate if required.

```
register-service-manager --server pimhost:9443 --username valerie
--password secret --service-manager HRAppService
--workspace C:\appid --install-certificate --os-user
valerie -x
```

Example: Registers a service management agent with options specified in short form.

```
register-service-manager -s pimhost:9443 -u valerie -p secret
  -n HRAppService -w C:\appid -i -o valerie -x
```

discover-services

Automatically discovers Windows services on resources that you specify then stores the list in a CSV file. You can use the CSV file to load discovered services into service center.

Syntax

discover-services [options]

Options

-h,--hosts <*hosts*>

Specify a comma-delimited list of resources where automatic service discovery takes place.

For example: 192.0.2.1,192.0.2.2,192.0.2.5

-f,--file <*path*>

Specifies the location and filename of the file to create.

For example: C:\temp\services.csv

-v,--verbose
Verbose output displays additional information.

-x,--silent

Quiet display option. No prompts are displayed to the user.

Common options

-?
Displays usage instructions.

Examples

Example: Discovers and identifies services on a host that you specify and stores the results in a CSV file. Options specified in short form.

```
discover-services -h 192.0.2.1 -f C:\discovered-services.csv
```

Chapter 7. Virtual appliance commands

Access the command line interface (CLI) of the virtual appliance by using either an `ssh` session or the console.

The following paragraphs are general notes about the usage of the CLI. Examples of specific commands by using the CLI are provided through the remainder of this document.

The following example shows the transcript of using an `ssh` session to access the virtual appliance.

```
usernameA@example.com> ssh -l admin pimva.example.com
admin@pimva.example.com's password:
Welcome to the IBM Security Privileged Identity Manager appliance
Enter "help" for a list of available commands
pimva.example.com> ispim
pimva.example.com:ispim> help
Current mode commands:
coredumps           Work with the ISPIM service coredump files.
firmware_update     Work with the ISPIM firmware settings.
service_properties  Work with the ISPIM properties settings.
service_trace       Work with the ISPIM trace settings.
trusted_certs       Manage trusted certificates.
Global commands:
back               Return to the previous command mode.
exit               Log off from the appliance.
help               Display information for using the specified command.
reboot             Reboot the appliance.
shutdown           End system operation and turn off the power.
top                Return to the top level.
pimva.example.com:ispim>
```

You can also access the console by using the appropriate VMware software. For example, VMware vSphere Client.

The IBM Security Privileged Identity Manager virtual appliance CLI commands are broadly divided into the following main sections:

- Current mode commands
- Global commands

Note: The CLI contains only a subset of the function available from the graphical user interface.

Current mode commands for the virtual appliance

The initial virtual appliance settings wizard runs the first time that an Administrator logs on to the command line interface (CLI) of an unconfigured IBM Security Privileged Identity Manager virtual appliance. The topic provides information about the sub sections of the IBM Security Privileged Identity Manager virtual appliance CLI command that is specific to IBM Security Privileged Identity Manager.

In the current mode commands, the **`ispim`** command is used to work with the IBM Security Privileged Identity Manager settings. When an Administrator or a user enters the **`ispim`** command, the following sub sections are listed.

firmware_update

The sub section provides options to work with IBM Security Privileged Identity Manager firmware updates.

delete_firmware
Deletes firmware updates from the system.

install_firmware
Installs the available firmware update to the system.

list_firmware
Lists firmware updates from a USB device.

transfer_firmware
Transfers firmware update from a USB device to the system.

service_properties

The sub section provides options to change the properties of the services.

You can see the list of modifiable properties at http://www.ibm.com/support/knowledgecenter/SSRMWJ_6.0.0.2/com.ibm.isim.doc_6.0.0.2/reference/ref/ref_ic_props_supp_table.htm. Use the IBM Security Privileged Identity Manager virtual appliance CLI for the properties that are not available in the graphical user interface.

list_properties
Lists all the properties added through CLI.

add_property
Adds a property that is managed through CLI.

update_property
Updates an existing property added through CLI.

list_syslog
Lists all the values of `syslog` properties.

update_syslog
Updates the values of `syslog` properties.

service_trace

The sub section provides options to manage the log levels for the services. This sub section is provided for the troubleshooting.

add_trace
Adds a service trace level that is managed through CLI.

list_trace
Lists all the service trace level added through CLI.

update_trace
Updates a service trace level added through CLI.

coredumps

The sub section provides options to manage the core dump files. This sub section is provided for the troubleshooting.

delete_coredump
Deletes the core dump files.

list_coredump
Lists all the core dump files.

trusted_certs

Manage the list of trusted certificates on IBM Security Privileged Identity Manager so that you can connect securely to managed targets, external data tiers, and adapters.

add Adds a certificate.

delete Deletes a certificate.

show Show details about a certificate.

Global commands

The IBM Security Privileged Identity Manager virtual appliance CLI commands are broadly divided into the two sections such as current mode commands and global commands. The topic provides information about the IBM Security Privileged Identity Manager virtual appliance CLI commands for the following functions.

The following list gives a high-level overview of the functions available from the command line interface.

fixpacks

The function works with the fix packs. The corresponding task can be completed by using the graphical user interface. Navigate to **Manage** > **Updates and Licensing** > **Fix Packs**.

install Installs the available fix packs on the inserted USB device.

list Lists the available fix packs on the inserted USB device.

rollback
Uninstalls the most recently installed fix pack.

view_history
Shows the installation history for all fix packs.

license

The function works with the licenses.

management

dns Works with the virtual appliance DNS settings.

hostname
Works with the virtual appliance host name.

interfaces
Works with the management interface settings.

set_password
Sets the virtual appliance password.

snapshots

The function works with the snapshots. The corresponding task can be completed by using the graphical user interface. Navigate to **Manage** > **System Settings** > **Snapshots**.

Note: You must restart the virtual appliance after you apply the snapshot.

apply Applies a policy snapshot file to the system.

create Creates a snapshot of current policy files.

delete Deletes a policy snapshot file.

download
Downloads a policy snapshot file to a USB flash drive.

get_comment
Views the comment that is associated with a policy snapshot file.

list Lists the policy snapshot files.

set_comment
Replaces the comment that is associated with a policy snapshot file.

upload
Uploads a policy snapshot file from a USB flash drive.

support

The function generates the support files. The corresponding task can be completed by using the graphical user interface. Navigate to **Manage** > **System Settings** > **Support Files**.

create Creates a support information file.

delete Deletes a support information file.

download
Downloads a support information file to a USB flash drive.

get_comment
Views the comment that is associated with a support information file.

list Lists the support information files.

set_comment
Replaces the comment that is associated with a support information file.

tools

nslookup
Queries internet domain name servers.

ping Sends an `ICMP ECHO_REQUEST` to network hosts.

traceroute
Traces a packet from a computer to a remote destination. Shows the required number of hops for a packet that is required to reach the destination and the duration of each hop.

Cleaning core dump files

You can clean core dump files through the command-line interface in the IBM Security Privileged Identity Manager virtual appliance.

About this task

To see a list of available commands, enter the `help` command at the command-line prompt. The **`help`** command provides detailed information about each command from the list.

Procedure

1. From the command-line interface, log on to the IBM Security Privileged Identity Manager virtual appliance.

   For example:

   ```
   usernameA@example.com> ssh -l admin pimvasrv
   admin@pimvasrv's password: admin
   ```

   The following message is displayed:

   ```
   Welcome to the IBM Security Privileged Identity Manager appliance
   Enter "help" for a list of available commands
   ```

2. Enter the `help` command at the `pimvasrv` prompt for a list of available commands. The following result is displayed:

   ```
   Current mode commands:
   firmware            Work with firmware images.
   fixpacks            Work with fix packs.
   ispim               Work with the ISPIM settings.
   license             Work with licenses.
   management          Work with management settings.
   snapshots           Work with policy snapshot files.
   support             Work with support information files.
   tools               Work with network diagnostic tools.
   updates             Work with firmware and security updates.
   Global commands:
   back                Return to the previous command mode.
   exit                Log off from the appliance.
   help                Display information for using the specified command.
   reboot              Reboot the appliance.
   shutdown            End system operation and turn off the power.
   top                 Return to the top level.
   ```

3. Enter the `ispim` command at the `pimvasrv` prompt.
4. Enter the `help` command at the `pimvasrv:ispim` prompt for a list of available commands. The following result is displayed:

   ```
   Current mode commands:
   coredumps           Work with the ISPIM service coredump files.
   firmware_update     Work with the ISPIM firmware settings.
   service_properties  Work with the ISPIM properties settings.
   service_trace       Work with the ISPIM trace settings.
   trusted_certs       Manage trusted certificates.
   Global commands:
   back                Return to the previous command mode.
   exit                Log off from the appliance.
   help                Display information for using the specified command.
   reboot              Reboot the appliance.
   shutdown            End system operation and turn off the power.
   top                 Return to the top level.
   ```

5. Enter the `coredumps` command at the `pimvasrv:ispim` prompt.
6. Enter the `help` command at the `pimvasrv:coredumps` prompt for a list of available commands. The following result is displayed:

   ```
   Current mode commands:
   delete_coredump     Delete coredump files.
   list_coredump       List coredump files.
   Global commands:
   back                Return to the previous command mode.
   exit                Log off from the appliance.
   ```

```
help                Display information for using the specified command.
reboot              Reboot the appliance.
shutdown            End system operation and turn off the power.
top                 Return to the top level.
```

7. For a detailed help on `list_coredump`, enter the `help list_coredump` command at the `pimvasrv:coredumps` prompt. The following result is displayed:

   ```
   List of coredump files.
   Usage: list_coredump
   ```

8. Enter the `list_coredump` command at the `pimvasrv:coredumps` prompt. The following result is displayed:

   ```
   List of core dump files:
   1: 4.0K /opt/IBM/TDI/core.2333.23442.22334.00004.dmp
   2: 4.0K /opt/IBM/ispim/core.2333.23442.22334.00007.dmp
   3: 4.0K /opt/IBM/wlp/lib/core.2333.23442.22334.00002.dmp
   4: 4.0K /opt/IBM/wlp/core.2333.23442.22334.00009.dmp
   5: 4.0K /opt/IBM/isamesso82/core.2333.23442.22334.00006.dmp
   6: 4.0K /opt/IBM/WebSphere85/core.2333.23442.22334.00005.dmp
   7: 4.0K /opt/IBM/HTTPServer/core.2333.23442.22334.00003.dmp
   ```

9. To get a detailed help on `delete_coredump`, enter the `help delete_coredump` command at the `pimvasrv:coredumps` prompt. The following result is displayed:

   ```
   Delete coredump files.
   Usage: delete_coredump
   ```

10. Enter the `delete_coredump` command at the `pimvasrv:coredumps` prompt. The following result is displayed:

    ```
    1: /opt/IBM/TDI/core.2333.23442.22334.00004.dmp
    2: /opt/IBM/ispim/core.2333.23442.22334.00007.dmp
    3: /opt/IBM/wlp/lib/core.2333.23442.22334.00002.dmp
    4: /opt/IBM/wlp/core.2333.23442.22334.00009.dmp
    5: /opt/IBM/isamesso82/core.2333.23442.22334.00006.dmp
    6: /opt/IBM/WebSphere85/core.2333.23442.22334.00005.dmp
    7: /opt/IBM/HTTPServer/core.2333.23442.22334.00003.dmp
    8: Delete All
    ```

11. Do one of the following actions.
    - Enter the index number for the core dump file that you want delete. For example, specify 1 at **Enter index**.

      The following message is displayed:

      ```
      Are you sure you want to delete this core dump file?
      ```

      – Type the input as `YES` to confirm and delete the core dump file that you want to delete. The following message is displayed:

        ```
        The core dump file '/opt/IBM/TDI/core.2333.23442.22334.00004.dmp' is deleted.
        ```

    - Enter the index number for the **Delete All** option to delete all the core dump files. For example, specify 8 at **Enter index**.

      – Type the input as `YES` to confirm and delete one or all the core dump files. The following message is displayed:

        ```
        The core dump files were deleted.
        ```

Example

The following example shows the entire transcript to delete one or all the core dump files.

```
usernameA@example.com> ssh -l admin pimvasrv
admin@pimvasrv's password:admin
Welcome to the IBM Security Privileged Identity Manager appliance
Enter "help" for a list of available commands
pimvasrv> help
Current mode commands:
firmware            Work with firmware images.
fixpacks            Work with fix packs.
ispim               Work with the ISPIM settings.
license             Work with licenses.
management          Work with management settings.
snapshots           Work with policy snapshot files.
support             Work with support information files.
```

```
tools               Work with network diagnostic tools.
updates             Work with firmware and security updates.
Global commands:
back                Return to the previous command mode.
exit                Log off from the appliance.
help                Display information for using the specified command.
reboot              Reboot the appliance.
shutdown            End system operation and turn off the power.
top                 Return to the top level.
pimvasrv> ispim
pimvasrv:ispim> help
Current mode commands:
coredumps           Work with the ISPIM service coredump files.
firmware_update     Work with the ISPIM firmware settings.
service_properties  Work with the ISPIM properties settings.
service_trace       Work with the ISPIM trace settings.
trusted_certs       Manage trusted certificates.
Global commands:
back                Return to the previous command mode.
exit                Log off from the appliance.
help                Display information for using the specified command.
reboot              Reboot the appliance.
shutdown            End system operation and turn off the power.
top                 Return to the top level.
pimvasrv:ispim> coredumps
pimvasrv:coredumps> help
Current mode commands:
delete_coredump     Delete coredump files.
list_coredump       List coredump files.
Global commands:
back                Return to the previous command mode.
exit                Log off from the appliance.
help                Display information for using the specified command.
reboot              Reboot the appliance.
shutdown            End system operation and turn off the power.
top                 Return to the top level.
pimvasrv:coredumps> help list_coredump
List coredump files.
Usage: list_coredump
pimvasrv:coredumps> list_coredump
List of core dump files:
1: 4.0K /opt/IBM/TDI/core.2333.23442.22334.00004.dmp
2: 4.0K /opt/IBM/ispim/core.2333.23442.22334.00007.dmp
3: 4.0K /opt/IBM/wlp/lib/core.2333.23442.22334.00002.dmp
4: 4.0K /opt/IBM/wlp/core.2333.23442.22334.00009.dmp
5: 4.0K /opt/IBM/isamesso82/core.2333.23442.22334.00006.dmp
6: 4.0K /opt/IBM/WebSphere85/core.2333.23442.22334.00005.dmp
7: 4.0K /opt/IBM/HTTPServer/core.2333.23442.22334.00003.dmp
pimvasrv:coredumps> help delete_coredump
Delete coredump files.
Usage: delete_coredump
pimvasrv:coredumps> delete_coredump
1: /opt/IBM/TDI/core.2333.23442.22334.00004.dmp
2: /opt/IBM/ispim/core.2333.23442.22334.00007.dmp
3: /opt/IBM/wlp/lib/core.2333.23442.22334.00002.dmp
4: /opt/IBM/wlp/core.2333.23442.22334.00009.dmp
5: /opt/IBM/isamesso82/core.2333.23442.22334.00006.dmp
6: /opt/IBM/WebSphere85/core.2333.23442.22334.00005.dmp
7: /opt/IBM/HTTPServer/core.2333.23442.22334.00003.dmp
8: Delete All
Enter index: 1
Are you sure you want to delete all the core dump files from the system?
Enter 'YES' to confirm: YES
The core dump file '/opt/IBM/TDI/core.2333.23442.22334.00004.dmp' is deleted
pimvasrv:coredumps> delete_coredump
1: /opt/IBM/ispim/core.2333.23442.22334.00007.dmp
2: /opt/IBM/wlp/lib/core.2333.23442.22334.00002.dmp
3: /opt/IBM/wlp/core.2333.23442.22334.00009.dmp
4: /opt/IBM/isamesso82/core.2333.23442.22334.00006.dmp
5: /opt/IBM/WebSphere85/core.2333.23442.22334.00005.dmp
6: /opt/IBM/HTTPServer/core.2333.23442.22334.00003.dmp
7: Delete All
Enter index: 7
Are you sure you want to delete all the core dump files from the system?
Enter 'YES' to confirm: YES
The core dump files were deleted.
pimvasrv:coredumps> delete_coredump
No coredump files were found.
pimvasrv:coredumps>
```

What to do next

You can do the following actions:

- View the existing list of core dump files.
- Delete some core dump files from the existing list.

Enabling trace for the virtual appliance services

You can add a service trace level through the CLI. From the **Appliance Dashboard**, restart the relevant virtual appliance service such as `Identity`, `SingleSignOn`, or `SessionRecorder`, and examine the log files for the new debug or trace messages.

Procedure

1. Log on to the virtual appliance.

   For example:

   ```
   usernameA@example.com> ssh -l admin pimva.example.com
   admin@pimva.example.com's password:
   ```

   The following message is displayed:

   ```
   Welcome to the IBM Security Privileged Identity Manager appliance
   ```

2. Enter the `ispim` command at the `pimva.example.com` prompt.
3. At the prompt, enter the **`help`** command for a list of available commands.
4. Enter the `service_trace` command at the `pimva.example.com:ispim` prompt.
5. At the prompt, enter the **`help`** command for a list of available commands. The following sub sections are listed under `service_trace`:

   `add_trace`
   Adds a service trace level.

   `list_trace_history`
   Lists the service trace level history.

   `update_trace`
   Updates a service trace level.

6. From the list of available commands, enter the `add_trace` commandat the `pimva.example.com:service_trace` prompt.
7. Type an index for the name of the service. For example, type the input as `2` at **Enter index** for `SingleSignOn`. The **Name of the service** can be as follows:

   ```
   1: Identity
   2: SingleSignOn
   3: SessionRecorder
   ```

8. Type the name of the package for the selected service at **Name of the package**. For example, `encentuate.*`.

   Note: The value for the name of the package can be only a single package or component name. For example, `encentuate.*`. Adding another package by using the **`add_trace`** command overwrites the current trace level setting.

9. Type an index to assign the value for the trace level of the package. For example, type the input as `8` at **Enter index** to assign `audit`. The values can be as follows:

   ```
   1: all
   2: finest
   3: finer
   4: fine
   5: detail
   ```

```
6: config
7: info
8: audit
9: warning
10: severe
11: fatal
12: off
```

Results

The property is updated with the new value. Complete these steps to apply the new settings:

1. Restart IBM Security Privileged Identity Manager to apply the new settings.
2. Enter the `list_trace` command at the `pimva.example.com:service_trace` prompt.

   View the following information:

   ```
   pimServiceName:SingleSignOn pimPackageName:encentuate.* pimTraceValue:audit
   ```

What to do next

Update a service trace level. For example, update the `Identity` virtual appliance service.

1. Enter the `update_trace` command at the `pimva.example.com:service_trace` prompt.
2. Type an index to assign the value for the trace level of the package. For example, type the input as `7` at **Enter index** to update to `info`.

   Note: The default value for the trace level is `info`.

The following example shows the transcript to set the trace level for the `Identity` service:

```
usernameA@example.com> ssh -l admin pimva.example.com
admin@pimva.example.com's password:
Welcome to the IBM Security Privileged Identity Manager appliance
Enter "help" for a list of available commands
pimva.example.com> ispim
Enter "help" for a list of available commands
1) firmware_update     Work with the ISPIM firmware settings.
2) service_properties  Work with the ISPIM properties settings.
3) service_trace       Work with the ISPIM trace settings.
pimva.example.com:ispim> service_trace
pimva.example.com:service_trace> help
Current mode commands:
add_trace           Add a new service trace level.
list_trace_history  List the service trace level history.
update_trace        Update an service trace level.
Global commands:
back                Return to the previous command mode.
exit                Log off from the appliance.
help                Display information for using the specified command.
reboot              Reboot the appliance.
shutdown            End system operation and turn off the power.
top                 Return to the top level.
pimva.example.com:service_trace> update_trace
 Name of the service :
1: Identity
2: SingleSignOn
3: SessionRecorder
Enter index: 2
 Name of the package : *
 Value for the trace level :
```

```
1: all
2: finest
3: finer
4: fine
5: detail
6: config
7: info
8: audit
9: warning
10: severe
11: fatal
12: off
Enter index: 7
pimva.example.com:service_trace> list_trace
 pimServiceName:SingleSignOn pimPackageName:* pimTraceValue:info
pimva.example.com:service_trace>
```

Chapter 8. Dynamic tags in mail templates

IBM Security Privileged Identity Manager mail templates allow dynamic retrieval, substitution, and decision making in creating a message.

Dynamic content tags and examples

IBM Security Privileged Identity Manager provides dynamic content tags to allow text substitution and enable translation. The tags are used for the emails that are generated by these tasks:

- Designing workflows
- Specifying mail activity
- Manual service notification
- Post office
- Reminder template
- Default system notifications
- Delegation notifications

These tags are associated with dynamic content:

JavaScript code

Handles JavaScript and runs the JavaScript content that is contained between the open and close tags. This tag contains child tags unless they return a string. JavaScript code is called in `<JS>MyJavaScriptCode</JS>` delimiters.

Table 2. Syntax and example of using JavaScript code to replace message content.

Syntax	Example

`<JS>text or JavaScript tag</JS>`	Enter each `<JS></JS>` statement as a single line:
	`An account request has been initiated for`
	`<JS>process.requesteeName;</JS>`
	`<JS>if (var x=process.getParent() !=null)`
	`return x`
	`</JS>`

`<JS escapeentities="false">text or JavaScript tag</JS>`	When specified as `"false"`, any text that is returned by the JavaScript execution does not have its HTML entity tags escaped. For instance, the < character does not return as `&lt;`. This option might be useful when the execution of the JavaScript code returns XML. For example, embedding XHTML body notifications inside the XHTML body of the post office template. The default for this attribute is `"true"`, so not specifying the tag escapes the characters.

Table 2. Syntax and example of using JavaScript code to replace message content. (continued)

Syntax	Example
`<JS removexhtmlheader="false">text or JavaScript tag</JS>`	If `removexhtmlheader="true"` is in the `JS` tag, any text that is returned from the JavaScript does not have the DTD statement in the XHTML content. The text that is returned from the JavaScript has the DTD statement in the XHTML content when either of the following conditions exist: • `removexhtmlheader="false".` • It is not placed in the `JS` tag. The default value of this attribute is `false`. Not specifying the flag in the tag puts the DTD statement in the XHTML content.

Replace tag

Formats the message that is represented by the key to allow string replacement. The formatted string can have zero or more parameters. Parameters can contain strings, activity IDs, or JavaScript. The string inside the key must exist in the `CustomLabels.properties` file. Strings are sourced from a `CustomLabels.properties` resource bundle file or from the `Labels.properties` file.

The key of the string replacement can be specified with the key attribute or by adding a **KEY** tag between **RE** tags. Specifying a key that uses both the attribute and tag at the same time results in an exception.

The tag has these parameters:

Key Represents the resource bundle key for a **RE** tag. For example:

```
<RE key="key">
</RE>
```

PARM Represents the parameters for a **RE** tag. For example:

```
<RE key="key">
<PARM>with plain text</PARM>
</RE>
```

Table 3. Syntax and examples of using a RE tag to replace message content.

Syntax	Example

`<RE key="message key">`
`<PARM>`*text or JavaScript tag*`</PARM>`
`</RE>`
 	
or enter each `<KEY></KEY>` statement as a single line:
`<RE><KEY>`*message key or JavaScript tag to return a key*`</KEY>`
`<PARM>`*text or JavaScript tag*`</PARM>`
`</RE>`
 	
The `KEY` can be specified by either an attribute on the `RE` tag, or as a subelement of the `RE` tag by using the tag `KEY`.
	`<RE key="message key">`
	`<PARM>with plain text</PARM>`
	`<PARM><JS>process.requesteeName;`
	`</JS></PARM></RE>`
	
	Output:
	This is a formatted string replacement example **with plain text** and JavaScript code for requestee name **John Smith.**

Table 3. Syntax and examples of using a RE tag to replace message content. (continued)

Syntax	Example

To enable string replacement for translation, specify a custom label in a `CustomLabels.properties` file to overwrite a `Labels.properties` key.
 	
For example, the `Labels.properties` file contains this key/value pair.
`readOnlyDateFormat=MMM dd, yyy hh:mm:ss z`
 	
To override this format, add the same key to the `CustomLabels.properties` file.
	`<RE key="readOnlyDateFormat">`
	`<PARM><JS>if (process.scheduled !=null)`
	`return process.scheduled.getTime();`
	`else`
	`return "";</JS></PARM></RE>`
	
	Output:
	`Apr 18, 2005 05:20:52 EDT`

Non-compliant message tag

Represents a message that describes the noncompliant attributes of an account. For example:

```
<CAMessage/>
```

Dynamic content message tags

Tags are delimited in `<TAG/>` syntax, such as the following examples:

Table 4. Syntax and example of using tags to replace message content.

Syntax	Example
`<TagName/>`	`<CAMessage/>` Returns a string that describes the non-compliant attributes of an account.
	`<ManualServiceAddAccount/>` Returns a string that contains the text body for manual service email notification.
	`<rfiActivityHasBeenSubmitted/>` Returns a string that contains the text body of an RFI activity that was submitted in an account request workflow.

ID tag Represents the activity ID in the form: `Process.ActivityId.` For example:

```
<ID/>
```

ITIMURL tag

Based on group membership of the person. It represents the URL of the IBM Security Privileged Identity Manager Server. A forced URL can be applied by using the `forcedurl` attribute of the tag. This attribute contains constant values such as the value `console`, `enduser`, or `ISC`.

Table 5. Syntax and examples of ITIMURL.

Syntax	Example
`<ITIMURL/>`	Based on group membership of the person. It represents the URL of the IBM Security Privileged Identity Manager Server.

Table 5. Syntax and examples of ITIMURL. (continued)

Syntax	Example
`<ITIMURL forcedurl="enduser"/>`	Represents the URL of the graphical user interface on the IBM Security Privileged Identity Manager Server. If the `forcedurl` attribute is used, the URL is not generated based on the group membership of the person. These values are associated with this attribute: **`enduser`** The URL points at the self-service graphical user interface. **`console`** The URL points at the administrator graphical user interface. **`servicecenter`** The URL points at the service center graphical user interface.
`<ITIMURL forcedurl="console"/>`	
`<ITIMURL forcedurl="servicecenter"/>`	

Properties file values

To change templates, you can add a property in the `CustomLabels.properties` file or create your own properties and values by using the Update Property page from the **Appliance Dashboard** of the

IBM Security Privileged Identity Manager virtual appliance

Required escape characters and JavaScript

The following characters must be escaped by using the appropriate HTML entity form that has the format `&`*`entity`*`;`. This action ensures that the notification template XML is well-formed.

Table 6. Escape characters

Escape character	Character
`<`	Less Than (<)
`>`	Greater Than (>)
`&quote;`	Quotation (")
`&apos;`	Apostrophe (')
`&`	Ampersand (&)

For example, to use the following JavaScript

```
if (i<4) return "less than four";
```

the dynamic content tag is

```
<JS> if (i&lt;4) return &quote;less than four&quote;;</JS>
```

Common formatting patterns in the XHTML body

Default messages are formatted with a common pattern in the XHTML body and also contain message-unique statements.

For example, the XHTML for the to-do reminder template calls a common style sheet (the imperatives.css file) and logos. Message-unique statements are similar to the following ones:

```
<!-- Start of notification body -->
      <textBody/>
        <RE key="escalation_note"/> <escalationTime/>
     </td>
    </tr>
<!-- End of notification body -->
```

The following example shows a complete set of statements in an XHTML body:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>$TITLE</title>
<meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />
<link type="text/css" title="Styles" rel="stylesheet"
 href="$BASE_URL/console/css/imperative.css" />
</head>

<!-- Put Next statement on one line -->

<body topmargin="0" marginheight="0" leftmargin="0" marginwidth="0"
 bgcolor="ffffff">

<!-- Block for the Template Header part  -->
<table width="100%" border="0" cellpadding="0" cellspacing="0">
 <tbody>
  <tr>
   <!-- Security logo -->
   <td width="186" background="$BASE_URL/console/html/images/mid-part-1.gif">
   <img src="$BASE_URL/console/html/images/left-tiv-1.gif" alt="$LOGO_ALT" /></td>
   <!-- Middle part -->
   <td background="$BASE_URL/console/html/images/mid-part-1.gif" width="692"></td>
   <!-- IBM logo -->
   <td background="$BASE_URL/console/html/images/ibm_banner.gif" width="96"></td>
  </tr>
 </tbody>
</table>

<!-- Title Bar -->
<table width="100%" border="0" cellpadding="0" cellspacing="0">
 <tbody>

  <tr>
   <td background="$BASE_URL/console/html/images/titlebar_middle.gif"
   height="23" width="8">
   <img border="0" src="$BASE_URL/console/html/images/titlebar_left.gif"
    width="10" height="23" /></td>
   <!-- ISIM Notification Lable -->
<td background="$BASE_URL/console/html/images/titlebar_middle.gif"
  height="23" classpath="portfolio-header" width="979">$TITLE</td>
   <td background="$BASE_URL/console/html/images/titlebar_middle.gif"
   height="23" width="5"><img
    border="0" src="$BASE_URL/console/html/images/titlebar_right.gif"
    width="5" height="23" /></td>
  </tr>
 </tbody>
</table>

<table width="100%" border="0" cellpadding="0" cellspacing="0">
 <tbody>
```

```
  <tr>
   <!-- Backgroud for the template body -->
   <td background="$BASE_URL/console/html/images/portfolio_background.gif"
   height="148">
   <table border="0" cellspacing="0" cellpadding="0" width="100%">
               <tr>
                   <td align="left" class="text-description" height="65">
                   <!-- Start of notification body -->
      <textBody/>
        <RE key="escalation_note"/> <escalationTime/>
     </td>
    </tr>
        <!-- End of notification body -->           </table>
     </td>
  </tr>
 </tbody>
</table>
<!-- Copy Right Table -->
<table width="100%" border="0" cellpadding="0" cellspacing="0">
 <tbody>
  <tr bgcolor="#9d9d9d" align="center" valign="middle">
   <td class="text-description"><span class="cont1" id="W57ea57ea"><span
    class="txt" id="text">IBM Copyright 2007</span></span></td>
  </tr>
 </tbody>
</table>
</body>
</html>
```

Mail templates

You define mail templates to deliver customized message notifications. The templates use several customization functions.

Templates have these main parts:

Subject
: Describes an activity to a recipient of the notification. The subject can consist of plain text and dynamic content tags. If no subject is specified for manual service activities, no email is sent.

Text body
: Describes the outcome of an activity, such as an account approval. The content can consist of plain text, dynamic content tags, and JavaScript code.

XHTML body
: Provides the content of the email as an HTML message.

Dynamic content can include dynamic content message tags, JavaScript code, and tags that replace variables with other values or reference a property that allows translation with the `CustomLabels.properties` file.

Generic workflow messages

IBM Security Privileged Identity Manager provides default generic workflow messages.

Default generic workflow templates

All the generic workflow notice templates can be customized. IBM Security Privileged Identity Manager provides these default generic workflow notice templates:

Activity Timeout Template

Provides information that the workflow activity is timed out and terminated. By default, this template is enabled.

For example, the template provides this message:

```
Workflow activity is being timed out and will be terminated
by the workflow system.

The following activity has timed out.The activity will be terminated
by the workflow system and the result set to Terminated.

Activity Information

View Changes: http://localhost:9090/itim/console
Activity ID: ADApproval
Activity: AD Account Approval
Time Started: Jun 09, 2007 12:28:45 IST
Time Completed:
Result Summary: Escalated
State: Running
Activity Type: Manual Approval/Reject

Process Information

 Process ID: 1099575082113388748
Activity: Default AD Account Approval Workflow
Description:
State:Running
Date submitted: Jun 09, 2007 12:23:41 IST
Time Completed:
Result Summary:
Requester: 1099572462907357646
Requestee: firstname lastname
Subject:
Comment:
Detail:
```

The subject statement is:

```
<RE key="activity_timeout_subject" />
```

The plain text is:

```
<RE key="activity_timeout_message" />

<RE key="activity_timeout_detail" />

<RE key="activityInformation" />
<ITIMURL/>
<RE key="activityID"/>: <JS>activity.id;</JS>
<RE key="name"/>: <JS>activity.name;</JS>
<RE key="timeStarted"/>: <RE key="readOnlyDateFormat"><PARM>
 <JS>if (activity.started != null)
 return activity.started.getTime();
 else return '';</JS></PARM></RE>
 <RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>if (activity.completed != null)
  return activity.completed.getTime();
  else return '';</JS></PARM></RE>
 <RE key="resultSummary"/>: <RE><KEY>
  <JS>process.STATE_PREFIX + activity.resultSummary;
  </JS></KEY></RE>
 <RE key="state"/>: <RE><KEY><JS>process.STATE_PREFIX+activity.state;
  </JS></KEY></RE>
 <RE key="activityType"/>: <RE><KEY>
  <JS>activity.TYPE_PREFIX + activity.type;</JS>
  </KEY></RE>
 <RE><KEY><JS>activity.TYPE_PREFIX + activity.subtype;</JS></KEY></RE>
```

```
<RE key="processInformation" />

<RE key="processID"/>: <JS>process.id;</JS>
<RE key="name"/>: <RE><KEY><JS>process.name;</JS></KEY></RE>
<RE key="description"/>: <RE><KEY>
 <JS>process.description;</JS></KEY></RE>
<RE key="state"/>: <RE><KEY><JS>process.STATE_PREFIX + process.state;
 </JS></KEY></RE>
<RE key="timeScheduled"/>: <RE key="readOnlyDateFormat"><PARM>
 <JS>if (process.scheduled != null) return process.scheduled.getTime();
 else return '';</JS></PARM></RE>
<RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
 <JS>if (process.completed != null) return process.completed.getTime();
 else return '';</JS></PARM></RE>
<RE key="resultSummary"/>: <RE><KEY>
 <JS>process.STATE_PREFIX + process.resultSummary;
 </JS></KEY></RE>
<RE key="requester"/>: <JS>process.requestorName;</JS>
<RE key="requestedFor"/>: <JS>process.requesteeName;</JS>
<RE key="subject"/>: <JS>process.subject;</JS>
<RE key="comment"/>: <JS>process.comment;</JS>
<RE key="detail"/>: <JS>process.resultDetail;</JS>
```

Delegation Template

Provides the default template for delegation, which includes the new delegation information. By default, this template is enabled and cannot be disabled. If any exception is thrown while evaluating JavaScript in the notification template or parsing the notification template, then the default delegation notification is sent.

For example, the template provides this message:

```
You have been selected to be the delegate:

For: John Doe

From: Tue Jul 03 08:00:13 IST 2012

To: Fri Jul 06 20:00:13 IST 2012

The subject statement is:
<RE key="delegationMailSubject"/>

The plain text is:
<RE key="delegationMailContent"/>

<RE key="delegationMailDelegator"/>:<JS>Delegate.getDelegator().name;</JS>

<RE key="delegationMailFrom"/>:<JS>Delegate.getStartDate();</JS>

<RE key="delegationMailTo"/>:<JS>Delegate.getEndDate();</JS>
```

Process Completion Template

Provides information that the workflow activity has completed. By default, this template is enabled.

For example, the template provides this message when an activity is completed without being canceled:

```
A workflow process, 1416721862784240178, has completed.
Result Summary: Success
The following process has completed

Process Information

View Changes: http://localhost:9090/itim/console
Process ID: 1416721862784240178
Activity:
Description: Modify Provisioning Policy Process
State: Completed
Date submitted: May 16, 2007 12:22:58 IST
Time Completed: May 16, 2007 01:44:17 IST
Result Summary: Success
Requester: System Administrator
```

```
Requestee:
Subject: Default Provisioning Policy for service Win Local Profile
Comment:
Detail:
```

For example, the template provides this message when an activity is canceled:

```
Subject: A workflow process, 6690130336188564930, has completed.
Result Summary: Failed
The following process has completed

Process Information

View Changes: http://localhost:80/itim/console
Process ID: 6690130336188564930
Activity: Person Add
Description: Person Add Process
State: Canceled
Date submitted: Jan 30, 2014 01:13:59 CST
Time Completed: Jan 29, 2014 01:13:22 CST
Result Summary: Failed
Requester: System Administrator
Requestee: firstname lastname
Subject:
Comment:
Detail:
Canceled By: System Administrator
Date Canceled:  Jan 29, 2014 01:13:22 CST
Canceled Justification: No longer needed
```

The subject statement is:

```
<RE key="processCompletedSubject"><PARM><JS>process.id;</JS></PARM>
 <PARM><RE key="resultSummaryValue"><PARM><RE><KEY>
 <JS>process.STATE_PREFIX + process.resultSummary;
 </JS></KEY></RE></PARM></RE></PARM></RE>
```

The plain text is:

```
 <RE key="process_completed_message" />

 <RE key="processInformation" />
 <ITIMURL/>
 <RE key="processID"/>: <JS>process.id;</JS>
 <RE key="name"/>: <RE><KEY><JS>process.name;</JS></KEY></RE>
 <RE key="description"/>: <RE><KEY><JS>process.description;</JS>
  </KEY></RE>
 <RE key="state"/>: <RE><KEY>
  <JS>process.STATE_PREFIX + process.state;</JS></KEY></RE>
 <RE key="timeScheduled"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>if (process.scheduled != null)
  return process.scheduled.getTime();
  else return '';</JS></PARM></RE>
 <RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>if (process.completed != null)
  return process.completed.getTime();
  else return '';</JS></PARM></RE>
 <RE key="resultSummary"/>: <RE><KEY>
  <JS>process.STATE_PREFIX + process.resultSummary;</JS>
  </KEY></RE>
 <RE key="requester"/>: <JS>process.requestorName;</JS>
 <RE key="requestedFor"/>: <JS>process.requesteeName;</JS>
 <RE key="subject"/>: <JS>process.subject;</JS>
 <RE key="comment"/>: <JS>process.comment;</JS>

 <RE key="detail"/>: <JS>process.resultDetail;</JS>
<JS>if (process.cancelor_name != null)
  { '<RE key="CanceledBy"/>: ' + process.cancelor_name; }</JS>
<JS>if (process.cancelor_name != null)
  { '<RE key="DateCanceled"/>: '; }</JS>
<RE key="readOnlyDateFormat"><PARM>
<JS>if (process.canceled_date != null) return process.canceled_date.getTime();
  else return '';</JS>
</PARM></RE>
```

```
<JS>if (process.cancelor_name != null) { '<RE key="CanceledReason"/>:
<JS>if (process.canceled_justification == null) { return ' '; }
  else { return process.canceled_justification;}
</JS>'; }</JS>
```

Process Timeout Template

Provides information that the workflow process has timed out. By default, this template is enabled.

For example, the template provides this message:

```
Workflow activity is being timed out and will be
 terminated by the workflow system

Activity Information
View Changes: http://localhost:9080/itim/console
Activity ID: RECERTAPPROVAL
Activity: $ITIM_RECERTIFY
Time Started: Aug 02, 2007 03:18:54 IST
Time Completed:
Result Summary: Pending
State: Running
Activity Type: Manual Approval/Reject

Process Information

Process ID: 8566433417513336819
Activity: Recertification of Account/Access
Description: Recertification of Account/Access
State: Running
Date submitted: Aug 02, 2007 03:18:54 IST
Time Completed:
Result Summary:
Requester: org
Requestee: Person B
Subject: personb
Comment:
Detail:
```

The subject statement is:

```
<RE key="process_timeout_subject" />
```

The plain text is:

```
 <RE key="process_timeout_message" />

 <RE key="processInformation" />
 <ITIMURL/>
 <RE key="processID"/>: <JS>process.id;</JS>
 <RE key="name"/>: <RE><KEY><JS>process.name;</JS></KEY></RE>
 <RE key="description"/>: <RE><KEY><JS>process.description;</JS></KEY></RE>
 <RE key="state"/>: <RE><KEY>
  <JS>process.STATE_PREFIX + process.TIMEOUT;</JS></KEY></RE>
 <RE key="timeScheduled"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>if (process.scheduled != null) return process.scheduled.getTime();
  else return '';</JS></PARM></RE>
 <RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>if (process.completed != null) return process.completed.getTime();
  else return '';</JS></PARM></RE>
 <RE key="resultSummary"/>: <RE><KEY>
  <JS>process.STATE_PREFIX + process.resultSummary;</JS></KEY></RE>
 <RE key="requester"/>: <JS>process.requestorName;</JS>
 <RE key="requestedFor"/>: <JS>process.requesteeName;</JS>
 <RE key="subject"/>: <JS>process.subject;</JS>
 <RE key="comment"/>: <JS>process.comment;</JS>

 <RE key="detail"/>: <JS>process.resultDetail;</JS>
```

To-Do Reminder Template

Provides the default template for workflow reminders, which are email messages that remind users about pending activities to which they not responded. By default, this template is disabled.

For example, the template provides this message:

```
Subject: Pending workflow action:
 Case 6167063972298972180.6167064647650050990

The following request has been submitted for your approval
View Changes: http://localhost:9080/itim/console
Description: ApprovalWorkflow
Requestee: firstname lastname
Subject: subject
Request Initiated: Sep 05, 2007 05:42:18 IST
Process Reference: 6167063972298972180


Requested by process:
     Process ID: 6167052766519381908
     Process Name: Provision Account
     Description: Provision Account Process
     Requester: System Administrator
     Requestee: firstname lastname
     Subject: subject

 This WorkItem will be escalated on: Saturday, September 8, 2007.
```

The subject statement is:

```
<originalSubject/>
```

The plain text is:

```
 <textBody/>

  <RE key="escalation_note"/> <escalationTime/>
```

Organization management default messages

IBM Security Privileged Identity Manager provides default organization management messages.

Default organization management templates

All the organization management notice templates can be customized. IBM Security Privileged Identity Manager provides these default organization management notice templates:

Change Account Template

Provides information that the workflow activity has modified account information. By default, this template is disabled.

For example, the template provides this message:

```
Modified Account Information from IBM Security Identity Manager

The following ITIM Service [ITIM] account has been modified:

View Changes: http://localhost:9090/itim/console
Process Reference: 875016861865594505
Account ID: myaccount
Owner Name: firstname lastname
Time Completed: Jun 08, 2007 09:52:24 IST
```

The subject statement is:

```
<RE key="change_account_subject"/>
```

The plain text is:

```
 <RE key="account_changed"><PARM>
  <RE key="service_name_with_profile_name"><PARM>
  <JS>EmailContext.getAccountServiceName();</JS></PARM>
  <PARM><RE><KEY><JS>EmailContext.getAccountServiceProfileName();
  </JS></KEY></RE></PARM></RE></PARM></RE>
 <ITIMURL/>
 <RE key="processRef"/>: <JS>process.id;</JS>
```

```
<JS>if (EmailContext.getTransactionId() != '0')
 { '<RE key="TRANSACTION_ID_LABEL"/>: ' + EmailContext.getTransactionId(); }
 </JS>
<RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
<RE key="accountOwnerName"/>: <JS>EmailContext.getAccountOwnerName();</JS>
<RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
 <JS>(new Date()).getTime();</JS></PARM></RE>
<JS>if (EmailContext.hasNewAccess()) { '<RE key="accountNewAccess"/>:
 <JS>EmailContext.getAccountNewAccessAsString();</JS>\n'; }</JS>
 <JS>if (EmailContext.hasRemovedAccess()) { '<RE key="accountRemovedAccess"/>:
 <JS>EmailContext.getAccountRemovedAccessAsString();</JS>\n'; }</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
 { '<RE key="RETRIEVE_PASSWORD_TITLE"/>: ' +
  EmailContext.getPasswordRetrievalUrl(); }
 </JS>
<JS>if (EmailContext.getTransactionId() != '0')
 { '<RE key="passwordExpireLabel"/>:
<JS>if (EmailContext.getPasswordExpirePeriod() == 0)
 { '<RE key="passwordneverexpire"/>'; }
  else { EmailContext.getPasswordExpirePeriod(); }</JS>'; }</JS>
<JS>if (EmailContext.getTransactionId() != '0')
 { '<JS>if (EmailContext.getPasswordExpirePeriod() == 0)
 { '<RE key="additionalMsgForPwdRetrieval"/>'; }</JS>'; }</JS>
```

Deprovision Account Template

Provides information that the workflow activity has removed an account. By default, this template is enabled.

For example, the template provides this message:

```
Your account has been removed by IBM Security Identity Manager.

The following Odessa Service [ADProfile] account has been deprovisioned.

View Changes: http://host:9080/itim/selfui
Process Reference: 5870349043636872731
Account ID: myaccount
Owner Name: myname
Reason: Policy Enforcement
Time completed: May 03, 2007 03:54:22 IST
```

The subject statement is:

```
<RE key="remove_account_subject" />
```

The plain text is:

```
<RE key="account_deprovisioned">
 <PARM><RE key="service_name_with_profile_name">
 <PARM><JS>EmailContext.getAccountServiceName();</JS></PARM>
 <PARM><RE><KEY><JS>EmailContext.getAccountServiceProfileName();
 </JS></KEY></RE></PARM></RE></PARM></RE>
<ITIMURL/>
<RE key="processRef"/>: <JS>process.id;</JS>
<RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
<RE key="accountOwnerName"/>: <JS>EmailContext.getAccountOwnerName();</JS>
<RE key="reason"/>: <JS>EmailContext.getReason();</JS>
<RE key="deprovisionCompleted"/>: <RE key="readOnlyDateFormat">
 <PARM><JS>(new Date()).getTime();</JS></PARM></RE>
```

New Account Template

Provides information that the workflow activity has created a new account. By default, this template is enabled.

For example, the template provides this message:

```
New Account Information from IBM Security Identity Manager

The following new ITIM Service [ITIM] account has been created for you:

View Changes: http://localhost:80/itim/console
Process Reference: 8498649245880216244
Password: bAMI#gai
Account ID: myaccount
Owner Name: firstname lastname
Time of service provision: Jun 29, 2007 10:55:58 IST
```

The subject statement is:

```
<RE key="new_account_subject"/>
```

The plain text is:

```
 <RE key="account_created"><PARM>
  <RE key="service_name_with_profile_name">
  <PARM><JS>EmailContext.getAccountServiceName();</JS></PARM>
  <PARM><RE><KEY><JS>EmailContext.getAccountServiceProfileName();
  </JS></KEY></RE></PARM></RE></PARM></RE>
 <ITIMURL/>
 <RE key="processRef"/>: <JS>process.id;</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="TRANSACTION_ID_LABEL"/>: '
  + EmailContext.getTransactionId(); } </JS>
 <RE key="password"/>: <JS>EmailContext.getAccountPassword();</JS>
 <RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
 <RE key="accountOwnerName"/>:
  <JS>EmailContext.getAccountOwnerName();</JS>
 <RE key="timeofprovision"/>: <RE key="readOnlyDateFormat">
  <PARM><JS>(new Date()).getTime();</JS></PARM></RE>
 <JS>if (EmailContext.hasNewAccess()) { '<RE key="accountNewAccess"/>:
  <JS>EmailContext.getAccountNewAccessAsString();</JS>\n'; }</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="RETRIEVE_PASSWORD_TITLE"/>: '
   + EmailContext.getPasswordRetrievalUrl(); }</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="passwordExpireLabel"/>:
  <JS>if (EmailContext.getPasswordExpirePeriod() == 0)
  { '<RE key="passwordneverexpire"/>'; }
  else { EmailContext.getPasswordExpirePeriod(); }</JS>'; }</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<JS>if (EmailContext.getPasswordExpirePeriod() == 0)
  { '<RE key="additionalMsgForPwdRetrieval"/>'; }</JS>'; }</JS>
```

New Password Template

Provides information that there is a new password for an account. By default, this template is enabled.

For example, the template provides this message:

```
Account new password information

The following is your new password for account myaccount:

View Changes: http://localhost:9090/itim/console
Process Reference: 2855285841498421007
New Password: secret
Account ID: myaccount
Account Service: ITIM Service
Account Service Profile: ITIM
Owner Name: firstname lastname
Time of service provision: Apr 25, 2007 12:54:05 IST
```

The subject statement is:

```
<RE key="password_change_subject"/>
```

The plain text is:

```
 <RE><KEY><JS>if (EmailContext.getTransactionId() == '0')
  { 'newAccountPassword' } else { 'newAccountPasswordPickUp'; }
   </JS></KEY>
  <PARM><JS>process.subject;</JS></PARM></RE>
 <ITIMURL/>
 <RE key="processRef"/>: <JS>process.id;</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="TRANSACTION_ID_LABEL"/>: ' +
   EmailContext.getTransactionId(); }
  </JS>
 <RE key="newPassword"/>: <JS>EmailContext.getAccountPassword();</JS>
 <RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
 <RE key="accountService"/>:
  <JS>EmailContext.getAccountServiceName();</JS>
 <RE key="accountServiceProfile"/>: <RE><KEY>
  <JS>EmailContext.getAccountServiceProfileName();</JS></KEY></RE>
```

```
<RE key="accountOwnerName"/>:
 <JS>EmailContext.getAccountOwnerName();</JS>
<RE key="timeofprovision"/>: <RE key="readOnlyDateFormat">
 <PARM><JS>(new Date()).getTime();</JS></PARM></RE>
<JS>if (EmailContext.getTransactionId() != '0')
 { '<RE key="RETRIEVE_PASSWORD_TITLE"/>: '
 + EmailContext.getPasswordRetrievalUrl(); }</JS>
<JS>if (EmailContext.getTransactionId() != '0')
 { '<RE key="passwordExpireLabel"/>:
 <JS>if (EmailContext.getPasswordExpirePeriod() == 0)
 { '<RE key="passwordneverexpire"/>'; }
 else { EmailContext.getPasswordExpirePeriod(); }</JS>'; }</JS>

<JS>if (EmailContext.getTransactionId() != '0')
 { '<JS>if (EmailContext.getPasswordExpirePeriod() == 0)
 { '<RE key="additionalMsgForPwdRetrieval"/>'; }</JS>'; }</JS>
```

Restore Account Template

Provides information that an account has been restored. By default, this template is enabled.

For example, the template provides this message:

```
Restored Account Information from IBM Security Identity Manager

The following ITIM Service [ITIM] account has been restored:

View Changes: http://localhost:9090/itim/console
Process Reference: 2857890686820910405
New Password: secret
Account ID: myaccount
Owner Name: firstname lastname
Time Completed: Apr 25, 2007 01:04:08 IST
```

The subject statement is:

```
<RE key="restore_account_subject"/>
```

The plain text is:

```
  <RE key="restore_account"><PARM>
  <RE key="service_name_with_profile_name"><PARM>
  <JS>EmailContext.getAccountServiceName();</JS></PARM>
  <PARM><RE><KEY>
  <JS>EmailContext.getAccountServiceProfileName();
  </JS></KEY></RE></PARM></RE></PARM></RE>
 <ITIMURL/>
 <RE key="processRef"/>: <JS>process.id;</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="TRANSACTION_ID_LABEL"/>: '
  + EmailContext.getTransactionId(); } </JS>
 <RE key="newPassword"/>: <JS>EmailContext.getAccountPassword();</JS>
 <RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
 <RE key="accountOwnerName"/>:
  <JS>EmailContext.getAccountOwnerName();</JS>
 <RE key="timeCompleted"/>: <RE key="readOnlyDateFormat">
 <PARM>
  <JS>(new Date()).getTime();</JS></PARM></RE>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="RETRIEVE_PASSWORD_TITLE"/>: '
  + EmailContext.getPasswordRetrievalUrl(); }</JS>
 <JS>if (EmailContext.getTransactionId() != '0')
  { '<RE key="passwordExpireLabel"/>:
 <JS>if (EmailContext.getPasswordExpirePeriod() == 0)
  { '<RE key="passwordneverexpire"/>'; }
  else { EmailContext.getPasswordExpirePeriod(); }</JS>'; }
  </JS>

 <JS>if (EmailContext.getTransactionId() != '0')
  { '<JS>if (EmailContext.getPasswordExpirePeriod() == 0)
  { '<RE key="additionalMsgForPwdRetrieval"/>'; }</JS>'; }</JS>
```

Suspend Account Template

Provides information that an account is suspended. By default, this template is enabled.

For example, the template provides this message:

```
Your account has been suspended by IBM Security Identity Manager

The following AD Service (RFI) [ADProfile] account has been suspended:

View Changes: http://localhost:9090/itim/console
Process Reference: 2857497715286893521
Account ID: myaccount
Owner Name: firstname lastname
Time Completed: Apr 25, 2007 01:02:43 IST
```

The subject statement is:

```
<RE key="suspend_account_subject" />
```

The plain text is:

```
 <RE key="account_suspended"><PARM>
  <RE key="service_name_with_profile_name">
  <PARM><JS>EmailContext.getAccountServiceName();</JS></PARM>
  <PARM><RE><KEY><JS>EmailContext.getAccountServiceProfileName();
  </JS></KEY></RE></PARM></RE></PARM></RE>
 <ITIMURL/>
 <RE key="processRef"/>: <JS>process.id;</JS>
 <RE key="accountID"/>: <JS>EmailContext.getAccountUserId();</JS>
 <RE key="accountOwnerName"/>:
  <JS>EmailContext.getAccountOwnerName();</JS>
 <RE key="timeCompleted"/>: <RE key="readOnlyDateFormat"><PARM>
  <JS>(new Date()).getTime();</JS></PARM></RE>
```

Shared or Application identity management default messages

IBM Security Privileged Identity Manager provides default generic workflow messages.

Default shared or application identity management templates

All the shared or application identity management templates can be customized. IBM Security Privileged Identity Manager provides these default generic workflow notice templates:

Application Service Reconfiguration Template

The subject statement is:

```
<RE key="reconfigureServiceSubject"><PARM><JS>process.requesteeName
</JS></PARM><PARM>
<RE><KEY><JS>process.STATE_PREFIX + process.resultSummary;</JS></KEY>
</RE></PARM></RE>
```

The plain text is:

```
<RE key="reconfigureServiceCompletedMessage" />

<RE key="reconfigureServiceInformation" />

<RE key="itimUrl"/>:<ITIMURL/>
<RE key= "resultSummary"/>: <RE><KEY><JS>process.STATE_PREFIX +
process.resultSummary;</JS></KEY></RE>
<RE key="applicationServiceOwner"/>: <JS>process.requestorName;</JS>
<RE key="applicationService"/>: <JS>process.requesteeName;</JS>
<RE key="applicationServiceManager"/>: <JS>process.subject;</JS>
<RE key="host"/>: <JS>EmailContext.getTargetHose();</JS>
<RE key="targetName"/>: <JS>EmailContext.getTargetName();</JS>
<RE key="targetDisplayName"/>: <JS>EmailContext.getTargetDisplayName();</JS>
<RE key="restartOption"/>: <RE><KEY><JS>EmailContext.getRestartOptionKey()
;</JS></KEY></RE>
<RE key="loginID"/>: <JS>EmailContext.getLoginID();</JS>
<RE key="resourceUID"/>: <JS>EmailContext.getResourceUID();</JS>
<RE key="configTime"/>: <JS>EmailContext.getConfigTime();</JS>
<RE key="additionalInfo"/>: <JS>EmailContext.getResultDescription();</JS>
```

```
<RE key="csvResult"/>:
<JS>EmailContext.getCSVHeader();</JS>
<JS>EmailContext.toCSV();</JS>;
```

Credential Password Rotation Template

The subject statement is:

```
<RE key="passwordRotationSubject"><PARM><RE><KEY><JS>process.STATE_PREFIX +
process.resultSummary;</JS></KEY></RE></PARM></RE>
```

The plain text body is:

```
<RE key="passwordRotationCompletedMessage" />

<RE key="credentialInformation" />

<RE key="itimUrl"/>:<ITIMURL/>
<RE key="resultSummary"/>: <RE><KEY><JS>process.STATE_PREFIX +
process.resultSummary;</JS></KEY></RE>
<RE key="loginID"/>: <JS>EmailContext.getLoginID();</JS>
<RE key="resourceUID"/>: <JS>EmailContext.getResourceUID();</JS>
<RE key="passwordRotationTime"/>: <JS>EmailContext.
getPasswordRotationTime();</JS>
<RE key="additionalInfo"/>: <JS>EmailContext.getResultDescription();</JS>

<RE key="csvResult"/>:
<JS>EmailContext.getCSVHeader();</JS>
<JS>EmailContext.toCSV();</JS>
```

Access Batch Processing Complete Template

The subject statement is:

```
<RE key="access_request_completed_subject"/>
```

The plain text body is:

```
<RE key="access_request_number"/>: <JS>process.id;</JS>
<RE key="requestedAccess"/>:<JS>
var result = '';
var accessStatusList = accessRequestBatch.get().
getAccessStatusList(process.id);
for (var i = 0; i < accessStatusList.length; i++)
{
  var accessNameStatus = accessStatusList[i];
  if (i == (accessStatusList.length - 1)){
    result += accessNameStatus;
   }else{
     result += accessNameStatus + ', \n    ';
   }
}
return result;
</JS>
<RE key="accessJustification"/>: <JS>accessRequestBatch.get().
getJustification();</JS>
<RE key="access_submitted_by"/>: <JS>process.requestorName;</JS>
<RE key="access_submitted_for"/>: <JS>accessRequestBatch.get().
getRequestee();</JS>
<RE key="access_status"/>: <RE><KEY><JS>"RequestAuditData.Status." +
accessRequestBatch.get().getStatus(process.id);</JS></KEY></RE>
<RE key="access_submitted_date"/>: <RE key="readOnlyDateFormat"><PARM><JS>if
(process.scheduled != null) return process.scheduled.getTime();
else return '';</JS></PARM></RE>
<RE key="access_completed_date"/>: <RE key="readOnlyDateFormat"><PARM>
<JS>(new
Date()).getTime();</JS></PARM></RE>
<RE key="access_view_request_information"/>:
<JS>accessRequestBatch.get().getViewRequestInformationURL(process.id);</JS>
```

Access Batch Processing Start Template

The subject statement is:

```
<RE key="new_access_request_subject:/>
```

The plain text body is:

```
<RE key="access_request_number"/>: <JS>process.id;</JS>
<RE key="requestedAccess"/>: <JS>
var result = '';
var accessList = accessRequestBatch.get().getAccessList();
 for (var i = 0; i < accessList.length; i++)
 {
   var accessName = accessList[i];
   if (i == (accessList.length - 1)){
     result += accessName;
   }else{
     result += accessName + ', ';
   }
 }
 return result;
</JS>
<RE key="accessJustification"/>: <JS>accessRequestBatch.get().
getJustification();</JS>
<RE key=""access_submitted_by"/>: <JS>process.requestorName;</JS>
<RE key="access_submitted_for"/>: <JS>accessRequestBatch.get().
getRequestee();</JS>
<RE key="access_subimitted_date"/>: <RE key="readOnlyDateFormat">
<PARM><JS>(new)
Date()).getTime();</JS></PARM></RE>
<RE key="access_view_request_information"/>:
<JS>accessRequestBatch.get().getViewRequestInformationURL(process.id);</JS>
```

Chapter 9. Sample virtual appliance configuration response file

You can set your configuration parameters for the IBM Security Privileged Identity Manager virtual appliance in a response file. After you complete the response file, you can upload the response file to configure the virtual appliance in the advanced configuration mode.

```
##################################################
#
# Complete the initial configuration of IBM Security Privileged Identity Manager
# Appliance by using a response file.
# Update the response file with correct values and provide it during the advanced
# mode of Initial configuration wizard.
#
##################################################
#
# Appliance Administrator User Credentials
#
ispim.appliance.adminUserPwd=<admin user password>

#
# Session Recording Activation Detail
# If you want to activate session recording, provide the activation key.
# For example 12345-ABCDE-67890-FGHIJ-KLMNO
# Else, you can delete this line or leave this field blank.
#

ispim.session.recording.activation.key=

#
# Application Identity Management Activation Detail
# If you want to activate application identity management, provide the activation key.
# For example 12345-ABCDE-67890-FGHIJ-KLMNO
# Else, you can delete this line or leave this field blank.
#

ispim.appid.activation.key=

# Certificate Information
# If you want to use default certificate, then leave these fields blank.
# Else, if you want to generate your own self-signed certificate,
# ispim.root.ca.certificate.common.name is required. Other fields are optional.
# Zipcode should be an integer
# Country should be empty or of length 2 characters
#
ispim.root.ca.certificate.common.name=
ispim.root.ca.certificate.organization=
ispim.root.ca.certificate.organizational.unit=
ispim.root.ca.certificate.locality=
ispim.root.ca.certificate.state.province=
ispim.root.ca.certificate.zipcode=
ispim.root.ca.certificate.country=


#
# Identity Data store configuration Properties
#
ispim.identity.datastore.hostName=<hostname>
ispim.identity.datastore.port=50000
ispim.identity.datastore.adminUser=piminst
ispim.identity.datastore.adminUserPwd=<admin password>
ispim.identity.datastore.dbName=idmdb

#
# Enterprise Single Sign-On Data store configuration Properties
#
ispim.signon.datastore.hostName=<hostname>
ispim.signon.datastore.port=50000
ispim.signon.datastore.adminUser=piminst
ispim.signon.datastore.adminUserPwd=<admin password>
ispim.signon.datastore.dbName=essodb
```

```
#
# Session Recording Data store configuration Properties
#
ispim.session.recording.datastore.hostName=<hostname>
ispim.session.recording.datastore.port=50000
ispim.session.recording.datastore.adminUser=piminst
ispim.session.recording.datastore.adminUserPwd=<admin password>
ispim.session.recording.datastore.dbName=psrdb

#
# Directory Server configuration properties
#
ispim.ldap.hostName=<hostname>
ispim.ldap.port=389
ispim.ldap.organization.shortname=org
ispim.ldap.organization.name=Organization
ispim.ldap.bindDN=cn=root
ispim.ldap.bindDNPwd=<password>
ispim.ldap.dnLocation=dc=com
ispim.ldap.connection.type=non-ssl

#
# Mail Server configuration properties
#
ispim.mail.server=localhost
ispim.mail.from=admin@example.com
```

Chapter 10. Schema reference

IBM Security Privileged Identity Manager records auditable events into a set of database tables.

Auditing schema tables

You can use auditing schema to track credential management, credential pool management, credential lease management, and shared access policy management. The audit event schema has a common base event table, `audit_event`, which contains fields common to all audit events.

Separate tables are created for an event type only if that event type contains attributes, which are not generic enough to keep in a common table. As a rule, any element that is common to most audit events is kept in the `audit_event` container table. This design choice helps reduce the number of table joins when event data is queried.

The auditing event information is in the following tables:

Table 7. Auditing schema tables

Event Category	Table Name
Application ID management	No event-specific table
Credential management	No event-specific table
Credential Pool management	No event-specific table
Credential Lease management	`AUDIT_MGMT_LEASE` This table is used only if the action is Checkout or if the credential is a pool member.
Shared Access Policy management	No event-specific table

AUDIT_EVENT table

The `AUDIT_EVENT` table is common for all audit events. However, the value for some columns is different depending on the event. See the specific event for the column values.

Table 8. `AUDIT_EVENT` table

Column Name	Column Description	Data type
`ID`*	ID by which this event is identified. Primary key.	Numeric
`ITIM_EVENT_CATEGORY`*	IBM Security Privileged Identity Manager type of the event	Character (50)
`ENTITY_NAME`	Name of the IBM Security Privileged Identity Manager entities altered by this event. The size of this column is 100 characters, which assumes that the name of the entity that is being audited is 100 or less character long.	Character (1000)
`ENTITY_DN`	DN of the entity involved in this event.	Character (1000)
`ENTITY_TYPE`	Type of the IBM Security Privileged Identity Manager entity.	Character (50)
`ACTION`*	The value of this column depends on the event type. Each event type has a set of actions.	Character (25)

Table 8. AUDIT_EVENT table (continued)

Column Name	Column Description	Data type
WORKFLOW_PROCESS_ID	Process ID of the workflow initiated. This column is applicable to workflow operations.	Numeric
INITIATOR_NAME	The user ID of the ITIM account that submitted the request.	Character (1000)
INITIATOR_DN	The distinguished name of the ITIM account that submitted the request.	Character (1000)
INITIATOR_TYPE	PERSON - Indicates that the request was submitted by a person. SYSTEM - Indicates that the request was submitted by the IBM Security Privileged Identity Manager system.	Character (50)
INITIATOR_PERSON_DN	Distinguished name of the person who submitted the request.	Character (1000)
INITIATOR_PERSON_NAME	Name of the person who submitted the request.	Character (1000)
CONTAINER_NAME	Name of the container that holds the entity.	Character (1000)
CONTAINER_DN	Distinguished name of the container that holds the entity.	Character (1000)
RESULT_SUMMARY	The results of an event: Success Failure If the operation is submitted to workflow, this column indicates whether the operation was successfully submitted to workflow.	Character (25)
TIMESTAMP*	The time when the audit event occurs. It is also a start time of the operation.	Character (50)
COMMENTS	Description for this event.	Character (1000)
TIMESTAMP2	The time stamp for when the event was completed.	Character (50)

* Indicates the column is required and not null.

IBM Security Privileged Identity Manager authentication

This section describes the columns used by events related to IBM Security Privileged Identity Manager authentication operations.

Values for columns in the AUDIT_EVENT table

The following table describes the values of columns used by authentication operations in the AUDIT_EVENT table.

Table 9. Column values in the AUDIT_EVENT table

Column Name	Values
ITIM_EVENT_CATEGORY	IBM Security Privileged Identity Manager Authentication
ENTITY_TYPE	Entity type: ChallengeResponse BasicAuth
ACTION	Authentication getAuthenticatedObject

Table columns in the AUDIT_EVENT table

The following list shows the columns for each IBM Security Privileged Identity Manager authentication action in the AUDIT_EVENT table.

Autheticate

entity_name, entity_type, result_summary, initiator_name, initiator_dn, timestamp

getAuthenticatedObject

entity_name, entity_type, result_summary, initiator_name, initiator_dn, timestamp

Person management

This section describes the columns used by events related to Person management, such as add, modify, delete, suspend, transfer, and restore.

In addition to the AUDIT_EVENT table, these tables are used by person management events: AUDIT_MGMT_TARGET, AUDIT_MGMT_ACCESS_REQUEST, AUDIT_MGMT_OBLIGATION, AUDIT_MGMT_OBLIGATION_ATTRIB, and AUDIT_MGMT_OBLIGATION_RESOURCE.

AUDIT_MGMT_TARGET table

The AUDIT_MGMT_TARGET table is used if the action is Transfer.

Table 10. AUDIT_MGMT_TARGET table

Column Name	Column Description	Data type
EVENT_ID*	Identification that is assigned to the event. References AUDIT_EVENT (ID).	Numeric
TARGET_ENTITY_NAME	The name of container to which the person is being transferred. Applicable if action=Transfer	Character (1000)
TARGET_ENTITY_DN	The DN of container to which the person is being transferred. Applicable if action=Transfer	Character (1000)
TARGET_ENTITY_TYPE	The type of container to which the person is being transferred.	Character (50)

* Indicates the column is required and not null.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the person management events in the AUDIT_EVENT table.

Table 11. Values for columns in the AUDIT_EVENT table

Column Name	Value
ITIM_EVENT_CATEGORY	Person Management.
ENTITY_NAME	Name of the person.
ENTITY_DN	Distinguished name of the person.
ENTITY_TYPE	Type of person, such as person, business person, or custom person.
INITIATOR_NAME	The user ID of the ITIM account that submitted the request.
INITIATOR_DN	The distinguished name of the ITIM account that submitted the request.

Table 11. Values for columns in the AUDIT_EVENT table (continued)

Column Name	Value
INITIATOR_TYPE	PERSON - Indicates that the request was submitted by a person. SYSTEM - Indicates that the request was submitted by the IBM Security Privileged Identity Manager system.
INITIATOR_PERSON_DN	Distinguished name of the person who submitted the request.
INITIATOR_PERSON_NAME	Name of the person who submitted the request.
CONTAINER_NAME	Name of the container that holds the entity.
CONTAINER_DN	Distinguished name of the container that holds the entity.
WORKFLOW_PROCESS_ID	Process ID of the initiated workflow.
RESULT_SUMMARY	Result of operation: Submitted – submitted to workflow successfully
ACTION	Types of actions: Add – add a person Modify – modify a person Delete – delete a person Suspend – suspend a person Restore – restore a person Transfer – transfer a person

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management event in the AUDIT_EVENT table.

Add Person event

```
entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Delete Person event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Modify Person event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Restore Person event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Suspend Person event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, container_name, container_dn, timestamp, result_summary

Transfer Person event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, ,workflow_process_id, container_name, container_dn, timestamp, result_summary

From **AUDIT_MGMT_TARGET** table: target_entity_name, target_entity_dn

Self-Register event

entity_name, entity_type, workflow_process_id, container_name, container_dn, timestamp, result_summary

Table columns for person management in the AUDIT_MGMT_ACCESS_REQUEST table

The following list shows the columns for all person management event in the AUDIT_MGMT_ACCESS_REQUEST table.

- Event_ID
- Wokflow_Process_Id
- Action
- Access_Obligations_Ids
- Status
- Completed_Date

Table columns for person management in the AUDIT_MGMT_OBLIGATION table

The following list shows the columns for all person management event in the AUDIT_MGMT_OBLIGATION table.

- Event_ID
- Id
- Obligation_Type
- System_Generated

Delegate authority

This section describes events related to delegate authority, such as add and modify.

AUDIT_MGMT_DELEGATE table

The AUDIT_MGMT_DELEGATE table is used if the action is to delegate a member.

Table 12. AUDIT_MGMT_DELEGATE table

Column Name	Column Description	Data type
EVENT_ID*	ID by which this event is identified. References AUDIT_EVENT (ID).	Numeric
DELEGATE_NAME	The name of the account to which authorities are delegated.	Character (1000)
DELEGATE_DN	The DN of the account to which authorities are delegated.	Character (1000)
DELEGATE_START_TIME	Start time of the delegation.	Character (1000)

Table 12. AUDIT_MGMT_DELEGATE table (continued)

Column Name	Column Description	Data type
DELEGATE_END_TIME	End time of the delegation.	Character (1000)

* Indicates the column is required and not null.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Person management operations in the AUDIT_EVENT table.

Table 13. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Delegate authority.
entity_name	Name of the account whose rights are being delegated.
entity_dn	Distinguished name of the account whose rights are being delegated.
entity_type	Account.
workflow_process_id	Process ID of the initiated workflow.
result_summary	Result of operation: Submitted – submitted to workflow successfully
Action	Types of actions: Add – Delegate authority Modify – Modify a delegate

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add Delegate event**

  entity_name, entity_dn, initiator_name, initiator_dn, timestamp, result_summary

  From Audit_Delegate table:

  delegate_name, delegate_dn, delegate_starttime, delegate_endtime

- **Modify Delegate event**

  entity_name, entity_dn, initiator_name, initiator_dn, timestamp, result_summary

  From Audit_Delegate table:

  delegate_name, delegate_dn, delegate_starttime, delegate_endtime

Policy management

This section describes events related to IBM Security Privileged Identity Manager polices, such as password policies.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the policy management events in the AUDIT_EVENT table.

Table 14. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Policy Management.
entity_name	Name of the policy.
entity_dn	Distinguished name of the policy.
entity_type	Types of policy entities: PasswordPolicy – A password policy specifies a set of rules that all passwords for one or more services must conform.
Action	Types of actions: Add – Add a policy Modify – Modify a policy Delete – Delete a policy

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each policy management event in the AUDIT_EVENT table.

Add Host Selection Policy event

```
entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Modify Host Selection Policy event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Delete Host Selection Policy event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Add Provisioning Policy event

```
entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Modify Provisioning Policy event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Delete Provisioning Policy event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary
```

Enforce Entire Provisioning Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, container_name, container_dn, timestamp, result_summary

Save Draft Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Commit Draft Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, container_name, container_dn, timestamp, result_summary

Delete Draft Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Add Identity Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Modify Identity Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Delete Identity Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Add Password Policy event

entity_name, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Modify Password Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp,result_summary

Delete Password Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Add Separation of Duty Policy event

entity_name, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, container_name, container_dn, timestamp, result_summary

Modify Separation of Duty Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,

initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Delete Separation of Duty Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Evaluate Separation of Duty Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Exempt a Violation for a Separation of Duty Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary, comments

Revoke an Exemption for a Separation of Duty Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary, comments

Add Recertification Policy event

entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Modify Recertification Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Delete Recertification Policy event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
workflow_process_id, container_name, container_dn, timestamp,
result_summary

Enforce Policy Import event

itim_event_category, action, workflow_process_id, initiator_name,
initiator_dn, initiator_type, initiator_person_dn,
initiator_person_name, result_summary

ACI management

This section describes the columns used by events related to IBM Security Privileged Identity Manager access control information (ACI).

In addition to the AUDIT_EVENT table, the AUDIT_MGNT_TARGET table is used by ACI management events.

AUDIT_MGMT_TARGET table

The AUDIT_MGMT_TARGET table is used if the action is Add Member or Remove.

Table 15. AUDIT_MGMT_TARGET table

Column Name	Column Description	Value Type	Required?
event_id	ID by which this event is identified. This column contains the foreign key to the ID column of the audit_event table.	long	Yes
target_entity_name	Name of the target ACI for Action = AddAuthOwneror Action=DeleteAuthOwner.	string	Yes for action = AddAuthOwner or Action=DeleteAuthOwner

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the policy management operations in the AUDIT_EVENT table.

Table 16. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	ACI Management.
entity_name	Name of the ACI.
entity_dn	Distinguished name of the ACI.
entity_type	Types of policy entities: aci – Access control list
action	Types of actions: Add – Add the ACI Modify – Modify the ACI Delete – Delete the ACI AddAuthorizationOwner – Add an authorization owner DeleteAuthorizationOwner – Delete an authorization owner

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each Person management action in the AUDIT_EVENT table.

- **Add ACI event**

  entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Modify ACI event**

  entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Delete ACI event**

  entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Add Authorization Owner event**

  entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_target: target_entity_name

- **Delete Authorization Owner event**

  `entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary`

  From `audit_mgmt_target`: `target_entity_name`

Access request management

Access request management describes the audit data that supports the viewing of access requests that are submitted through the Identity Service Center user interface.

In addition to the `AUDIT_EVENT` table, access request management events use the following tables.

- `AUDIT_MGMT_ACCESS_REQUEST`
- `AUDIT_MGMT_OBLIGATION`
- `AUDIT_MGMT_OBLIGATION_ATTRIB`
- `AUDIT_MGMT_OBLIGATION_RESOURCE`
- `AUDIT_MGMT_MESSAGE`

AUDIT_MGMT_ACCESS_REQUEST table

The **`AUDIT_MGMT_ACCESS_REQUEST`** table contains information about account, group, person, and role provisioning that is submitted through the Administrative console, Self-service user interface, and Identity Service Center user interface.

The **`AUDIT_MGMT_ACCESS_REQUEST`** table includes extra audit data that is related to rows in the **`AUDIT_EVENT`** table for which the **`ITIM_EVENT_CATEGORY`** column contains these values: **`PersonManagement`**, **`AccountManagement`**, **`AccessManagement`**, **`OrgRoleManagement`**, and **`AccessRequest`**.

Table 17. **`AUDIT_MGMT_ACCESS_REQUEST`** *table for access request management*

Column name	Column description	Data type
`EVENT_ID`*	Identifier that is assigned to this event. References **`AUDIT_EVENT (ID)`**.	Numeric
`WORKFLOW_PROCESS_ID`*	Identifier of the workflow process to which this additional audit data is related.	Numeric
`ACTION`*	The supported actions are `ADD`, `MODIFY`, `CHANGE`, `DELETE`, `SUSPEND`, `RESTORE`, `TRANSFER`, `CHANGE_PASSWORD`, `ADDMEMBER`, `REMOVEMEMBER`, and `SELF_REGISTER`.	Character (25)
`PERSON_NAME` **	Name of the person for whom the access request was submitted.	Character (1000)
`PERSON_DN` **	Distinguished name of the person for whom the access request was submitted.	Character (1000)
`ACCESS_CATALOG_ID` **	Access catalog identifier of the service, group, or role for which the access request was submitted.	Numeric
`ACCESS_CATALOG_NAME`**	Access catalog name of the service, group, or role for which the access request was submitted.	Character (1000)
`ACCESS_CATALOG_DESCRIPTION`**	Access catalog description of the service, group, or role for which the access request was submitted.	Character (1000)

Table 17. **AUDIT_MGMT_ACCESS_REQUEST** *table for access request management (continued)*

Column name	Column description	Data type
ACCESS_CATALOG_CATEGORY**	Access catalog description of the service, group, or role for which the access request was submitted.	Character (1000)
ACCESS_CATALOG_ICON **	URL of the access catalog icon of the service, group, or role for which the access request was submitted.	Character (1000)
ACCESS_CATALOG_BADGE_1 **	First access catalog badge of the service, group, or role for which the access request was submitted.	Character (3000)
ACCESS_CATALOG_BADGE_2 **	Second access catalog badge of the service, group, or role for which the access request was submitted.	Character (3000)
ACCESS_CATALOG_BADGE_3 **	Third access catalog badge of the service, group, or role for which the access request was submitted.	Character (3000)
ACCESS_CATALOG_BADGE_4 **	Fourth access catalog badge of the service, group, or role for which the access request was submitted.	Character (3000)
ACCESS_CATALOG_BADGE_5 **	Fifth access catalog badge of the service, group, or role for which the access request was submitted.	Character (3000)
ACCESS_OBLIGATION_IDS*	List of obligation IDs separated by semicolons that identifies the obligations that must be fulfilled for the access request.	Character (4000)
SERVICE_NAME	Name of the service for which the account or access request was submitted.	Character (1000)
STATUS*	Status of the access request. The **STATUS** contains one of the following values. FULFILLED NOT_FULFILLED PENDING	Character (25)
COMPLETED_DATE*	Date and time when the access request is completed or canceled.	Character (50)

* Indicates the column is required and not null.

** Indicates the column is null if the event category is not **AccessRequest**.

Note: The **AUDIT_MGMT_ACCESS_REQUEST** table contains multiple rows that have the same **WORKFLOW_PROCESS_ID** column value if there is more than one access that is associated with the corresponding request.

AUDIT_MGMT_OBLIGATION table

The **AUDIT_MGMT_OBLIGATION** table contains information about obligations that are related to access requests submitted through the Administrative console, Self-service user interface, and Identity Service Center user interface.

The **AUDIT_MGMT_OBLIGATION** table contains the following columns.

Table 18. **AccessRequest** *values for the* **AUDIT_MGMT_OBLIGATION** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**	Numeric
ID*	Identifier of the activity. The value of this column serves as a foreign key for the **AUDIT_MGMT_OBLIGATION_ATTRIB** and **AUDIT_MGMT_OBLIGATION_RESOURCE** tables.	Numeric
PERSON_DN*	Distinguished name of the person for whom the access request was submitted to which the obligation is related.	Character (1000)
OBLIGATION_TYPE*	Type of the obligation. CREATE_ACCOUNT, MODIFY_ACCOUNT, DELETE_ACCOUNT, SUSPEND_ACCOUNT, and RESTORE_ACCOUNT SET_SYNCPASSWORD, SELECT_ACCOUNTS, and CHANGE_PASSWORD CREATE_PERSON, MODIFY_PERSON, DELETE_PERSON, SUSPEND_PERSON, RESTORE_PERSON, TRANSFER_PERSON, and SELF_REGISTER.	Character (50)
SYSTEM_GENERATED*	Indicates whether the obligation was system-generated. Values are Y or N	Character (1)
ACCESS_FORM_TEMPLATE	Form template in JSON format that presents related attributes in the **CREATE_ACCOUNT** obligation. Form template will be shown only if the create account request is submitted from Identity Service Center.	Long character (100 K)

* Indicates the column is required and not null.

AUDIT_MGMT_OBLIGATION_ATTRIB table

The **AUDIT_MGMT_OBLIGATION_ATTRIB** table contains information about attributes of the obligations that are related to access requests submitted through the Identity Service Center user interface.

The **AUDIT_MGMT_OBLIGATION_ATTRIB** table contains the following columns.

Table 19. **AccessRequest** *values for the* **AUDIT_MGMT_OBLIGATION_ATTRIB** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**	Numeric
OBLIGATION_ID*	Identifier of the obligation to which the resources are related.	Numeric
ATTRIBUTE_NAME*	Name of an attribute that is associated to the obligation.	Character (225)
ATTRIBUTE_VALUE*	Data value of an attribute that is associated to the obligation.	Character (4000)
SEQUENCE_NO*	A generated numeric value that starts at 1 and increments by 1. It enables the persistence of an attribute name with multiple attribute values.	SMALLINT

* Indicates the column is required and not null.

AUDIT_MGMT_OBLIGATION_RESOURCE table

The **AUDIT_MGMT_OBLIGATION_RESOURCE** table contains information about the obligation resource attributes.

The **AUDIT_MGMT_OBLIGATION_RESOURCE** table contains the following columns.

Table 20. **AccessRequest** *values for the* **AUDIT_MGMT_OBLIGATION_RESOURCE** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**	Numeric
OBLIGATION_ID*	Identifier of the obligation to which the resources are related.	Numeric
RESOURCE_TYPE*	The value can be **ACCOUNT**, **PERSON**.	Character (50)
RESOURCE_NAME*	Name of the resource to which access is requested.	Character (1000)
RESOURCE_DN*	Distinguished name of the resource to which access is requested.	Character (1000)

* Indicates the column is required and not null.

AUDIT_MGMT_MESSAGE table

The **AUDIT_MGMT_MESSAGE** table contains messages that are related to access requests. It includes extra audit data that is related to rows in the **AUDIT_EVENT** table for which the **ITIM_EVENT_CATEGORY** column contains the value **AccessRequest**.

The **AUDIT_MGMT_MESSAGE** table contains the following columns.

Table 21. **AUDIT_MGMT_MESSAGE** *table for access request management*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**	Numeric
WORKFLOW_PROCESS_ID*	Identifier of the workflow process to which this additional audit data is related.	Numeric
MESSAGE*	Message that is related to the request.	Character (1000)

* Indicates the column is required and not null.

Note: The **AUDIT_MGMT_MESSAGE** table contains multiple rows that have the same **WORKFLOW_PROCESS_ID** column value if there is more than one message that is associated with the corresponding request.

Values for columns in the AUDIT_EVENT table that is used by access request management

The AUDIT_EVENT table is common for all audit events. However, the value for some columns is different depending on the event. See the specific event for the column values.

Table 22. AUDIT_EVENT table for access request management

Column Name	Column Description	Data type
ID*	ID by which this event is identified. Primary key.	Numeric

Table 22. AUDIT_EVENT table for access request management (continued)

Column Name	Column Description	Data type
ITIM_EVENT_CATEGORY*	**AccessRequest**.	Character (50)
ACTION*	ADD	Character (25)
WORKFLOW_PROCESS_ID	The identifier of the request.	Numeric
INITIATOR_NAME	The user ID of the ITIM account that submitted the request.	Character (1000)
INITIATOR_DN	The distinguished name of the ITIM account that submitted the request.	Character (1000)
INITIATOR_TYPE	PERSON - Indicates that the request was submitted by a person. SYSTEM - Indicates that the request was submitted by the IBM Security Privileged Identity Manager system.	Character (50)
INITIATOR_PERSON_DN	Distinguished name of the person who submitted the request.	Character (1000)
INITIATOR_PERSON_NAME	Name of the person who submitted the request.	Character (1000)
RESULT_SUMMARY	The status of the request. The **RESULT_SUMMARY** contains one of the following values. 0 - PENDING 1 - NOT_FULFILLED 2 - PARTIALLY_FULFILLED 3 - FULFILLED	Character (25)
TIMESTAMP*	The time stamp for when the request was submitted.	Character (50)
COMMENTS	The justification for the request.	Character (1000)
TIMESTAMP2	The time stamp for when the request was completed.	Character (50)

* Indicates the column is required and not null.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each IBM Security Privileged Identity Manager access request management action in the AUDIT_EVENT table.

Request access event

```
id, itim_event_category, action, workflow_process_id,
initiator_name, initiator_dn, initiator_type, initiator_person_dn,
initiator_person_name, result_summary, timestamp, comments,
timestamp2
```

Manual activity events

This section describes events that are related to manual activities. Some examples of manual activities include approvals, requests for information, and work orders.

In addition to the AUDIT_EVENT table, the AUDIT_MGMT_ACTIVITY and AUDIT_MGMT_PARTICIPANT tables are used by manual activity events

Create manual activity event

The create manual activity event is audited when a manual activity is created.

The following are the manual activities that are created by the system:

Approval

Request for information
Work order
User recertification
Compliance alert
Separation of duty approval

In addition to the AUDIT_EVENT table, create manual activity events use the following tables.

- AUDIT_MGMT_ACTIVITY
- AUDIT_MGMT_PARTICIPANT

AUDIT_MGMT_ACTIVITY table:

The **AUDIT_MGMT_ACTIVITY** table contains information about the manual activity that was created and its status.

The **AUDIT_MGMT_ACTIVITY** table contains the following columns.

Table 23. Create manual activity values for the **AUDIT_MGMT_ACTIVITY** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**	Numeric
ROOT_WORKFLOW_PROCESS_ID*	Identifier of the root workflow process in which the manual activity was created.	Numeric
WORKFLOW_PROCESS_ID*	Identifier of the workflow process in which the manual activity was created.	Numeric
ID*	Identifier of the manual activity. The value of this column serves as a foreign key for the **AUDIT_MGMT_ACTIVITY_PARTICIPANT** table.	Numeric
WORKITEM_ID*	Identifier of the work item that represents the current participant assignments and the due date for the manual activity. **Note:** The value of this column is updated when the manual activity is escalated. It then becomes the identifier of the work item that represents the escalation participant assignments and the due date.	Numeric
TYPE*	Type of the activity. APPROVAL COMPLIANCE_ALERT RFI SOD USER_RECERTIFICATION WORK_ORDER	Character (25)
NAME *	Name of the manual activity. The administrator can specify a translated label by using the syntax `$labelKey`.	Character (1000)
PERSON_NAME *	Name of the person for whom the manual activity was created.	Character (1000)

Table 23. Create manual activity values for the **AUDIT_MGMT_ACTIVITY** *table (continued)*

Column name	Column description	Data type
PERSON_DN	Distinguished name of the person for whom the manual activity was created. The **PERSON_DN** is populated only if the person exists at the time that the activity is created.	Character (1000)
SERVICE_NAME	Name of the service for which the manual activity was created.	Character (1000)
SERVICE_DN	Distinguished name of the service for which the manual activity was created.	Character (1000)
ACCOUNT_USERID	User ID of the account for which the manual activity was created.	Character (1000)
ACCOUNT_DN	Distinguished name of the account for which the manual activity was created. The **ACCOUNT_DN** is populated only if the account exists at the time that the activity was created.	Character (1000)
ACCESS_CATALOG_ID	Identifier of the access catalog item for which the manual activity was created.	Numeric
ACCESS_CATALOG_NAME	Name of the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_DESCRIPTION	Description of the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_CATEGORY	Category of the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_ICON	URL of the access catalog icon for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_BADGE_1	Text and style of the first badge for the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_BADGE_2	Text and style of the second badge for the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_BADGE_3	Text and style of the third badge for the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_BADGE_4	Text and style of the fourth badge for the access catalog item for which the manual activity was created.	Character (1000)
ACCESS_CATALOG_BADGE_5	Text and style of the fifth badge for the access catalog item for which the manual activity was created.	Character (1000)
CREATED_DATE*	Date and time when the manual activity was created.	Character (50)
ESCALATED_DATE	Date and time when the manual activity was escalated. **Note:** This column is not set when the manual activity is created.	Character (50)

Table 23. Create manual activity values for the **AUDIT_MGMT_ACTIVITY** *table (continued)*

Column name	Column description	Data type
DUE_DATE	Date and time when the activity escalates or times out if it is already escalated, or times out if no escalation participants exist. **Note:** This column is updated to set the new due date and time for the escalation participants.	Character (50)
COMPLETED_DATE	Date and time when the manual activity is completed, canceled, or times out. This column is not set when the manual activity is created.	Character (50)
COMPLETION_CODES	Valid completion or result codes for the manual activity.	Character (50)
COMPLETION_CODE	Completion or result code that is specified by the participant when the manual activity is completed. **Note:** This column is not set when the manual activity is created.	Character (50)
STATUS*	Status of the manual activity. APPROVED CANCELED FAILED PENDING REJECTED SKIPPED SUCCESS TIMED_OUT_FAILED TIMED_OUT_SUCCESS WARNING	Character (25)
JUSTIFICATION	Justification that is specified when the user submitted the request that created the manual activity.	Character (4000)
COMMMENTS	Comments that are specified by the participant that completed the activity. This column is not set when the manual activity is created.	Character (1000)

* Indicates the column is required and not null.

Note: The columns for a specified row in the **AUDIT_MGMT_ACTIVITY** table might change as the manual activity changes from one state to another.

- If a participant completes the manual activity, the COMPLETED_DATE column is updated with the date and time that the manual activity was completed. The COMPLETION_CODE column is updated with the completion or result code that is specified by the participant at the time of completion. The COMMENTS column is updated with any comments specified by the participant who completed the activity. The STATUS column is updated accordingly.
- If the manual activity is canceled, skipped, or times out, the COMPLETED_DATE column is updated to contain the date and time of the occurrence. The STATUS column is updated accordingly.

- If the manual activity escalates, the EVENT_ID column remains the unchanged. The WORKITEM_ID column is updated to contain the identifier of the work item that represents the escalation participant assignments and the due date. The ESCALATED_DATE column is updated to contain the date and time when the manual activity was escalated. The DUE_DATE column is updated to contain the new due date for the escalation participant assignments.

AUDIT_MGMT_PARTICIPANT table:

The **AUDIT_MGMT_PARTICIPANT** table contains information about participants of manual activities. It includes extra audit data that is related to rows in the **AUDIT_EVENT** table for which the **ITIM_EVENT_CATEGORY** column contains the value **ManualActivity**.

The **AUDIT_MGMT_PARTICIPANT** table contains the following columns.

Table 24. Create manual activity values for the **AUDIT_MGMT_PARTICIPANT** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**.	Numeric
ROOT_WORKFLOW_PROCESS_ID*	Identifier of the root workflow process for the manual activity to which the participant is assigned.	Numeric
WORKFLOW_PROCESS_ID*	Identifier of the workflow process for the manual activity to which the participant is assigned.	Numeric
ACTIVITY_ID*	Identifier of the activity. References **AUDIT_MGMT_ACTIVITY (ID)**.	Numeric
WORKITEM_ID*	Identifier of the work item that represents the current participant assignments and the due date for the manual activity.	Numeric
PERSON_NAME	Name of the person who is a participant of the manual activity.	Character (1000)
PERSON_DN	Distinguished name of the person who is a participant of the manual activity.	Character (1000)
ACCOUNT_USERID	User ID of the IBM Security Privileged Identity Manager account that is a participant of the manual activity.	Character (1000)
ACCOUNT_DN	Distinguished name of the IBM Security Privileged Identity Manager that is a participant of the manual activity.	Character (1000)
STATUS*	Status of the assignment for the manual activity that is assigned to the participant. CANCELED COMPLETED COMPLETED_OTHER ESCALATED PENDING SKIPPED TIMED_OUT	Character (25)

* Indicates the column is required and not null.

Note: The **AUDIT_MGMT_PARTICIPANT** table contains multiple rows that have the same ACTIVITY_ID column value if there is more than one participant for the corresponding activity.

The rows for a specific **ACTIVITY_ID** might change as the manual activity changes from one state to another:

- Initially the rows represent the original participants for the manual activity.
- If the manual activity escalates, the STATUS column for the original participant rows is updated to ESCALATED. New rows that represent the escalation participants are added with the STATUS column set to PENDING.
- If a participant completes the manual activity, the STATUS column is updated to COMPLETED. Other participants for which the STATUS was PENDING, are updated to COMPLETED_OTHER.
- If the manual activity is canceled or times out, the rows for the participants for which the STATUS is PENDING is updated to CANCELED or TIMED_OUT.

Values for columns in the AUDIT_EVENT table for the create manual activity event:

The AUDIT_EVENT table is common for all audit events. However, the value for some columns is different depending on the event. See the specific event for the column values.

Table 25. AUDIT_EVENT table for the create manual activity event

Column Name	Column Description	Data type
ID*	ID by which this event is identified. Primary key.	Numeric
ITIM_EVENT_CATEGORY*	**ManualActivity**.	Character (50)
ACTION*	Create	Character (25)
TIMESTAMP*	The time stamp for when the manual activity was created.	Character (50)

* Indicates the column is required and not null.

Escalate manual activity event

The escalate manual activity event is audited when a manual activity is escalated.

Normal escalation occurs when the activity is not completed by the due date. Escalation also occurs when the participant for the activity cannot be resolved. In this case, the activity is created in an escalated state.

In addition to the AUDIT_EVENT table, escalate manual activity events use the following tables.

- AUDIT_MGMT_ACTIVITY
- AUDIT_MGMT_PARTICIPANT

AUDIT_MGMT_ACTIVITY table:

The **AUDIT_MGMT_ACTIVITY** table is modified if a manual activity event is escalated normally or if the participants that are assigned cannot be resolved.

Normal escalation

If the manual activity was not completed by the original participants by the due date and was reassigned to the escalation participants the following columns are changed in the **AUDIT_MGMT_ACTIVITY** table. That table was created when the create manual activity event occurred.

Table 26. Escalate manual activity values for the **AUDIT_MGMT_ACTIVITY** *table*

Column name	Column description	Data type
WORKITEM_ID*	Identifier of the work item that represents the escalation participant assignments and the due date for the manual activity.	Numeric
ESCALATED_DATE	Date and time when the manual activity was escalated. **Note:** This column is not set when the manual activity is created.	Character (50)
DUE_DATE	Date and time when the activity is due or times out. **Note:** This column is updated to set the new due date and time for the escalation participants.	Character (50)

* Indicates the column is required and not null.

Participants cannot be resolved

If the participants for a manual activity cannot be resolved, the activity is created in an escalated state. An example of unresolved participants is an activity that is assigned to a group or role that has no members. The content of the **AUDIT_MGMT_ACTIVITY** table is the same as the create manual activity event with the following modifications.

Table 27. Escalate manual activity values for the **AUDIT_MGMT_ACTIVITY** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that corresponds to the ID column in the AUDIT_EVENT table for the escalate manual activity event.	Numeric
ESCALATED_DATE	Date and time when the manual activity was escalated.	Character (50)
DUE_DATE	Date and time that is set for the escalation participants.	Character (50)

* Indicates the column is required and not null.

AUDIT_MGMT_PARTICIPANT table:

The **AUDIT_MGMT_PARTICIPANT** table contains information about participants of manual activities. It includes extra audit data that is related to rows in the **AUDIT_EVENT** table for which the **ITIM_EVENT_CATEGORY** column contains the value **ManualActivity**.

The **AUDIT_MGMT_PARTICIPANT** table contains the following columns.

Table 28. Escalate manual activity event values for the **AUDIT_MGMT_PARTICIPANT** *table*

Column name	Column description	Data type
EVENT_ID*	Identifier that is assigned to this event. References **AUDIT_EVENT (ID)**.	Numeric
ROOT_WORKFLOW_PROCESS_ID*	Identifier of the root workflow process for the manual activity to which the escalation participant is assigned.	Numeric
WORKFLOW_PROCESS_ID*	Identifier of the workflow process for the manual activity to which the escalation participant is assigned.	Numeric
ACTIVITY_ID*	Identifier of the activity. References **AUDIT_MGMT_ACTIVITY (ID)**.	Numeric
WORKITEM_ID*	Identifier of the work item that represents the escalation participant assignments and the due date for the manual activity.	Numeric
PERSON_NAME	Name of the person who is an escalation participant of the manual activity.	Character (1000)
PERSON_DN	Distinguished name of the person who is an escalation participant of the manual activity.	Character (1000)
ACCOUNT_USERID	User ID of the IBM Security Privileged Identity Manager account that is an escalation participant of the manual activity.	Character (1000)
ACCOUNT_DN	Distinguished name of the IBM Security Privileged Identity Manager that is an escalation participant of the manual activity.	Character (1000)
STATUS*	Status of the manual activity assignment for the escalation participant. CANCELED COMPLETED COMPLETED_OTHER ESCALATED PENDING SKIPPED TIMED_OUT	Character (25)

* Indicates the column is required and not null.

Note: The **AUDIT_MGMT_PARTICIPANT** table contains multiple rows that have the same ACTIVITY_ID column value if there is more than one participant for the corresponding activity.

The rows for a specific **ACTIVITY_ID** might change as the manual activity changes from one state to another:

- Initially the rows represent the original participants for the manual activity.
- If the manual activity escalates, the STATUS column for the original participant rows is updated to ESCALATED. New rows that represent the escalation participants are added with the STATUS column set to PENDING.
- If a participant completes the manual activity, the STATUS column is updated to COMPLETED. Other participants for which the STATUS was PENDING, are updated to COMPLETED_OTHER.

- If the manual activity is canceled or times out, the rows for the participants for which the STATUS is PENDING is updated to CANCELED or TIMED_OUT.

Values for columns in the AUDIT_EVENT table for the escalate manual activity event:

The AUDIT_EVENT table is common for all audit events. However, the value for some columns is different depending on the event. See the specific event for the column values.

Table 29. AUDIT_EVENT table for the escalate manual activity event

Column Name	Column Description	Data type
ID*	ID by which this event is identified. Primary key.	Numeric
ITIM_EVENT_CATEGORY*	**ManualActivity**.	Character (50)
ACTION*	Escalate	Character (25)
TIMESTAMP*	The time stamp for when the manual activity was escalated.	Character (50)

* Indicates the column is required and not null.

Table columns used in the AUDIT_Event table

The following list shows the columns for each IBM Security Privileged Identity Manager manual activity event in the AUDIT_EVENT table.

Create manual activity event

```
id, itim_event_category, action, timestamp
```

Escalate manual activity event

```
id, itim_event_category, action, timestamp
```

Lifecycle rule events

When a lifecycle rule is run, information about who submitted the request is audited. If the lifecycle rule creates other root workflow processes, a lifecycle rule event is audited for each created root workflow process.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the lifecycle rule events in the AUDIT_EVENT table.

Table 30. AUDIT_EVENT table for lifecycle rule events

Column Name	Column Description	Data type
ID*	ID by which this event is identified. Primary key.	Numeric
ITIM_EVENT_CATEGORY*	**LifecycleRule**.	Character (50)
ACTION*	Run	Character (25)
WORKFLOW_PROCESS_ID	The identifier of the root workflow process in which the lifecycle is run or its created workflow processes.	Numeric
INITIATOR_NAME	The user ID of the ITIM account that ran the lifecycle rule.	Character (1000)
INITIATOR_DN	The distinguished name of the ITIM account that ran the lifecycle rule.	Character (1000)
INITIATOR_TYPE	PERSON - Indicates that the lifecycle rule was run by a person. SYSTEM - Indicates that the lifecycle rule was run by the IBM Security Privileged Identity Manager system.	Character (50)

Table 30. AUDIT_EVENT table for lifecycle rule events (continued)

Column Name	Column Description	Data type
INITIATOR_PERSON_DN	Distinguished name of the person who ran the lifecycle rule.	Character (1000)
INITIATOR_PERSON_NAME	Name of the person who ran the lifecycle rule.	Character (1000)
TIMESTAMP*	The time stamp for when the lifecycle rule was run.	Character (50)

* Indicates the column is required and not null.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each IBM Security Privileged Identity Manager lifecycle rule event in the AUDIT_EVENT table.

lifecycle rule

```
id, itim_event_category, action, workflow_process_id,
initiator_name, initiator_dn, initiator_type, initiator_person_dn,
initiator_person_name, timestamp
```

Account management

These AUDIT_EVENT columns are used by events that are related to account management, such as add, modify, suspend, restore, delete, admin change password, password pickup, and adopt.

In addition to the AUDIT_EVENT table, these tables are used by account management events: AUDIT_MGMT_PROVISIONING, AUDIT_MGMT_ACCESS_REQUEST, AUDIT_MGMT_OBLIGATION, AUDIT_MGMT_OBLIGATION_ATTRIB, and AUDIT_MGMT_OBLIGATION_RESOURCE.

AUDIT_MGMT_PROVISIONING table

Table 31. AUDIT_MGMT_PROVISIONING table

Column Name	Column Description	Data type
EVENT_ID*	Identifier assigned to this event. References AUDIT_EVENT (ID).	Numeric
OWNER_NAME	Name of the account owner.	Character (1000)
OWNER_DN	Distinguished name of the owner.	Character (1000)
SERVICE_NAME*	Name of the service to which the account belongs.	Character (1000)
SERVICE_DN*	Distinguished name of the service.	Character (1000)
ACCESS_NAME[1]	Name of the access type that the account acquired.	Character (1000)
ACCESS_DN[1]	Distinguished name of the access type.	Character (1000)

* Indicates the column is required and not null.

[1] Indicates the column was added in release 5.0.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the account management events in the AUDIT_EVENT table.

Table 32. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Account Management.

Table 32. Values for columns in the AUDIT_EVENT table (continued)

Column Name	Value
entity_name	Name of the account.
entity_dn	Distinguished name of the account.
entity_type	Types of the account (service). For example, Active Directory, Oracle, LDAP, Windows 2000, or IBM Security Privileged Identity Manager.
action	Types of actions: Add – Provision a new account on the target resource Modify – Modify an existing account Delete – Delete existing account Suspend – Suspend existing account Restore – Restore existing account ChangePassword – Change password for an account PasswordPickup – Pick a password for an account identified by the provisionTarget Adopt – Adopt an orphan account Orphan – Orphan an account

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each account management event in the AUDIT_EVENT table.

- **Add Account event**

  entity_name, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn

- **Modify Account event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn

- **Delete Account event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn

- **Suspend Account event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn

- **Restore Account event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn
- **Change Password event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn
- **Synchronize Password event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

  From audit_mgmt_provisioning: owner_name, owner_dn, service_name, service_dn
- **Adopt Account event**

  entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, timestamp, result_summary

  From audit_mgmt_provisioning: owner_dn, service_dn
- **Orphan Account event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, timestamp, result_summary

  From audit_mgmt_provisioning: owner_dn, service_dn

Table columns for account management in the AUDIT_MGMT_ACCESS_REQUEST table

The following list shows the columns for all account management event in the AUDIT_MGMT_ACCESS_REQUEST table.

- Event_ID
- Wokflow_Process_Id
- Action
- Access_Obligations_Ids
- Status
- Completed_Date

Table columns for account management in the AUDIT_MGMT_OBLIGATION table

The following list shows the columns for all account management event in the AUDIT_MGMT_OBLIGATION table.

- Event_ID
- Id
- Person_Dn
- Obligation_Type
- System_Generated

Database views

The tables described in this section are used for database views.

V_PIM_CICO_HISTORY view

This view shows the history of credentials that are checked in and checked out for all resource types.

Table 33. V_PIM_CICO_HISTORY view

Column	Description	""
ACCOUNT_UID	Credential user name.	No
CRED_DN	Credential distinguished name in data store.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
CRED_BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
RESOURCE_PROPERTY_n	Refer to the “Resource property mapping.”	Yes
USER_NAME	User name of the person who used the credential.	No
CHECKOUT_TIMESTAMP	Time when the person started using the credential.	No
CHECKIN_TIMESTAMP	Time when the credential is returned. If the credential is still in use, this will be null.	Yes
CHECKIN_USER_NAME	The person who checked in the credential, which can be the person who used it or an administrator.	Yes
ACTION	How the credential is checked in: "CheckIn" indicates that a user checked in the credential, "NotifyCheckinExpiredLease" indicates that the credential was not checked in until its lease expires, and an automatic check-in was performed.	Yes
JUSTIFICATION	The justification that is provided for using the credential.	Yes
LEASE_DN	Distinguished name of the lease.	No

Resource property mapping

The resource property mapping table shows the resource properties to be used for the V_PIM_CICO_HISTORY view.

Table 34. Resource property mapping

	Resource Types			
Column	**WinAD**	**WinLocal**	**Database**	**""**
RESOURCE_PROPERTY_1	Domain NetBIOS name	Host name	IP address	N/A
RESOURCE_PROPERTY_2	Domain DNS name	N/A	Port	N/A
RESOURCE_PROPERTY_3	N/A	N/A	Host name	N/A
RESOURCE_PROPERTY_4	N/A	N/A	Database type	N/A

Table 34. Resource property mapping (continued)

Column	Resource Types			
	WinAD	**WinLocal**	**Database**	**""**
RESOURCE_PROPERTY_5	N/A	N/A	N/A	N/A

V_PIM_CICO_HISTORY_DB_RSRC view

This view shows the history of credentials that are checked in and checked out for the **database** resource type.

Table 35. V_PIM_CICO_HISTORY_DB_RSRC view

Column	Description	""
ACCOUNT_UID	Credential user name.	No
CRED_DN	Credential distinguished name in data store.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
CRED_BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
DB_IP_ADDRESS	Database IP address.	No
DB_PORT	Database port.	No
DB_TYPE	Database type.	No
DB_HOST_NAME	Database host name.	Yes
USER_NAME	User name of the person who used the credential.	No
CHECKOUT_TIMESTAMP	Time when the person started using the credential.	No
CHECKIN_TIMESTAMP	Time when the credential is returned. If the credential is still in use, this will be null.	Yes
CHECKIN_USER_NAME	The person who checked in the credential, which can be the person who used it or an administrator.	Yes
ACTION	How the credential is checked in: "CheckIn" indicates that a user checked in the credential, "NotifyCheckinExpiredLease" indicates that the credential was not checked in until its lease expires, and an automatic check-in was performed.	Yes
JUSTIFICATION	The justification that is provided for using the credential.	Yes
LEASE_DN	Distinguished name of the lease.	No

V_PIM_CRED_INFO view

This view shows the information of credentials that are checked in and checked out for all resource types.

Table 36. V_PIM_CRED_INFO view

Column	Description	""
L_DN	Credential distinguished name in data store.	No

Table 36. V_PIM_CRED_INFO view (continued)

Column	Description	""
ACCOUNT_UID	Credential user name.	No
USE_GLOBAL_SETTINGS	Whether the credential uses global settings or its own custom settings.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
RESET_PASSWORD	Whether or not the credential password will be reset when it is checked in (for exclusive credentials only).	Yes
PWD_ROTATION_INTERVAL	Number of days between automatic password resets.	Yes
PSWD_LAST_CHANGED	Time when the credential password was last reset.	Yes
L_RESOURCE_DN	Credential resource distinguished name.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
L_BU_DN	Distinguished name of the organization container where the credential is in.	No
BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No

V_PIM_CRED_INFO_DB_RSRC view

This view shows the information of credentials that are checked in and checked out for the database resource type.

Table 37. V_PIM_CRED_INFO_DB_RSRC view

Column	Description	""
L_DN	Credential distinguished name in data store.	No
ACCOUNT_UID	Credential user name.	No
USE_GLOBAL_SETTINGS	Whether the credential uses global settings or its own custom settings.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
RESET_PASSWORD	Whether or not the credential password will be reset when it is checked in (for exclusive credentials only).	Yes
PWD_ROTATION_INTERVAL	Number of days between automatic password resets.	Yes
PSWD_LAST_CHANGED	Time when the credential password was last reset.	Yes
L_RESOURCE_DN	Credential resource distinguished name.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
DB_IP_ADDRESS	Database IP address.	No
DB_PORT	Database port.	No
DB_TYPE	Database type.	No
DB_HOST_NAME	Database host name.	Yes

Table 37. V_PIM_CRED_INFO_DB_RSRC view (continued)

Column	Description	""
L_BU_DN	Distinguished name of the organization container where the credential is in.	No
BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No

V_PIM_CRED_DETAILS view

This view shows the details of credentials that are checked in and checked out for all resource types.

Note: This view does not follow the First Normal Form.

Table 38. V_PIM_CRED_DETAILS view

Column	Description	""
L_DN	Credential distinguished name in data store.	No
ACCOUNT_UID	Credential user name.	No
USE_GLOBAL_SETTINGS	Whether the credential uses global settings or its own custom settings.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
RESET_PASSWORD	Whether or not the credential password will be reset when it is checked in (for exclusive credentials only).	Yes
PWD_ROTATION_INTERVAL	Number of days between automatic password resets.	Yes
PSWD_LAST_CHANGED	Time when the credential password was last reset.	Yes
CREDENTIAL_TAG	One of the credential's tags, if any.	Yes
L_RESOURCE_DN	Credential resource distinguished name.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
RESOURCE_ALIAS	One of the credential resource's aliases, if any.	Yes
RESOURCE_TAGS	One of the credential resource's tags, if any.	Yes
ERLESSEENAME	User name of the person that is currently using the credential.	Yes
ERLEASEEXPIRATIONTIME	Time when the current lease expires.	Yes
ERJUSTIFICATION	Justification for the current usage of the credential.	Yes
ERLEASESTATUS	Status of the current lease.	Yes
L_BU_DN	Distinguished name of the organization container where the credential is in.	No
BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No
LEASE_DN	Distinguished name of the current lease.	Yes
LESSEE_DN	Distinguished name of the user who is currently using the credential.	Yes

V_PIM_CRED_DETAILS_DB_RSRC view

This view shows the details of credentials that are checked in and checked out for the **database** resource type.

Note: This view does not follow the First Normal Form.

Table 39. V_PIM_CRED_DETAILS_DB_RSRC view

Column	Description	""
L_DN	Credential distinguished name in data store.	No
ACCOUNT_UID	Credential user name.	No
USE_GLOBAL_SETTINGS	Whether the credential uses global settings or its own custom settings.	No
IS_EXCLUSIVE	Whether or not an entitled user has to check out this credential to see its password.	No
RESET_PASSWORD	Whether or not the credential password will be reset when it is checked in (for exclusive credentials only).	Yes
PWD_ROTATION_INTERVAL	Number of days between automatic password resets.	Yes
PSWD_LAST_CHANGED	Time when the credential password was last reset.	Yes
CREDENTIAL_TAG	One of the credential's tags, if any.	Yes
L_RESOURCE_DN	Credential resource distinguished name.	No
RESOURCE_UID	Credential resource unique ID.	No
RESOURCE_NAME	Credential resource name.	No
RESOURCE_TYPE	Credential resource type: "winad", "winlocal", "database", or "" (generic).	No
DB_IP_ADDRESS	Database IP address.	No
DB_PORT	Database port.	No
DB_TYPE	Database type.	No
DB_HOST_NAME	Database host name.	Yes
RESOURCE_ALIAS	One of the credential resource's aliases, if any.	Yes
RESOURCE_TAGS	One of the credential resource's tags, if any.	Yes
ERLESSEENAME	User name of the person that is currently using the credential.	Yes
ERLEASEEXPIRATIONTIME	Time when the current lease expires.	Yes
ERJUSTIFICATION	Justification for the current usage of the credential.	Yes
ERLEASESTATUS	Status of the current lease.	Yes
L_BU_DN	Distinguished name of the organization container where the credential is in.	No
BU_NAME	Name of the organization container (e.g. admin domain) where the credential is in.	No
LEASE_DN	Distinguished name of the current lease.	Yes
LESSEE_DN	Distinguished name of the user who is currently using the credential.	Yes

Container management

This section describes the columns used by events related to events specific to container management, such as add, modify, and delete.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the AUDIT_EVENT table.

Table 40. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Container Management.
entity_name	Name of the container.
entity_dn	Distinguished name of the container.
entity_type	Types of entities: Organization Org_unit Business_Partner_Organization Location Admin_Domain
action	Types of actions: Add – Add a container Modify – Modify an existing container Delete – Delete a container

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add Container event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Container event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Delete Container event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

Organization role management

This section describes the columns used by events related to organization role management, such as add, modify, and delete.

In addition to the AUDIT_EVENT table, the AUDIT_MGMT_TARGET table is used by organization role management events.

AUDIT_MGMT_TARGET table

The AUDIT_MGMT_TARGET table is used if the action is Add Member or Remove Member.

Table 41. AUDIT_MGMT_TARGET table

Column Name	Column Description	Value Type	Required?
event_id	Identifier for the event. Foreign key to the ID column of the table audit_event.	long	Yes
target_entity_name	The name of the member that is being added to or removed from the role. Applicable if action= Add Member/ Remove Member.	string	Yes, when action= Add Member or Remove Member
target_entity_dn	The distinguished name of the member that is being added to or removed from the role. Applicable if action= Add Member/ Remove Member.	string	Yes, when action= Add Member or Remove Member
target_entity_type	The type of the member that is being added to or removed from the role. Applicable if action= Add Member/ Remove Member.	string	Yes, when action= Add Member or Remove Member

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the organization role management events in the AUDIT_EVENT table.

Table 42. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Organizational Role Management
entity_name	Name of the role.
entity_dn	Distinguished name of the role.
entity_type	Types of entities: static_org_role – Static organizational role that is involved in this event. dynamic_org_role – Dynamic organizational role that is involved in this event.
action	Types of actions: Add – Add a role. Modify – Modify an existing role. This action also involves modifying membership. Delete – Delete a role. AddMember – Add a member to the role. RemoveMember – Remove a member from the role.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each organization role management event in the AUDIT_EVENT table.

Add Static Role event

```
entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
container_name, container_dn, timestamp, result_summary
```

Modify Static Role event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Delete Static Role event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Add Member to Static Role event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, timestamp, result_summary

AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn, target_entity_type

Remove Member from Static Role event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, timestamp, result_summary

AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn, target_entity_type

Add Dynamic Role event

entity_name, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Modify Dynamic Role event

entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Delete Dynamic Role event

entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Group management

These AUDIT_EVENT columns are used by events that are related to group management, such as add, modify, and delete.

In addition to the AUDIT_EVENT table, the AUDIT_MGMT_TARGET table is used by group management events.

AUDIT_MGMT_TARGET table

The AUDIT_MGMT_TARGET table is used if the action is Add Member or Remove Member.

Table 43. AUDIT_MGMT_TARGET table

Column Name	Column Description	Value Type	Required?
event_id	Identifier associated with this event. Foreign key to the ID column of the table audit_event.	long	Yes

Table 43. AUDIT_MGMT_TARGET table (continued)

Column Name	Column Description	Value Type	Required?
target_entity_name	The name of the member that is being added to or removed from the ITIM group. Applicable if action= Add Member or Remove Member.	string	Yes, when action= Add Member or Remove Member
target_entity_dn	The distinguished name of the member that is being added to or removed from the ITIM group. Applicable if action= Add Member or Remove Member.	string	Yes, when action= Add Member or Remove Member
target_entity_type	The type of the member that is being added to or removed from the ITIM group. Applicable if action= Add Member or Remove Member.	string	Yes when action= Add Member or Remove Member

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the group management events in the AUDIT_EVENT table.

Table 44. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	ITIM Group Management
entity_name	Name of the ITIM group.
entity_dn	Distinguished name of the ITIM group.
entity_type	Types of entities: static_org_role – Static organizational role that is involved in this event. dynamic_org_role – Dynamic organizational role that is involved in this event.
action	Types of actions: Add – Add an ITIM group. Modify – Modify an ITIM group. This action also involves modifying membership. Delete – Delete an ITIM group. AddMember – Add a member to the ITIM group. RemoveMember – Remove a member from the ITIM group.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each group management event in the AUDIT_EVENT table.

Add ITIM Group event

```
entity_name, entity_type, initiator_name, initiator_dn,
initiator_type, initiator_person_dn, initiator_person_name,
container_name, container_dn, timestamp, result_summary
```

Modify ITIM Group event

```
entity_name, entity_dn, entity_type, initiator_name, initiator_dn,
```

initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Delete ITIM Group event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, container_name, container_dn, timestamp, result_summary

Add Member to ITIM Group event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, timestamp, result_summary

AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn, target_entity_type

Remove Member from ITIM Group event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, workflow_process_id, timestamp, result_summary

AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn, target_entity_type

Service management

This section describes the columns used by event-specific to service, such as add, modify, and delete.

In addition to the AUDIT_EVENT table, the AUDIT_MGNT_TARGET table is used by account management events.

AUDIT_MGNT_TARGET table

Table 45. AUDIT_MGNT_TARGET table

Column Name	Column Description	Value Type	Required?
event_id	Identifier associated with this event. Foreign key to the ID column of the table audit_event.	long	Yes
target_entity_name	Name of the target (service, service profile, or all services) for the adoption rule. Applicable if action= Add, Modify, or Delete an adoption rule.	string	Yes for action= Add, Modify, or Delete an adoption rule
target_entity_dn	The distinguished name of the target (service, service profile, or all services) for adoption rule. Applicable if action= Add, Modify, or Delete an adoption rule.	string	Yes for action= Add, Modify, or Delete an adoption rule
target_entity_type	The type of the target (service, service profile, or all services) for adoption rule. Applicable if action= Add, Modify, or Delete an adoption rule.	string	Yes for action= Add, Modify, or Delete an adoption rule

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the AUDIT_EVENT table.

Table 46. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Service Management.
entity_name	Name of the service.
entity_dn	Distinguished name of the service.
entity_type	Types of resource the service represents. For example: Active Directory, Oracle, LDAP, Windows 2000, or IBM Security Privileged Identity Manager.
action	Types of actions: Add – Add a service. Modify – Modify a service. This action includes the change compliance alert operation. Delete – Delete a service. Add_adoption_rule – Add an adoption rule for this service group. Update_adoption_rule – Update adoption rule for this service/service type. Delete_adoption_rule – Delete adoption rule for this service/service type.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add Service event**

  entity_name, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Modify Service event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Delete Service event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Add Adoption rule Service event**

  entity_name, entity_dn, initiator_name, initiator_dn, timestamp, result_summary

  AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn
- **Modify Adoption rule Service event**

  entity_name, entity_dn, initiator_name, initiator_dn, timestamp, result_summary

  AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn
- **Delete Adoption rule Service event**

  entity_name, entity_dn, initiator_name, initiator_dn, timestamp, result_summary

  AUDIT_MGMT_TARGET table: target_entity_name, target_entity_dn

Reconciliation

This section describes the columns used by events specific to reconciliation, such as runRecon, setServiceParams, and setReconUnit.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the reconciliation events in the AUDIT_EVENT table.

Table 47. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Reconciliation.
entity_name	Name of the service.
entity_dn	Distinguished name of the service.
entity_type	Type of the resource the service represents. For example: Active Directory, Oracle, LDAP, Windows 2000, or IBM Security Privileged Identity Manager.
action	Types of actions: Runrecon – Start the reconciliation. SetServiceReconParameters – Set the service reconciliation parameters. SetReconUnit – Set the service reconciliation unit.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each reconciliation event in the AUDIT_EVENT table.

Run Reconciliation event

entity_name, entity_dn, entity_type, workflow_process_id, initiator_name, initiator_dn, initiator_type, initiator_person_dn, initiator_person_name, action, timestamp, result_summary

Set Recon Unit event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, timestamp, result_summary

Set Service Recon Parameters event

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, timestamp, result_summary

Entitlement workflow management

This section describes the columns used by events specific to custom workflow management, such as add, modify, and delete.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the AUDIT_EVENT table.

Table 48. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Entitlement Workflow management.
entity_name	Name of the workflow.
entity_dn	Distinguished name of the workflow.
entity_type	Types of entities: global – Applied to any policy regardless of the service type *service_type* – Type of service to which this workflow is applicable

Table 48. Values for columns in the AUDIT_EVENT table (continued)

Column Name	Value
action	Types of actions: Add – Add a workflow. Modify – Update a workflow. Delete – Delete a workflow.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add Entitlement workflow event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Delete Entitlement workflow event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Modify Entitlement workflow event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

Entity operation management

This section describes the columns used by events specific to system workflow management, such as add, modify, and delete.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the AUDIT_EVENT table.

Table 49. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Entity Operation Management.
entity_name	Name of the operation that is being managed.
entity_dn	Distinguished name of the workflow.
entity_type	Type of the entity whose operation is being managed. For example, Person, Account, Bpperson, ITIMAccount, SQLAccount, and others.
action	Types of actions: Add – Add an operation. Modify – Update an operation. Delete – Delete an operation.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add Entity Operation event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Delete Entity Operation event**

  `entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary`
- **Modify Entity Operation event**

  `entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary`

System configuration

This section describes the columns used by events specific to IBM Security Privileged Identity Manager configuration performed through the Configuration tab.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the `AUDIT_EVENT` table.

Table 50. Values for columns in the `AUDIT_EVENT` table

Column Name	Value
`itim_event_category`	IBM Security Privileged Identity Manager System Configuration.
`entity_name`	Name of the entity. The value is specific to the type of entity type that is being updated.
`entity_dn`	Distinguished name of the entity or entity type if the entity that is being updated is an attribute.
`entity_type`	Types of entity: `FormTemplate` – `Formtemplate` for IBM Security Privileged Identity Manager object profiles `LogonProperties` – IBM Security Privileged Identity Manager logon properties `PostOfficeConfigurationProperties` – Post Office configuration properties `WorkflowNotificationProperties` – Workflow notification properties `ChallengeResponseProperties` – IBM Security Privileged Identity Manager challenge and response properties `Serviceprofile` – Service profile *<ITIM System Entity >* – System defined entities. For example, `Person`, `Account`, `Organization`, `BPOrganization`, `ITIMAccount`, `SQLAccount`, and others.
`action`	Types of actions: `Add` – Add a property or system entity from the Configuration tab. `Modify` – Update a property or system entity from the Configuration tab. `Delete` – Delete a property or system entity from the Configuration tab.

Value of the entity_name column:

This section describes the value for the `entity_name` column for each `entity_type` value defined for system configuration events.

Table 51. Value of the entity_name column table

entity_type	Value	Example
FormTemplate	Name of the profile whose form is being modified.	Admin Domain, Person, AIX Account, DSML2Service, SQLService, Organization
JoinDirective	Name of the attribute whose join directive is being updated.	Errole, eruid, erhomepage
Compliance Alert Rule	Name of the attribute whose Compliance alert rule is being updated.	Errole, eruid, erhomepage
LogonProperties	Property name.	erLostPswdByMail, erResponseEmail, erNumLogonAttempt
Policy Enforcement Properties	Property name.	
Post Office Configuration Properties	Property name.	
Workflow Notification Properties	Property name.	
Challenge Response Properties	Property name.	erChallengeDefMode, erChallengeMode, erResponseEnable
<ITIM System Entity>	Attribute of the entity that is being updated.	erAttrMap, erSearchAttr, erCustomClass, erRdnAttr, erLifeCycleRule.
Serviceprofile	Name of the service profile that is being installed or uninstalled.	Win2kService, BroadVisionService, SolarisService

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Add System Entity event**

  entity_name, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Delete System Entity event**

  entity_name, entity_dn, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Modify System Entity event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Add Life Cycle Rule event**

  entity_name, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Delete Life Cycle Rule event**

  entity_name, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Modify Life Cycle Rule event**

  entity_name, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Challenge Config event**

  initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Challenges event**

initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Form Template event**

  entity_name, entity_dn, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Password Properties event**

  initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Post Office Properties event**

  initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Privilege Rule event**

  entity_name, entity_dn, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Workflow Notification Properties event**

  initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

- **Set Workflow Notification Template event**

  entity_name, entity_dn, initiator_name, initiator_dn, action, container_name, container_dn, timestamp, result_summary

Runtime events

This section describes the columns used by event related to IBM Security Privileged Identity Manager start and stop events.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the container management operations in the AUDIT_EVENT table.

Table 52. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	IBM Security Privileged Identity Manager runtime events.
action	Types of actions: Start_itim – Start command for IBM Security Privileged Identity Manager. MStop_itim – Stop command for IBM Security Privileged Identity Manager.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each person management action in the AUDIT_EVENT table.

- **Start ITIM Server event**

  action, timestamp, result_summary

- **Stop ITIM Server event**

  action, timestamp, result_summary

Self-password change

This section describes the columns that are used by events that are related to password change.

If a self-password change request affects at least one ITIM account and at least one non-ITIM account, two separate events are audited for the request. One self-password change event is audited for the ITIM accounts. Another self-password change event is audited for the non-ITIM accounts.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the self-password change events in the AUDIT_EVENT table.

Table 53. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Self-password change.
action	Types of actions: ChangePassword – Changing a self-password. ResetPassword – Resetting a self-password.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each self-password change event in the AUDIT_EVENT table.

Change self-password event

```
entity_name, entity_dn, action, workflow_process_id, initiator_name,
initiator_dn, initiator_type, initiator_person_dn,
initiator_person_name, timestamp, result_summary
```

Reset self-password event

```
entity_name, entity_dn, action, workflow_process_id, initiator_name,
initiator_dn, initiator_type, initiator_person_dn,
initiator_person_name, timestamp, result_summary
```

Credential management

This section describes the columns used by events related to Credential management. For example, add to vault, modify, delete, register password, view password history, or get password for non-exclusive credential.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Credential management operations in the AUDIT_EVENT table.

Table 54. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	CredentialManagement
entity_name	Credential name.
entity_dn	Distinguished name of the credential.
entity_type	Credential
workflow_process_id	Process ID of the initiated workflow. Only applicable to Add action.
result_summary	Result of operation: Submitted – submitted to workflow successfully Success – completed successfully

Table 54. Values for columns in the AUDIT_EVENT table (continued)

Column Name	Value
action	Types of actions: Add – add a credential to vault Modify – modify a credential Delete – delete a credential from vault RegisterPassword – register credential password in the vault PasswordHistory – view credential password history in the vault GetPassword – get password of non-exclusive credential from vault Connect – connect a credential to an account

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each Credential management action in the AUDIT_EVENT table.

- **Add to Vault event**

  entity_name, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary, comments
- **Delete Credential event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Modify Credential event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Register Password event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **View Password History event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Get Password event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Connect credential event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary, comments

Credential Pool management

This section describes the columns used by events related to Credential Pool management, such as add, modify, or delete.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Credential Pool management operations in the AUDIT_EVENT table.

Table 55. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	CredentialPoolManagement
entity_name	Credential pool name.
entity_dn	Distinguished name of the credential pool.
entity_type	CredentialPool
result_summary	Result of operation: Success – completed successfully
action	Types of actions: Add – add a credential pool Modify – modify a credential pool Delete – delete a credential pool

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each Credential Pool management action in the AUDIT_EVENT table.

- **Add Credential Pool event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Delete Credential Pool event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary
- **Modify Credential Pool event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

Credential Lease management

This section describes the columns used by events related to Credential Lease management. For example, check out, check in, get password, notify expired lease, or notify and check in expired lease.

Credential Lease management events uses the following tables:

- AUDIT_EVENT
- AUDIT_MGMT_LEASE
- AUDIT_MGMT_TARGET

AUDIT_MGMT_LEASE table

The AUDIT_MGMT_LEASE table is used for specific events.

The events are:

- Checkout event
- Checkin event
- Expired lease notify and checkin event
- All other events if the credential is a pool member

Table 56. AUDIT_MGMT_LEASE table

Column Name	Column Description	Data type
event_id*	Identification assigned to the event. References AUDIT_EVENT (ID).	Numeric
lease_expiration_date	The lease expiration time. Only applicable to the Checkout action.	Character (500)
justification	The business justification for checkout. Only applicable to the Checkout action.	Character (2000)
pool_name	The credential pool name. Applicable to all actions if the credential is a pool member.	Character (256)
pool_dn	The credential pool DN. Applicable to all actions if the credential is a pool member.	Character (2000)
custom_attribute_1 to custom_attribute_5	The lease custom attribute values. Only applicable to the Checkout action.	Character (2000)
lease_dn	The lease DN.	Character (2000)

* Indicates the column is required and not null.

AUDIT_MGMT_TARGET table

The AUDIT_MGMT_TARGET table is used for specific events.

The events are:

- Checkout event
- Checkin event
- NotifyCheckinExpiredLease event

Table 57. AUDIT_MGMT_TARGET table

Column Name	Column Description	Value type	Required
event_id	The unique identifier associated with this event. Foreign key to the ID column of the audit_event table.	long	Yes
target_entity_name	The name of the resource that is associated with the credential being checked out or checked in.	string	Yes, when action=Checkout, Checkin, NotifyCheckinExpiredLease
target_entity_dn	The unique identifier of the resource that is associated with the credential being checked out or checked in.	string	Yes, when action=Checkout, Checkin, NotifyCheckinExpiredLease
target_entity_type	The resource profile name. The default value is DefaultCredentialServic	string	Yes, when action=Checkout, Checkin, NotifyCheckinExpiredLease

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Credential Lease management operations in the AUDIT_EVENT table.

Table 58. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	CredentialLeaseManagement
entity_name	Credential name.
entity_dn	Distinguished name of the credential.
entity_type	Credential
workflow_process_id	Process ID of the initiated workflow. Applicable to Checkin, Checkout, NotifyExpiredLease, and NotifyCheckinExpiredLease actions.
result_summary	Result of operation: Submitted – submitted to workflow successfully. Success – completed successfully. Only applicable to GetPassword action. Failure – failed. Only applicable to the second Checkin event, which tries to check in a credential already checked in by someone else.
action	Types of actions: Checkout – check out a credential. Checkin – check in a credential. GetPassword – get password of a checked out credential. NotifyExpiredLease – Notify an expired lease. NotifyCheckinExpiredLease – Notify and check in an expired lease.

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each Credential Lease management action in the AUDIT_EVENT table.

- **Checkout event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary

  AUDIT_MGMT_LEASE table: lease_expiration_time, justification, pool_name, pool_dn, custom_attribute_1, custom_attribute_2, custom_attribute_3, custom_attribute_4, custom_attribute_5

- **Checkin event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary, comments

  Note: If a user or an IBM Security Access Manager ESSO session tries to check in a credential already checked in by someone else, then the second checkin attempt is audited as a Checkin event. The result_summary is FAILURE and the comment is Invalid lease during checkin.

  AUDIT_MGMT_LEASE table: pool_name, pool_dn, lease_dn

- **Get Password event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

  AUDIT_MGMT_LEASE table: pool_name, pool_dn

- **Notify Expired Lease event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary

  AUDIT_MGMT_LEASE table: pool_name, pool_dn

- **Notify and Checkin Expired Lease event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, workflow_process_id, container_name, container_dn, timestamp, result_summary

  AUDIT_MGMT_LEASE table: pool_name, pool_dn, lease_dn

Shared Access Policy management

This section describes the columns used by events related to Shared Access Policy management, such as add, modify, or delete.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Shared Access Policy management operations in the AUDIT_EVENT table.

Table 59. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	SharedAccessPolicyManagement
entity_name	Name of the shared access policy.
entity_dn	Distinguished name of the shared access policy.
entity_type	SharedAccessPolicy
result_summary	Result of operation: Success – completed successfully
action	Types of actions: Add – add a policy Modify – modify a policy Delete – delete a policy

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each Shared Access Policy management action in the AUDIT_EVENT table.

- **Add Shared Access Policy event**

entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

- **Delete Shared Access Policy event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

- **Modify Shared Access Policy event**

  entity_name, entity_dn, entity_type, initiator_name, initiator_dn, container_name, container_dn, timestamp, result_summary

Application ID management

The audit log monitors all activities of an application instance's registration and credential retrieval for OAuth 2.0 Token authentication.

Values for columns in the AUDIT_EVENT table

The following table describes the column values for the Application ID management operations in the AUDIT_EVENT table.

Table 60. Values for columns in the AUDIT_EVENT table

Column Name	Value
itim_event_category	Authentication
entity_name	Application instance name. For example: hrapp@server1
entity_type	OAuth
result_summary	Result of operation: SUCCESS FAILURE
action	Types of actions: OAuthPasswordGrant OAuthRefresh OAuthLogin
initiator_type	APP_INSTANCE
comments	Specifies additional information for the following actions in JSON format: OAuthRefresh OAuthLogin

Table columns used in the AUDIT_EVENT table

The following list shows the columns for each application identity management action in the AUDIT_EVENT table.

Successful registration of an application instance

Table 61. Example audit log entry when an application instance is registered successfully

ITIM_EVENT_CATEGORY	ENTITY_NAME	ENTITY_TYPE	ACTION	INITIATOR_NAME	INITIATOR_TYPE	RESULT_SUMMARY
Authentication	hrapp_1	OAuth	OAuthPasswordGrant	valerie	PERSON	SUCCESS

Successful credential retrieval by an application instance

Table 62. Example audit log entry when an application instance retrieves a credential successfully

ITIM_EVENT_CATEGORY	ENTITY_NAME	ENTITY_TYPE	ACTION	INITIATOR_NAME	INITIATOR_TYPE	RESULT_SUMMARY	COMMENTS
Authentication	hrapp_1	OAuth	OAuthRefresh	hrapp_1	APP_INSTANCE	SUCCESS	{ "refreshToken": {"version":"1.0", "hashAlgo":"SHA-256", "tokenHash": "ehmL10Y+\/ 1LRIlld81fRq51fG2 Dll9r9yKlw6hGeYiM="} }
Authentication	hrapp_1	OAuth	OAuthLogin	hrapp_1	APP_INSTANCE	SUCCESS	{"accessToken": {"version":"1.0", "hashAlgo": "SHA-256", "tokenHash": "wRpltFHsBafWDAhXey4 roZ4uCqQYbD3VJgDnjzPX9Ng="} }

In the example, the application instance *hrapp_1* authenticated successfully, obtained a new OAuth token, and was able to retrieve any credential based on its credential entitlement.

- The first event, `OAuthRefresh`, indicates the success of the OAuth Token renewal. The `COMMENTS` column records the hash token of the OAuth Refresh Token which was used to renew.
- The second event, `OAuthLogin`, indicates the success of the OAuth Token login. The `COMMENTS` column records the hash token of the OAuth Access Token which was used to login.

Failed credential retrieval by an application instance

In the following example, the `RESULT_SUMMARY` of the OAuth Token renewal indicates FAILURE and in the `COMMENTS` column indicates `INVALID_TOKEN`.

This means that the application instance *hrapp_1* used an invalid OAuth token.

Table 63. Example audit entry when an application instance is unable to retrieve a credential due to an invalid token

ITIM_EVENT_CATEGORY	ENTITY_NAME	ENTITY_TYPE	ACTION	INITIATOR_NAME	INITIATOR_TYPE	RESULT_SUMMARY	COMMENTS
Authentication	hrapp_1	OAuth	OAuthRefresh	hrapp_1	APP_INSTANCE	FAILURE	{ "resultCode": "INVALID_TOKEN", "refreshToken": {"version": "1.0", "hashAlgo": "SHA-256", "tokenHash": "OQVuUd7+oNwS1VFB ezoobVLc4FDkxGvq796nRYDYWL0="} }

Table 64. Example audit entry when an application instance is unable to retrieve a credential because the application fingerprint does not match

ITIM_EVENT_CATEGORY	ENTITY_NAME	ENTITY_TYPE	ACTION	INITIATOR_NAME	INITIATOR_TYPE	RESULT_SUMMARY	COMMENTS
Authentication	hrapp_1	OAuth	OAuthRefresh	hrapp_1	APP_INSTANCE	FAILURE	{"resultCode": "INVALID_FINGERPRINT", "clientIp": "1.2.3.4", "nonMatchingFingerprint": {"userName":"user1", "hostName":"mycomputer", "group":"HRScript", "hostTimezone":"Asia/Singapore", "os":{"name": "Windows Server 2008 R2", "version": "6.1", "arch": "amd64"}} }

The `COMMENTS` column is useful for determining the cause of failure. In the event of an invalid fingerprint:

- `clientIp` always records the originating IP address of the machine where the application instance attempted to retrieve a credential.
- `nonMatchingFingerprint` lists all the mismatched fingerprint features together with the values which were used to retrieve a credential, except the originating IP address because it will always be logged as `clientIp`.

In the example, the following features did not match:

- OS user name
- host name
- group id

- host time zone
- OS

Interpreting the information in parts of the AUDIT_EVENT table is useful for troubleshooting, or for detecting any attempts of a malicious attack.

Investigating credential retrieval failures

The COMMENTS column for each audit log entry contains information that is useful both for troubleshooting and for identifying patterns of potential malicious attack attempts.

1. When there are one or more OAuthRefresh failure events for an application instance with an INVALID_TOKEN, you can start by tracing the recorded hash token information. The following possibilities might be occurring:
   - The OAuth Token of the application instance is stolen and misused by an attacker. If the hashed token in the last successful OAuthRefresh event is the same as the subsequent failed OAuthRefresh events, it indicates that the OAuth Token is stolen or used more than once. The same application instance cannot possibly have used the same OAuth Token twice because once an OAuth Token is authenticated successfully, the token is invalidated. The application instance be issued with a new token. A suspected third party could have stolen the OAuth Token and used it elsewhere where the fingerprint matches.

     Example of audit log event entries:

     Note: In an actual audit log, there might be other events in between.

ITIM_EVENT_CATEGORY	ENTITY_NAME	ENTITY_TYPE	ACTION	INITIATOR_NAME	INITIATOR_TYPE	RESULT_SUMMARY	COMMENTS
Authentication	hrapp_1	OAuth	OAuthRefresh	hrapp_1	APP_INSTANCE	SUCCESS	{ "refreshToken":{"version":"1.0", "hashAlgo":"SHA-256","tokenHash":"A7nRw31OE5 sXHSUknur60hal4L\/JT1HTSy\/1YGh+dec="} }
Authentication	hrapp_1	OAuth	OAuthRefresh	hrapp_1	APP_INSTANCE	FAILURE	{ "resultCode":"INVALID_TOKEN", "refreshToken":{"version":"1.0", "hashAlgo":"SHA-256","tokenHash":"A7nRw31OE 5sXHSUknur60hal4L\/JT1HTSy\/ 1YGh+dec="}}

   - The OAuth Token of the application instance in the application host machine is tampered. If the hashed token in the failed OAuthRefresh event is different from the preceding successful OAuthRefresh event, it indicates that someone might have modified the OAuth Token file.
2. When there is one or more OAuthRefresh failure events for an application instance with INVALID_FINGERPRINT, the client IP, and the non-matching fingerprint features and environment values aid troubleshooting.
   a. Unidentified IP address or suspicious environment variable values might mean that an attacker is attempting to forge the application host environment to retrieve credentials.
   b. If Group ID is used by an application instance and since the value is not automatically captured by the AppID, but must be typed by someone, if there is a mismatch, the chance of the mismatch being a malicious attack is high.
   c. One or more of the application host environments have changed. For example: IP address, OS upgrade. If the source of change is known, for example, if the change is simply part of a scheduled system maintenance, then the threat is lowered.

Chapter 11. Scenarios

Usage scenarios describe the goals for a persona, such as a privileged administrator or privileged user, and the workflow to help them achieve the desired results with the solution.

Creating an access request workflow

A privileged administrator can select the workflow to be used for each access.

About this task

The IBM Security Privileged Identity Manager administrator can create workflows that are started when a user requests access. Access request workflows are activities that are typically performed when users request access. Typically, this consists of approval and email notification.

After this workflow is created, a privileged user can request access to the managed resource.

For instance, if Judith User wants to access a managed resource, Judith can request access. The manager (Chuck Manager) is notified of the request. Chuck can approve or reject the request. If Chuck does not respond to the request within the specified period, the access owner must respond.

Procedure

1. Log in to the administrative console as PIM Manager.
2. From the navigation tree, select **Configure System** > **Manage Access Request Workflows**.
3. On the Manage Access Request Workflows page, in the **Access Request Workflows** table, click **Create**.
4. On the General notebook page, type the following information for your workflow:

   Name A name to identify the workflow. For example: `Windows DBA server access workflow`.

   Description
   A description of the intended purpose of the workflow. For example: `Approval for the DBA group`.

   Business unit
   Indicates the business unit. Click Search to locate and select a business unit. This field is read-only when you are making a change to a workflow.
5. Click the **Activities** tab. On the Activities page, select **Create an approval activity** and click **Go**.
6. On the Approval Activity page, specify the following information and click **OK**:

   Activity name
   A name to identify the activity. For example: `Manager Approval for Windows DBA server access`.

Approver type
Users who can approve the request. For example: **Manager**.

Escalation time in days
The request escalates to the specified escalation participant when this interval of time expires. For example: `10`

Escalation participant type
Users who can approve the request after it has been escalated. For example **Access owner**.

7. On the Activities notebook page, click **OK** again.
8. On the Success page, click **Close**.

Privileged user: Logging on to a managed resource

You can automatically log on to a managed resource using the shared access credential that you checked out.

The IBM Security Access Manager for Enterprise Single Sign-On AccessAgent client automates the check-out and check-in of shared access credentials. AccessAgent automatically checks in shared access credentials when you log out, exit, or close the managed resource.

If the shared access credential check-in process is not triggered automatically, the shared access credential remains checked out to the user until the lease time expires. You can check out a shared access credential only for a limited amount of time. The specific amount of time is the lease time.

You can use the following applications to log on to a managed resource.

Logging on with PuTTY or SecureCRT

As a privileged user, you can use PuTTY or SecureCRT to log on to a remote terminal host from Windows with shared privileged identities.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start PuTTY or SecureCRT.
2. Specify the target host name or IP address.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.

7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with the Microsoft Remote Desktop Connection (RDP) client

You can log on to a remote desktop with shared privileged identities with Remote Desktop Connection.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start the Microsoft Remote Desktop Connection client by clicking **Start** > **All Programs** > **Accessories** > **Remote Desktop Connection**.
2. Click **Connect**.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with IBM Personal Communications

Use the IBM Personal Communications application to log on to a mainframe application with shared access identity. You must configure the bundled Privileged Identity Management AccessProfile for your mainframe application before check-out and check-in automation can work.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

For check-out and check-in automation to work with your custom mainframe applications, the Privileged Administrator must apply specific changes to the bundled Privileged Identity Management AccessProfile. The Privileged Administrator must customize the Privileged Identity Management AccessProfile for IBM Personal Communications application before the privileged user can use it. Customization is necessary because:

- Each mainframe or terminal application might contain different output phrases.
- The AccessProfile or application signature must contain a similar phrase as the one displayed by the mainframe application. So, when the application displays the phrase, the logon automation by the AccessProfile can proceed.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start IBM Personal Communications.
2. Specify the target host name or IP address.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with SQL Server Management Studio

Use the SQL Server Management Studio application to administer Microsoft SQL Server databases with shared access credentials.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start SQL Server Management Studio.
2. Specify the target host name or IP address.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with DB2 Data Studio

Use the DB2® Data Studio application to administer DB2 databases with shared access credentials.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start IBM DB2 Data Studio.
2. Specify the target host name or IP address.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with the VMware vSphere Client

Use the VMware vSphere Client to log on to a virtual machine with shared access credentials.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfile for different logon prompts. See Modifying AccessProfiles. If the pre-configured Privileged Identity Management AccessProfile is updated, see Uploading AccessProfiles to the virtual appliance.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Start the **VMware vSphere Client**.
2. Specify the target host name or IP address.
3. When prompted to log on with shared access credentials, choose **Yes**.
4. When prompted with the reauthentication prompt, specify your password.
5. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
6. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
7. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

Logging on with SoftLayer

As a privileged user, you can use the web browser to log on to a SoftLayer® portal with a shared privileged identity.

Before you begin

Disable caching when you use Mozilla Firefox to access the SoftLayer console. Run in private browsing mode.

About this task

The Privileged Administrator can configure the Privileged Identity Management AccessProfiles for different logon prompts. See installing/cpt/c_accessprofiles_sso_pim.dita. If the pre-configured Privileged Identity Management AccessProfiles is updated, see installing/tsk/t_upload_accessprofies.dita.

The Wallet must contain IBM Security Privileged Identity Manager credentials.

Procedure

1. Go to the SoftLayer console: https://control.softlayer.com/
2. When prompted to log on with shared access credentials, choose **Yes**.
3. When prompted with the reauthentication prompt, specify your password. If the Wallet does not contain any IBM Security Privileged Identity Manager credentials, you are prompted to provide them.
4. When prompted with the Shared Access Selection window, select a credential pool to check out shared access credentials.

   Note: Enter a justification for the credential check-out.
5. When prompted to provide consent to be recorded, choose **Yes**. Session recording is started.

Results

The AccessProfile checks out the shared access credential from IBM Security Privileged Identity Manager and injects the shared access credential in the logon prompt.

More examples that can trigger check-out and check-in automation

Different events can determine the automation behavior. For example, when you start multiple sessions or when sessions are ended abnormally.

Table 65. More events that can trigger automated check-out or check-in behavior.

When	Automated check-out or check-in behavior
You • Start a second client application session. • Connect to the same resource as your client application session. • Choose the same credential pool.	The user is prompted whether to use an already checked out credential. The user can choose to reuse or check out a new credential. **Note:** If you choose a different credential pool, a separate check-out occurs.
• You use a client application. • A session is ended abnormally because of a system crash or deliberate termination.	AccessAgent checks in credentials that were used for the abnormally terminated application.
There is no connection to the IBM Security Privileged Identity Manager Server.	After the client application closes properly or ends, AccessAgent continuously attempts to check in all credentials that a user checked out. This process prevents any checked out credentials from being used outside the IBM Security Access Manager for Enterprise Single Sign-On domain.
You restart a client computer, and there are still credentials that are pending for check-in.	AccessAgent tries the check-in again when a corresponding user logs on to IBM Security Access Manager for Enterprise Single Sign-On. This approach avoids locking credentials so that they can be checked out by users.

Table 65. More events that can trigger automated check-out or check-in behavior. (continued)

When	Automated check-out or check-in behavior
You use the managed resource by using a checked out credential, from the client logon application, and after the lease expires on the checked out credential. For example: • You are finished with using the client logon application and the managed resource but forget to close the client logon application. • You are away from the computer for a long time.	AccessAgent checks in credentials when the IBM Security Privileged Identity Manager Administrator configured lease time expires. **Note:** One hour before the lease time expiration, a notification informs you when the lease time is almost expiring. You must stop the use of the credentials or have AccessAgent close the application when the lease expires. See the IBM Security Privileged Identity Manager Information Center for more information about lease expiry configurations.
You use the managed resource by using a checked out credential, from the client logon application, and after the lease expires on the checked out credential. For example: The computer goes into hibernate mode, and the credential is not checked in.	IBM Security Privileged Identity Manager handles lease expiry that is based on how lease expiry handling is configured. For example: • The credentials can be checked in or • Notification emails can be sent out. See the IBM Security Privileged Identity Manager Information Center for more information.

Integration with IBM Security Identity Manager

Privileged administrators, privileged users, and Privileged Identity Manager administrators can be on-boarded from IBM Security Identity Manager into IBM Security Privileged Identity Manager.

The following scenarios demonstrate how Annie Lewis (Privileged Administrator) accomplishes the following goals:

1. Add privileged accounts, for example `root` on a Linux host named *Pinnacle*, to the credential vault on IBM Security Privileged Identity Manager.
2. Define a policy to authorize who can use the privileged accounts that she creates on IBM Security Privileged Identity Manager.
3. Let users request access on the IBM Security Identity Manager console.
4. Approve such requests on IBM Security Identity Manager console.
5. Allow users to check out on IBM Security Privileged Identity Manager.

The following sections describe additional information for an integration scenario:

- Setting up a request and approval workflow for ISPIM roles on IBM Security Identity Manager
- Tasks that remain the same with or without integration with IBM Security Identity Manager
- Reports

Table 66. Types of users to be on-boarded

User	Description
Annie Lewis (Privileged Administrator)	• These employees own or are responsible for one or more privileged IDs on one or more systems. • Use IBM Security Privileged Identity Manager to manage and control the sharing of privileged credentials with co-workers.
James Smith (Privileged User)	These are employees who have a business need to access one or more privileged credentials that are managed by the IBM Security Privileged Identity Manager.
Jake Smith (Privileged Identity Manager Administrator)	This administrator can provision ISPIM accounts for Annie Lewis (Privileged Administrator) and James Smith (Privileged User) through IBM Security Identity Manager by adding Annie Lewis and James Smith into the correct ISPIM group (system role) when they take on the specified business roles.

Annie Lewis (Privileged Administrator) needs an ISPIM account and an ISPIM administrative domain. Annie Lewis (Privileged Administrator) requires the account and administrative domain so that she can add her privileged accounts, for example *root*, to the credential vault so that she can share them with other privileged users. Annie Lewis (Privileged Administrator) contacts Jake Smith (Privileged Identity Manager Administrator) to set up the account and administrative domain that she needs.

Table 67. Create an ISPIM administrative domain and ISPIM account

Persona	Jake Smith (Privileged Identity Manager Administrator)
Console	IBM Security Identity Manager Administrative console
Tasks	1. Create an ISPIM administrative domain for Annie Lewis (Privileged Administrator) where she can manage her shared credentials. For example: *Annie domain*. See Administrator domains and Creating groups. 2. Create an ISPIM account for Annie Lewis (Privileged Administrator) on the ISPIM service. For example: *Annie Lewis*. 3. Associate the Annie Lewis (Privileged Administrator) ISPIM account with the *Privileged Administrator Group* and *Annie domain*. 4. Create an account request workflow for the ISPIM service. See Adding an entitlement workflow.

After Annie Lewis (Privileged Administrator) gets a domain that she can use to manage her accounts, Annie logs on to the IBM Security Privileged Identity Manager Administrative console to add her credential to the credential vault, create an ISPIM role, and create a shared access policy before James Smith (Privileged User) can check out the credential to install the DB2 database on a server.

Table 68. Create a shared access policy

Persona	Annie Lewis (Privileged Administrator)

Table 68. Create a shared access policy (continued)

Console	IBM Security Privileged Identity Manager Administrative console IBM Security Privileged Identity Manager Service Center
Tasks	1. Add a shared access credential that requires check-out. See Credentials in the credential vault. a. Link the credential to the ISPIM administrative domain. In this setup: *Annie domain*. b. Specify the shared access credential in the **User ID** field. For example: *root*. c. Specify the managed resource. For example, set resource name as *Pinnacle*. d. Specify the check-out duration. For example: *1 week*. 2. Create an ISPIM role to represent the administrators who can check out the credential. For example: *Pinnacle admins*. See Creating roles. 3. Create a shared access policy to allow all members of the created ISPIM role to check out the created credential. For example: *Pinnacle admin access* policy. See Creating shared access policies. a. Add the *Pinnacle admins* role as members of the *Pinnacle admin access* policy. b. Set **Entitlement** as credential then specify the shared access credential. In this setup: *root*. on the Pinnacle system

Table 69. Assign Privileged Administrator as access owner

Persona	Jake Smith (Privileged Identity Manager Administrator)
Console	IBM Security Identity Manager Administrative console
Tasks	Reconcile the ISPIM service to sync Annie Lewis (Privileged Administrator) *Pinnacle admins* role in IBM Security Identity Manager. 1. Specify the group name. In this setup: Organization / *Annie domain* / *Pinnacle admins*. 2. Enable Annie's *Pinnacle admins* in IBM Security Identity Manager as an access and common access. Specify the shared access policy name *Pinnacle admin access*. 3. Assign Annie Lewis (Privileged Administrator) as the **Access Owner**.

James Smith (Privileged User) must check-out the administrative account, *root*, on *Pinnacle admin access* to install the IBM DB2 database for the reservation application.

Table 70. Check out the administrative account

Persona	James Smith (Privileged User)
Console	IBM Security Identity Manager, Version 7.0: IBM Security Identity Manager Service Center IBM Security Identity Manager, Version 5.1 and 6.0: IBM Security Identity Manager Self-service console

Table 70. Check out the administrative account (continued)

Tasks	Request access to *Pinnacle admin access*. An approval request is sent to Annie Lewis (Privileged Administrator). **Note:** After the request is approved, James Smith (Privileged User) can use IBM Security Access Manager for Enterprise Single Sign-On to check out the administrative account, *root*, on *Pinnacle* and install the DB2 database on the *Pinnacle server*.

Table 71. Grant access to the Pinnacle Server

Persona	Annie Lewis (Privileged Administrator)
Console	IBM Security Identity Manager, Version 7.0: IBM Security Identity Manager Service Center IBM Security Identity Manager, Version 5.1 and 6.0: IBM Security Identity Manager Self-service console
Tasks	Approve the request from James Smith (Privileged User) to access *Pinnacle admin access*.

Setting up a request and approval workflow for ISPIM roles on IBM Security Identity Manager

Annie Lewis (Privileged Administrator) uses IBM Security Privileged Identity Manager to on-board credentials (with IBM Security Privileged Identity Manager Service Center), create roles, and set up shared access policies (with IBM Security Privileged Identity Manager administrative console).

Annie Lewis (Privileged Administrator) needs to set up a role called `Linux Admins` and defines that this will use shared credentials under IBM Security Privileged Identity Manager.

To configure a request and approval workflow for ISPIM roles on IBM Security Identity Manager, the following steps occur:

1. On the IBM Security Identity Manager administrative console, Jake Smith (Privileged Identity Manager Administrator) sets up the ISPIM Service and reconciles the `Linux Admin` role into IBM Security Identity Manager.
2. On the IBM Security Identity Manager administrative console, Jake Smith (Privileged Identity Manager Administrator) enables the `Linux Admin` role as a comon access and makes Annie Lewis (Privileged Administrator), the access owner, as an approver of the request.

   See “Manage Access Approval Workflows” and “Manage Groups” in the IBM Security Identity Manager documentation.
3. On the IBM Security Identity Manager administrative console, James Smith (Privileged User) requests access to the `Linux Admin` role.
4. On the IBM Security Identity Manager administrative console, Annie Lewis (Privileged Administrator) approves the access request from James Smith (Privileged User).

   Note: Approval workflows for the `Linux Admin` role are not associated on IBM Security Privileged Identity Manager. If you do this, access approvals are required from both IBM Security Identity Manager and IBM Security Privileged Identity Manager.

Tasks that remain the same with or without integration with IBM Security Identity Manager

IBM Security Privileged Identity Manager Privileged Administrator or Privileged Administrator tasks:

- On-boarding of credentials into the credential vault
- Management of credential settings
- Management of automatic password resets on credentials
- Setting up shared access roles and policies

Privileged user tasks:

- Manual check-in and check-out with self service console
- Automatic check-in and check-out with session recording through AccessAgent.

Reports

IBM Security Privileged Identity Manager reports contains the shared access entitlements (role-based), and shared access history for privileged users. IBM Security Identity Manager reports contain a user's “individual” account entitlements (no shared access entitlements), including IBM Security Privileged Identity Manager account and role and group memberships.

Integration with SoftLayer

Privileged administrators can use IBM Security Privileged Identity Manager to manage privileged credentials that are used to log on to IBM SoftLayer.

SoftLayer AccessProfile

The SoftLayer AccessProfile provides single sign-on functionality to log on to SoftLayer.

The SoftLayer AccessProfile supports the following functions:

- Check out the credentials from IBM Security Privileged Identity Manager.
- Inject the credentials after the web browser is closed.
- Check in the credentials to IBM Security Privileged Identity Manager.

You must consider the following issues and limitations before using the SoftLayer AccessProfile:

- It can only be used in IBM Security Access Manager for Enterprise Single Sign-On supported web browsers. For example: Microsoft Internet Explorer 9, Microsoft Internet Explorer 10, or Mozilla Firefox 31.
- The profile is not checked in if a tab is closed. Close the web browser to check in the profile.
- The user is not prompted to check out the credential after restarting the web browser due to the cache that is maintained by the web browser. The user is logged on to SoftLayer automatically.

SoftLayer Adapter

The SoftLayer Adapter enables connectivity between the IBM Security Privileged Identity Manager and SoftLayer.

This adapter automates several administrative tasks on the SoftLayer server. You can use the adapter to automate the following tasks:

- Create, modify, suspend, restore, change password, and delete a user.
- Reconcile user and user attributes.

The SoftLayer Adapter is bundled with the IBM Security Privileged Identity Manager virtual appliance. As such, you only need to:

1. Create an identity provider for the SoftLayer profile (**SoftLayerProfile**) . See Adding identity providers.
   a. Specify a name that defines the adapter service on the server. For example, SoftLayer.

      Note: Do not use forward (/) or backward slashes (\) in the service name.
   b. Specify the URL which the adapter can use to communicate with SoftLayer. For the current SoftLayer release, use **https://api.softlayer.com**.
2. Create a credential and connect it to the SoftLayer identity provider that you created. See Adding credentials with Service Center and Connecting a credential to an identity provider.

   Note: In the **Resource** field, specify `control.softlayer.com`.
3. Define access to credentials and grant privileged users membership to access. See Creating access.
4. (Optional) If the credential check-in fails because the default SoftLayer password policy is not strong enough, modify the password strength rule in **Manage Password Policies**.

Integration with IBM QRadar Security Intelligence Platform

The collection of events from IBM Security Privileged Identity Manager for analysis in IBM QRadar® Security Intelligence Platform is now supported with IBM QRadar Security Intelligence Platform Device Support Modules (DSMs) for IBM Security Privileged Identity Manager.

Download the IBM QRadar Security Intelligence Platform DSMs for IBM Security Privileged Identity Manager.

The following `.rpm` packages must be downloaded:

- `7.2.0-QRADAR-PROTOCOL-JDBC-7.2-<version>.noarch.rpm`

Note: This is **not** applicable if the package is already installed.

DSMs:

- `7.2.0-QRADAR-DSM-IBMSecurityAccessManagerESSO-7.2-<version>.noarch.rpm`
- `7.2.0-QRADAR-DSM-IBMSecurityPrivilegedIdentityManager-7.2-<version>.noarch.rpm`
- `7.2.0-QRADAR-DSM-IBMPrivilegedSessionRecorder-7.2-<version>.noarch.rpm`

Content package:

- `7.2.0-QRADAR-ContentPackage-CustomProperties-IBMSecurityPrivilegedIdentityManager-7.2-<version>.x86_64.rpm`
- `7.2.0-QRADAR-ContentPackage-CustomProperties-IBMPrivilegedSessionRecorder-7.2-<version>.x86_64.rpm`

- 7.2.0-QRADAR-ContentPackage-CustomProperties-IBMSecurityAccessManagerESSO-7.2-<version>.x86_64.rpm

Configuring the log sources

Configure the log sources for Privileged Credential Management,Privileged Session Recorder, and IBM Security Access Manager for Enterprise Single Sign-On so that IBM QRadar Security Intelligence Platform can process the database.

Procedure

1. Perform the following tasks to configure the log sources for Privileged Credential Management:
   a. Review the The IBM Security QRadar DSM for IBM Security Privileged Identity Manager.
   b. Create a database view. See Configuring IBM Security Privileged Identity Manager.
   c. Configure a log source for IBM Security Privileged Identity Manager. See Adding a log source
2. Perform the following tasks to configure the log sources for Privileged Session Recorder:
   a. Review the IBM Security QRadar DSM for Privileged Session Recorder.
   b. Collect data from the IBM Security Privileged Identity Manager Session Recorder.
   c. Configure a log source for IBM Privileged Session Recorder.
3. Perform the following tasks to configure the log sources for IBM Security Access Manager for Enterprise Single Sign-On:
   a. Enable **syslog** by using the IBM Security Privileged Identity Manager CLI. See "Enabling syslog on the IBM Security Privileged Identity Manager virtual appliance"
   b. Configure a log source for IBM Security Access Manager for Enterprise Single Sign-On.

Enabling syslog on the IBM Security Privileged Identity Manager virtual appliance

Enable syslog for the IBM Security Access Manager for Enterprise Single Sign-On server on the IBM Security Privileged Identity Manager virtual appliance.

Procedure

1. Log on to the IBM Security Privileged Identity Manager virtual appliance command line interface.
2. In the command line interface, type update_syslog.
3. Enable the syslog attributes. For example,

```
rwrangler.example.com:service_properties> update_syslog

Enable syslog
  logSystemManagementActivity [true/false]: true
  logUserAdminActivity [true/false]: true
  logUserService [true/false]: true
  logUserActivity [true/false]: true
Syslog server port: 514
Syslog server hostname: 10.1.13.127
```

```
Syslog logging facility:
Syslog field-separator: ###
Syslog settings will be updated.
Restart ISPIM to apply the new settings.
```

where,

Enable syslog
Type `true` for each of the categories.

Syslog server port

Specify the server port number that is used for forwarding events to QRadar. Specify `514`.

Syslog server hostname

Specify the IP address or hostname of your QRadar Console or event collector.

Syslog logging facility

Specify the facility of the events forwarded to QRadar. Default value: 20.

Syslog field-separator

Specify the characters used to separate name-value pair entries in syslog payload.

4. Restart the virtual appliance.

Results

The log source is added to QRadar. Syslog events are automatically discovered. Events forwarded to QRadar are displayed on the **Log Activity** tab.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law :

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web

sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not

been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability
These terms and conditions are in addition to any terms of use for the IBM website.

Personal use
You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use
You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at http://www.ibm.com/legal/copytrade.shtml.

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Privacy Policy Considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings

can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering uses other technologies that collect each user's user name, password or other personally identifiable information for purposes of session management, authentication, single sign-on configuration, usage tracking, or functional purposes. These technologies can be disabled, but disabling them will also eliminate the functionality they enable.

This Software Offering does not use cookies to collect personally identifiable information. The only information that is transmitted between the server and the browser through a cookie is the session ID, which has a limited lifetime. A session ID associates the session request with information stored on the server.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details/us/en sections entitled "Cookies, Web Beacons and Other Technologies" and "Software Products and Software-as-a Service".

Index

A

access
  workflow, designing 187
access request management 145
  AUDIT_EVENT values 148
  AUDIT_MGMT_ACCESS_REQUEST values 145
  AUDIT_MGMT_MESSAGE values 148
  AUDIT_MGMT_OBLIGATION _ATTRIB values 147
  AUDIT_MGMT_OBLIGATION _RESOURCE values 148
  AUDIT_MGMT_OBLIGATION values 146
AccessAgent 11
AccessProfiles 188
  IBM Personal Communications 190
  SQL Server 190
  VMware vSphere Client 192
account management 158
ACI
  management 143
  management events 143
activity object, JavaScript extension 17
Activity.auditEvent object, JavaScript extension 18
Activity.description object, JavaScript extension 19
Activity.duedate object, JavaScript extension 19
Activity.getSubProcesses(), JavaScript extension 19
Activity.guid object, JavaScript extension 20
Activity.id object, JavaScript extension 20
Activity.index object, JavaScript extension 20
Activity.name object, JavaScript extension 21
Activity.participant object, JavaScript extension 21
Activity.resultDetail object, JavaScript extension 21
Activity.resultSummary object, JavaScript extension 21
Activity.setResult object, JavaScript extension 22
Activity.started object, JavaScript extension 22
Activity.state object, JavaScript extension 22
Activity.subtype object, JavaScript extension 23
Activity.type object, JavaScript extension 23
API 11
APIs
  web services API 5
Application ID management
  column values 183
Application identity management 183
Application Programming Interface (API)
  See API
AttributeChangeOperation object, JavaScript extension 24
AttributeChangeOperation.attr object, JavaScript extension 24
AttributeChangeOperation.op object, JavaScript extension 24
AttributeChangeOperation.values, JavaScript extension 25
AUDIT_EVENT 135
  access request management 148
  create manual activity event 154
  escalate manual activity event 157
  lifecycle rule 157
AUDIT_EVENT table 144, 158, 166
  column values 136, 137, 140, 141, 144, 158, 166, 167, 169, 171, 172, 173, 174, 176, 177
  table columns 137, 138, 140, 141, 144, 149, 157, 158, 159, 166, 167, 169, 171, 172, 173, 175, 176, 177, 182, 183
AUDIT_MGMT_ACCESS_REQUEST
  access request management 145
AUDIT_MGMT_ACTIVITY
  create manual activity event 150
  escalate manual activity event 155
AUDIT_MGMT_DELEGATE 139
AUDIT_MGMT_MESSAGE
  access request management 148
AUDIT_MGMT_OBLIGATION
  access request management 146
AUDIT_MGMT_OBLIGATION _ATTRIB
  access request management 147
AUDIT_MGMT_OBLIGATION _RESOURCE
  access request management 148
AUDIT_MGMT_PARTICIPANT
  create manual activity 153
  escalate manual activity 155
AUDIT_MGMT_PROVISIONING table 158
AUDIT_MGMT_TARGET 137
AUDIT_MGMT_TARGET table 144, 167, 168
AUDIT_MGNT_TARGET table 170
auditing schema tables 135
authentication 136

C

check-in
  examples 193
check-out
  examples 193
CheckIn 13
CheckOut 11
client application 193
CLT command 95
command line interface
  clean 109
  global commands 107
  virtual appliance 105
  virtual appliance services 112
commands, IBM Security Privileged Identity Manager 105
configuration response samples 133
container management 166
ContainerSearch object, JavaScript extension 25
ContainerSearch.searchByFilter object, JavaScript extension 25
ContainerSearch.searchByURI object, JavaScript extension 26
content tags
  dynamic tags 115
  examples 115
Context object, JavaScript extension 26
Context.getAccountParameter object, JavaScript extension 28
Context.getActivityResult object, JavaScript extension 28
Context.getActivityResultById object, JavaScript extension 28
Context.getLoopCount object, JavaScript extension 29
Context.getLoopCountByID object, JavaScript extension 29
Context.getProcessType object, JavaScript extension 30
Context.getRequestee object, JavaScript extension 30
Context.getService object, JavaScript extension 30
core dump
  command line interface 109
create manual activity 149
  AUDIT_MGMT_PARTICIPANT values 153
create manual activity event
  AUDIT_EVENT values 154
  AUDIT_MGMT_ACTIVITY values 150
credential
  shared access module 31
Credential Lease management
  AUDIT_MGMT_LEASE 179, 180
  column values 181, 182
  table columns 181
Credential management 177
  column values 177
  table columns 178
Credential Pool management 178, 179
  column values 179
  table columns 179
Credential. isResetPasswordAtCheckin() object, JavaScript extension 34
Credential.getAccessMode() 32

Credential.isNotifyOnly() object, JavaScript extension 34
Credential.isPasswordViewable() object, JavaScript extension 34

D

default generic workflow templates
  generic workflow default messages 121
default organization management templates
  organization management default messages 125
default shared or application identity management templates
  generic workflow default messages 129
delegate authority 139
DirectoryObject object, JavaScript extension 36
DirectoryObject. getPropertyNames object, JavaScript extension 40
DirectoryObject.addProperty object, JavaScript extension 36
DirectoryObject.dn object, JavaScript extension 37
DirectoryObject.getChanges object, JavaScript extension 38
DirectoryObject.getProperty object, JavaScript extension 38
DirectoryObject.getPropertyAsDate object 39
DirectoryObject.getPropertyAsString object 40
DirectoryObject.name object, JavaScript extension 40
DirectoryObject.profileName object, JavaScript extension 41
DirectoryObject.setProperty, JavaScript extension
  object 42
dynamic tags
  content tags
    examples 115

E

EmailContext object, JavaScript extension 43
enable trace
  command line interface 112
Enrole object, JavaScript extension 45
Enrole.generatePassword object, JavaScript extension 46
Enrole.getAttributeValue object, JavaScript extension 46
Enrole.getAttributeValues object, JavaScript extension 47
Enrole.localize object, JavaScript extension 47
Enrole.log object, JavaScript extension 48
Enrole.logError object, JavaScript extension 48
Enrole.loginfo object, JavaScript extension 49
Enrole.logWarning object, JavaScript extension 50
Enrole.toGeneralizedTime object, JavaScript extension 50
Enrole.toMilliseconds object, JavaScript extension 51
Enrole.traceMax object, JavaScript extension 51
Enrole.traceMid object, JavaScript extension 51
Enrole.traceMin object, JavaScript extension 52
Entitlement workflow management 170, 172
entity operation management 173
entity_name column values 175
Error object, JavaScript extension 53
Error.getErrorCode object, JavaScript extension 54
Error.getMessage object, JavaScript extension 54
Error.setErrorCode object, JavaScript extension 54
Error.setMessage object, JavaScript extension 53
escalate manual activity 149, 154
  AUDIT_MGMT_PARTICIPANT values 155
escalate manual activity event
  AUDIT_EVENT values 157
  AUDIT_MGMT_ACTIVITY values 155
examples
  check-in behavior 193
  check-out behavior 193
  mail templates 120

G

generic workflow default messages
  default generic workflow templates 121
get-credential 100
getRoleName()
  RoleAssignmentAttribute 83
global commands, command line interface 107

I

IBM Personal Communications 190
IBM Security Access Manager for Enterprise Single Sign-On
  check-in 193
  check-out 193
IBM Security Identity Manager
  check-in 193
  check-out 193
ITIM group management
  account management events 168
  table 168

J

Javascript extension
  RoleAssignmentAttribute.getName() 83
  RoleAssignmentAttribute.getRoleDN 84
  RoleAssignmentObject.getAssignedRoleDN() 86
JavaScript extension
  objec
    Person.updateRoleAssignmentData() 64
  object
    activity 17
    Activity.auditEvent 18
    Activity.description 19
    Activity.duedate 19
    Activity.getSubProcesses() 19
    Activity.guid 20
    Activity.id 20
    Activity.index 20
    Activity.name 21
    Activity.participant 21
    Activity.resultDetail 21
    Activity.resultSummary 21
    Activity.setResult 22
    Activity.started 22
    Activity.state 22
    Activity.subtype 23
    Activity.type 23
    AttributeChangeOperation 24
    AttributeChangeOperation.attr 24
    AttributeChangeOperation.op 24
    ContainerSearch 25
    ContainerSearch.searchByFilter 25
    ContainerSearch.searchByURI 26
    Context 26
    Context.getAccountParameter 28
    Context.getActivityResult 28
    Context.getActivityResultById 28
    Context.getLoopCount 29
    Context.getLoopCountByID 29
    Context.getProcessType 30
    Context.getRequestee 30
    Context.getService 30
    Credential.getCheckoutDuration() 32
    Credential.getNotificationRecipient() 33
    Credential.isCheckoutSearchEnable() 33
    Credential.isResetPasswordAtCheckin() 34
    Credential.getNotifyOption() 33
    Credential.isNotifyOnly() 34
    Credential.isPasswordViewable() 34
    DirectoryObject 36
    DirectoryObject.getPropertyNames 40
    DirectoryObject.addProperty 36
    DirectoryObject.dn 37
    DirectoryObject.getChanges 38
    DirectoryObject.getProperty 38
    DirectoryObject.name 40
    DirectoryObject.profileName 41
    EmailContext 43
    Enrole 45

JavaScript extension *(continued)*
object *(continued)*
Enrole.generatePassword 46
Enrole.getAttributeValue 46
Enrole.getAttributeValues 47
Enrole.localize 47
Enrole.log 48
Enrole.logError 48
Enrole.loginfo 49
Enrole.logWarning 50
Enrole.toGeneralizedTime 50
Enrole.toMilliseconds 51
Enrole.traceMax 51
Enrole.traceMid 51
Enrole.traceMin 52
Error 53
Error.getErrorCode 54
Error.getMessage 54
Error.setErrorCode 54
Error.setMessage 53
Participant 54
Participant.implementation 55
Participant.name 56
Participant.type 56
ParticipantType 56
Person 58
Person.
getAllAssignmentAttributes() 59
Person.
getRoleAssignmentData() 60
Person. removeRole 63
Person.
removeRoleAssignmentData() 63
Person.getNewRoles 62
Person.getRemovedRoles 62
Person.getRoleAssignmentData 60
Person.getRoles 61
Person.isInRole 62
PersonSearch 64
PersonSearch.searchByFilter 65
PersonSearch.searchByURI 65
PostOffice 66
PostOffice.
getAllEmailMessages() 66
PostOffice.getEmailAddress 67
PostOffice.getPerson
ByEmailAddress 67
PostOffice.getTopic 67
Process 68
Process.
getRootRequesterName() 72
Process.auditEvent 70
Process.comment 70
Process.description 70
Process.getActivity 70
Process.getParent 71
Process.getRootProcess() 71
Process.getSubProcesses() 72
Process.guid 72
Process.id 73
Process.name 73
Process.parentId 73
Process.requesteeDN 73
Process.requesteeName 74
Process.requestorDN 74
Process.requestorName 74
Process.requestorType 75
JavaScript extension *(continued)*
object *(continued)*
Process.resultDetail 75
Process.resultSummary 75
Process.setRequesteeData 75
Process.setResult 76
Process.setSubjectData 76
Process.started 77
Process.state 77
Process.subject 77
Process.type 78
ProcessData 78
ProcessData.get 78
ProcessData.set 79
Reminder 79
Role 80
Role.getAssignmentAttributes 81
Role.getOwner 82
Role.setAssignmentAttributes 82
RoleSearch 89
RoleSearch.searchByName 90
RoleSearch.searchByURI 90
service 91
ServiceSearch 91
objects 15
Role.getAllAssignmentAttributes 81
RoleAssignment.addProperty
object 87
RoleAssignmentAttribute 83
RoleAssignmentObject.
getChanges() 87
RoleAssignmentObject.
getDefinedRoleDN() 86
RoleAssignmentObject.getProperty
object 88
RoleAssignmentObject.getPropertyNames
object 88
RoleAssignmentObject.removeProperty
object 89
RoleAssignmentObject.setProperty
object 89
ServiceSearch.searchByFilter
object 91
ServiceSearch.searchByName
object 92
ServiceSearch.searchByURI object 93
ServiceSearch.searchForClosestToPerson
object 93

L

lifecycle rule 157
AUDIT_EVENT values 157

M

mail templates
examples 120
mainframe applications 190
methods
RoleAssignmentObject 85
Microsoft Remote Desktop
Connection 189
Microsoft Remote Desktop Services (RDP)
See RDP
Microsoft Remote Desktop Services (RDS)
terminal server
See terminal server

O

object 25, 32
Context.isAccountDataChanged object,
JavaScript extension 31
delegate JavaScript extension 35
DirectoryObject.
removeProperty(name,value) ,
JavaScript extension 41
DirectoryObject.getPropertyAsDate 39
DirectoryObject.getPropertyAsString 40
DirectoryObject.removeProperty ,
JavaScript extension 41
DirectoryObject.setProperty object,
JavaScript extension 42
JavaScript extension
activity 17
Activity.auditEvent 18
Activity.description 19
Activity.duedate 19
Activity.getSubProcesses() 19
Activity.guid 20
Activity.id 20
Activity.index 20
Activity.name 21
Activity.participant 21
Activity.resultDetail 21
Activity.resultSummary 21
Activity.setResult 22
Activity.started 22
Activity.state 22
Activity.subtype 23
Activity.type 23
AttributeChangeOperation 24
AttributeChangeOperation.attr 24
AttributeChangeOperation.op 24
ContainerSearch 25
ContainerSearch.searchByFilter 25
ContainerSearch.searchByURI 26
Context 26
Context.getAccountParameter 28
Context.getActivityResult 28
Context.getActivityResultById 28
Context.getLoopCount 29
Context.getLoopCountByID 29
Context.getProcessType 30
Context.getRequestee 30
Context.getService 30
Credential.
getNotificationRecipient() 33
Credential.
isCheckoutSearchEnable() 33
Credential.
isResetPasswordAtCheckin() 34
Credential.getCheckoutDuration() 32
Credential.getNotifyOption() 33
Credential.isNotifyOnly() 34
Credential.isPasswordViewable() 34
DirectoryObject 36
DirectoryObject.
getPropertyNames 40
DirectoryObject.addProperty 36
DirectoryObject.dn 37

object *(continued)*
JavaScript extension *(continued)*
DirectoryObject.getChanges 38
DirectoryObject.getProperty 38
DirectoryObject.name 40
DirectoryObject.profileName 41
EmailContext 43
Enrole 45
Enrole.generatePassword 46
Enrole.getAttributeValue 46
Enrole.getAttributeValues 47
Enrole.localize 47
Enrole.log 48
Enrole.logError 48
Enrole.loginfo 49
Enrole.logWarning 50
Enrole.toGeneralizedTime 50
Enrole.toMilliseconds 51
Enrole.traceMax 51
Enrole.traceMid 51
Enrole.traceMin 52
Error 53
Error.getErrorCode 54
Error.getMessage 54
Error.setErrorCode 54
Error.setMessage 53
Oerson.isInRole 62
Participant 54
Participant.implementation 55
Participant.name 56
Participant.type 56
ParticipantType 56
Person 58
Person. getAllAssignmentAttributes() 59
Person. getRoleAssignmentData() 60
Person. removeRole 63
Person. removeRoleAssignmentData() 63
Person. updateRoleAssignmentData() 64
Person.getNewRoles 62
Person.getRemovedRoles 62
Person.getRoleAssignmentData 60
Person.getRoles 61
PersonSearch 64
PersonSearch.searchByFilter 65
PersonSearch.searchByURI 65
PostOffice 66
PostOffice. getAllEmailMessages() 66
PostOffice.getEmailAddress 67
PostOffice.getPerson ByEmailAddress 67
PostOffice.getTopic 67
Process 68
Process. getRootRequesterName() 72
Process.auditEvent 70
Process.comment 70
Process.description 70
Process.getActivity 70
Process.getParent 71
Process.getRootProcess() 71
Process.getSubProcesses() 72
Process.guid 72
object *(continued)*
JavaScript extension *(continued)*
Process.id 73
Process.name 73
Process.parentId 73
Process.requesteeDN 73
Process.requesteeName 74
Process.requestorDN 74
Process.requestorName 74
Process.requestorType 75
Process.resultDetail 75
Process.resultSummary 75
Process.setRequesteeData 75
Process.setResult 76
Process.setSubjectData 76
Process.started 77
Process.state 77
Process.subject 77
Process.type 78
ProcessData 78
ProcessData.get 78
ProcessData.set 79
Reminder 79
Role 80
Role.getAssignmentAttributes 81
Role.getOwner 82
Role.setAssignmentAttributes 82
RoleSearch 89
RoleSearch.searchByName 90
RoleSearch.searchByURI 90
service 91
ServiceSearch 91
organization management default messages
default organization management templates 125
organization role management 166

P

Participant object, JavaScript extension 54
Participant.implementation object, JavaScript extension 55
Participant.name object, JavaScript extension 56
Participant.type object, JavaScript extension 56
ParticipantType object, JavaScript extension 56
person management 137
Person object, JavaScript extension 58
Person. getAllAssignmentAttributes(), JavaScript extension 59, 60
Person. removeRoleAssignmentData(), JavaScript extension 63
Person. removeRoles object, JavaScript extension 63
Person. updateRoleAssignmentData(), JavaScript extension 64
Person.getNewRoles object, JavaScript extension 62
Person.getRemovedRoles object, JavaScript extension 62
Person.getRoleAssignmentData, JavaScript extension 60
Person.getRoles object, JavaScript extension 61
Person.isInRole object, JavaScript extension 62
PersonSearch object, JavaScript extension 64
PersonSearch.searchByFilter object, JavaScript extension 65
PersonSearch.searchByURI object, JavaScript extension 65
policy
management 140
PostOffice object, JavaScript extension 66
PostOffice. getAllEmailMessages(), JavaScript extension 66
PostOffice.getEmailAddress object, JavaScript extension 67
PostOffice.getPersonByEmailAddress object, JavaScript extension 67
PostOffice.getTopic object, JavaScript extension 67
Process object, JavaScript extension 68
Process. getRootRequesterName(), JavaScript extension 72
Process.auditEvent object, JavaScript extension 70
Process.comment object, JavaScript extension 70
Process.description object, JavaScript extension 70
Process.getActivity object, JavaScript extension 70
Process.getParent object, JavaScript extension 71
Process.getRootProcess(), JavaScript extension 71
Process.getSubProcesses(), JavaScript extension 72
Process.guid object, JavaScript extension 72
Process.id object, JavaScript extension 73
Process.name object, JavaScript extension 73
Process.parentId object, JavaScript extension 73
Process.requesteeDN object, JavaScript extension 73
Process.requesteeName object, JavaScript extension 74
Process.requestorDN object, JavaScript extension 74
Process.requestorName object, JavaScript extension 74
Process.requestorType object, JavaScript extension 75
Process.resultDetail object, JavaScript extension 75
Process.resultSummary object, JavaScript extension 75
Process.setRequesteeData object, JavaScript extension 75
Process.setResult object, JavaScript extension 76
Process.setSubjectData object, JavaScript extension 76

Process.started object, JavaScript extension 77
Process.state object, JavaScript extension 77
Process.subject object, JavaScript extension 77
Process.type object, JavaScript extension 78
ProcessData object, JavaScript extension 78
ProcessData.get object, JavaScript extension 78
ProcessData.set object, JavaScript extension 79
PuTTY, log on 188

R

RDP 189
Reconciliation 172
register-additional-instance 98
register-first-instance 96
Reminder object, JavaScript extension 79
Remote Desktop Protocol (RDP)
  See RDP
Remote Desktop Services (RDS) RDP
  See RDP
remote terminals 188
Role object, JavaScript extension 80
Role.getAllAssignmentAttributes object
  JavaScript extension 81
Role.getAssignmentAttributes object, JavaScript extension 81
Role.getOwner object, JavaScript extension 82
Role.setAssignmentAttributes object, JavaScript extension 82
RoleAssignment.addProperty object
  JavaScript extension 87
RoleAssignmentAttribute
  getRoleName() 83
RoleAssignmentAttribute object, JavaScript extension 83
RoleAssignmentAttribute. getName()
  Javascript extension 83
RoleAssignmentAttribute. getRoleDN
  Javascript extension 84
RoleAssignmentObject
  methods 85
RoleAssignmentObject. getAssignedRoleDN()
  Javascript extension 86
RoleAssignmentObject. getChanges()
  JavaScript extension 87
RoleAssignmentObject. getDefinedRoleDN()
  JavaScript extension 86
RoleAssignmentObject.getProperty object
  JavaScript extension 88
RoleAssignmentObject.getPropertyNames object
  JavaScript extension 88
RoleAssignmentObject.removeProperty object
  JavaScript extension 89
RoleAssignmentObject.setProperty object
  JavaScript extension 89
RoleSearch object, JavaScript extension 89
RoleSearch.searchByName object, JavaScript extension 90
RoleSearch.searchByURI object, JavaScript extension 90
runtime events 176

S

SA_VAULT_SERVICE_ALIAS table 1
schema
  access request management 145
  create manual activity 149
  escalate manual activity 149, 154
  lifecycle rule 157
SecureCRT, log on 188
self-password change 177
service object, JavaScript extension 91
ServiceSearch object, JavaScript extension 91
ServiceSearch.searchByFilter object
  JavaScript extension 91
ServiceSearch.searchByName object
  JavaScript extension 92
ServiceSearch.searchByURI object
  JavaScript extension 93
ServiceSearch.searchForClosestToPerson object
  JavaScript extension 93
shared access
  JavaScript APIs 9
shared access module
  credential 31
Shared Access Policy management 182
shared or application identity management default messages
  default shared or application identity management templates 129
SQL Server 190
SQL Server Management Studio 190
sub sections, IBM Security Privileged Identity Manager commands 105
system configuration 174

T

terminal host 188
terminal server 188

V

virtual appliance 192
  command line interface 105
virtual machine 192
VMware vSphere Client 192

W

workflow
  account request 187

Printed in USA