SailPoint Quick Reference Guide for Gateway Connectors

Version 7.2

Table of Contents

About this guide

This document provides the necessary installation and configuration steps with which to configure the Gateway Connectors as an individual application within IdentityIQ for compliance and provisioning activities.

Note: The following Gateway Connectors (Agents) residing on mainframe security systems communicate with IdentityIQ through the Connector Gateway:
- RACF
- CA-ACF2
- CA-Top Secret

Gateway Connector Licensing

Customers who licensed IdentityIQ on or after July 15, 2013 are entitled to all IdentityIQ connectors for aggregation, provisioning and password management use cases.

Customers who licensed IdentityIQ before July 15, 2013 are entitled to use connectors for reading data (aggregation). However, entitlement to provisioning and password management functionality through the IdentityIQ connectors (whether Direct, Gateway or Agent based) requires the purchase of the SailPoint Provisioning Engine.

Related documentation

In addition to this book, you can find useful information in the publications listed in the following table. These publications are available on request from SailPoint IdentityIQ.

The following related publications supplement this book:

Document	Description
SailPoint IdentityIQ Installation Guide	Provides detailed information about the installation, deployment and configuration of IdentityIQ.
SailPoint IdentityIQ Administration Guide	Provides detailed information about Configuration, Provisioning, System Administration, and Password Management.
SailPoint Integration Guide	Provides detailed information about Provisioning Integration, Service Desk Integration and Resource Connectors.

Chapter 1: Overview

The following topics are discussed in this chapter:

What is SailPoint IdentityIQ?

SailPoint IdentityIQ is unique and comprehensive Identity Governance solution that addresses the digital security concerns of today's global enterprises. The product provides automated analysis, monitoring and management of risk associated with access to critical or sensitive applications and data via centralized visibility to and control of identity data across the enterprise. Through the use of patent-pending technologies and annalistic, IdentityIQ improves security, lowers the cost of operations, and improves an organization's ability to meet compliance and governance demands.

SailPoint IdentityIQ is a business-oriented identity governance solution that delivers risk-aware compliance management, adaptive role management, access request management, and identity intelligence. Some of the world’s largest organizations are using IdentityIQ to improve security, minimize risk and streamline their compliance efforts.

IdentityIQ minimizes the access burden placed on IT staff by empowering end users across the organization to request and manage their own access.

Organizations strive for better visibility into potential risk factors across their business. With Identity Intelligence from IdentityIQ, organizations can transform technical identity data scattered across multiple enterprise systems into centralized, easily understood and business-relevant information. The visibility and insights offered by IdentityIQ through dashboards, risk metrics and reporting provide a clear understanding of identity and access information and help to pro actively manage and focus compliance efforts strategically across even the most complex enterprise environments.

SailPoint IdentityIQ components

SailPoint IdentityIQ includes the core components illustrated in Figure 1—SailPoint IdentityIQ components.

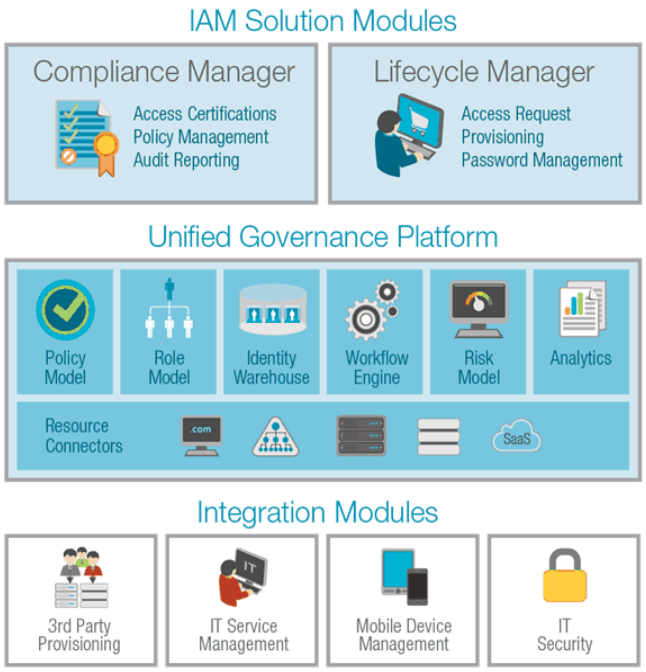


Figure 1—SailPoint IdentityIQ components

Together, these components provide a unique and comprehensive identity governance solution. The components are:

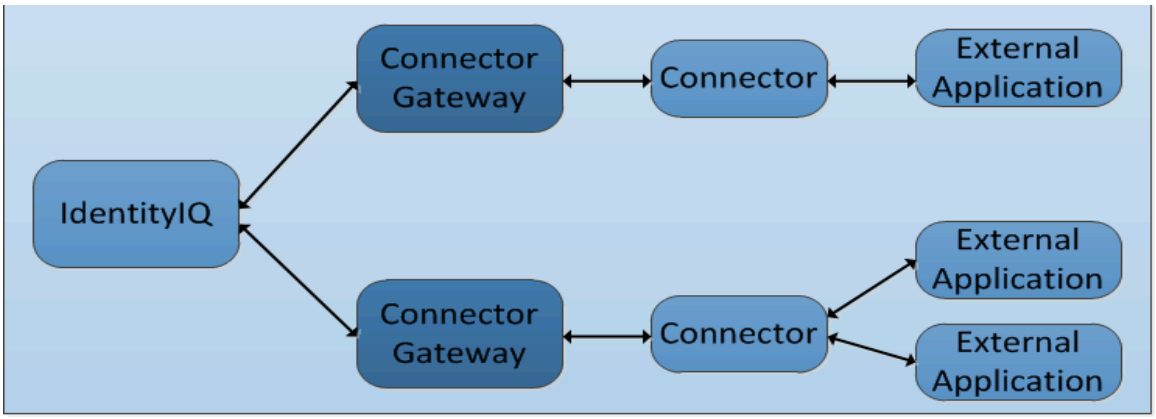
- IAM Solution Modules: The IAM Solution Modules consists of the following modules:
  - **Compliance Manager**: IdentityIQ Compliance Manager automates access certifications, policy management, and audit reporting through a unified governance framework. This allows you to streamline compliance processes and improve the effectiveness of identity governance-all while lowering costs.
  - **Lifecycle Manager**: IdentityIQ Lifecycle Manager manages changes to access through user-friendly self-service request and password management interfaces and automated lifecycle events. It provides a flexible, scalable provisioning solution for addressing the constantly evolving access needs of your business in a way that's both efficient and compliant.
- **Unified Governance Platform**: IdentityIQ's Governance Platform lays the foundation for effective IAM within the enterprise. It establishes a single framework that centralizes identity data, captures business policy, models roles, and takes a risk-based, proactive approach to managing users and resources. The Governance Platform also offers extensible analytics that transforms disparate, technical identity data into relevant business information. Additionally, robust resource connectivity is provided that allows organizations to directly connect to applications running in the datacenter or in the cloud.

  A unified governance platform allows organizations to build a single preventive and detective control model that supports all identity business processes, across all applications-in the datacenter and the cloud. SailPoint IdentityIQ applies consistent governance across compliance, provisioning and access management processes, maximizing investment and eliminating the need to buy and integrate multiple products.

- **Integration Modules**: The Integration Modules consists of the following modules:
  - 3rd Party Provisioning
  - IT Service Management
  - Mobile Device Management
  - IT Security

IdentityIQ Agent architecture

The Agent architecture diagram:



The Connector (Gateway Connectors) as shown in the above figure executes the requests (Aggregation, Provisioning Entitlements and Role Assignments, and Password Management) sent by the Provisioning Engine on the target application.

The Connector Gateway accepts concurrent requests originating from one or more IdentityIQ instances. Each request maintains its dedicated communication channel to the Connector Gateway for sending and receiving information necessary to perform the request. The Connector Gateway forwards these requests to the Connector.

The Connectors provide communications between User Administration components and the managed system. The Connectors are bound to the application they manage. Each supported application has an associated Connector. To service requests for a given application, it's associated Connector is required. Refer the "Applications supported and associated connectors" on page 6 for the complete list of the supported applications and their associated Connector.

The information exchanged between IdentityIQ and the Connector can be secured using an encrypted channel.

Before creating an application, ensure that the following components are installed and configured:

- **Connector**: installed and configured to manage the targeted application.

  For more information on individual installations of the connector, see the respective Administration Guides as follows:
  - *SailPoint IdentityIQ Connector for CA-Top Secret Administration Guide*
  - *SailPoint IdentityIQ Connector for RACF Administration Guide*
  - *SailPoint IdentityIQ Connector for CA-ACF2 Administration Guide*

- **Connector Gateway**: for each Connector, a Connector Gateway is required to communicate with the Connector.

  For more information on installation and configuration of Connector Gateway, see Chapter 2, “Installing and Configuring of Connector Gateway.”

Applications supported and associated connectors

The following table lists the supported applications and associated connectors:

Table 1—Supported applications and associated connectors

Application	Connector to be used	Application Type
RACF	SailPoint IdentityIQ Connector for RACF	RACF - Full
CA-ACF2	SailPoint IdentityIQ Connector for CA-ACF2	ACF2 - Full
CA-TopSecret	SailPoint IdentityIQ Connector for CA-Top Secret	TopSecret - Full

Supported features

SailPoint IdentityIQ Gateway Connectors support the following features:

- **Create/Update/Delete/Unlock/Enable/Disable on Account**: Ability for user to perform such account provisioning operations.
- **Change/Reset Password**: Ability for user to change and reset password.
- **Group provisioning**: Group provisioning is supported for all Gateway Connectors in IdentityIQ. Group provisioning supports add, delete and update group.
- **Password Interception**: SailPoint IdentityIQ provides the capability of Password Interception, by which password of an account changed on managed system is intercepted to IdentityIQ.

  New password is intercepted and sent to IdentityIQ to be propogated to other managed systems on which the account is defined.

  For proper functioning of the Password Interception, the Password Interception settings are mandatory. For more information on setting the Password Interception from IdentityIQ, see “Settings for configuring Password Interceptor and Online Interceptor” on page 28.
- **Target aggregations**: Aggregation of permissions on objects.

  Note: After upgrading to IdentityIQ version 7.2, perform target aggregation again and configure certifications as the old formatted permissions can not be used to revoke permissions.
- **Auto revocation of accounts and group permissions:** SailPoint IdentityIQ provides the ability of automatic revocations of accounts and groups.
- **Online Interception:** SailPoint IdentityIQ provides the capability of Online Interception, by which any account or group or account-group connection changed on managed system is intercepted to IdentityIQ.

  For proper functioning of the Online Interception, the Online Interception settings are mandatory (Same settings as for Password Interception, so this should be defined only once in IdentityIQ). For more information on setting the Online Interception from IdentityIQ, see "Settings for configuring Password Interceptor and Online Interceptor" on page 28.

- **TLS support**: SailPoint IdentityIQ provides support for TLS 1.2 for connection between Provisioning Engine and Connector Gateway.

  For more information on implementing the support for TLS 1.2, see "Enabling TLS Communication" on page 23.

Chapter 2: Installing and Configuring of Connector Gateway

This chapter provides the steps for the installation and configuration of Connector Gateway.

The following topics are discussed:

Overview

Note: Java Development Kit 1.7 (32-bit and 64-bit) or later is required.

The Connector Gateway can be installed on any computer with network connectivity to IdentityIQ and the Gateway Connector (Agent).

Create an installation directory on the computer where the Connector Gateway will run and extract the contents of the `ConnectorGateway-releaseVersion.zip` file that is included with IdentityIQ, where *releaseVersion* is the version of the current IdentityIQ release. Use the operating system-specific installation instructions located in the associated README file included in the `ConnectorGateway-releaseVersion.zip`.

Before you begin

- The Connector Gateway and Agent can be started / stopped in any order.
- If there is a firewall between Connector Gateway and Connector, then the port and port +1 on which Connector is listening must be opened.

Additional configuration parameters on IdentityIQ Server

This section describes the default values of the parameters (System Configuration and Application Configuration) of the IdentityIQ Debug page.

System configuration parameters

- **syncSchema**: If set to **true**, IdentityIQ application schema will be synchronized with the Connector.
  Default value is true.

  For example, `<entry key="syncSchema" value="true"/>`
- **smReadTimeout**: Timeout value in minutes for messages exchanged between IdentityIQ and the Connector for a transaction.
  Default value is 10 minutes.

  For example, `<entry key="smReadTimeout" value="10"/>`
- **smSocketConnectRetry**: Controls the number of times Connector Gateway tries to send the transaction if a connection cannot be initiated.
  Default value is 10.

  For example, `<entry key="smSocketConnectRetry" value="15"/>`

- **smMaxActiveTransactions**: Maximum transactions that can be initiated from IdentityIQ.
  Default value is 10.

  For example, `<entry key="smMaxActiveTransactions" value="15"/>`

Application configuration parameters

- **multiColumnSeperator**: Delimiter for multi-column attributes.
  Default value is '#'.

  If the separator needs to be changed other than the default value then change the separator into the application template as displayed in the following example:

  ```
  <entry key="multiColumnSeperator" value="+" />
  ```
- **IBMcharacterSet**: The related character set in IdentityIQ for code page used on the Connector side.
  Default value is IBM1047.

  The value can be modified as follows:

  ```
  <entry key="IBMcharacterSet" value="IBM1047"/>
  ```

  For more information on the Code page related data, see the following link:

  http://www-01.ibm.com/software/globalization/ccsid/ccsid_registered.html
- **disableHotpathForUpdate:** Disables hotpath for all update account operations. This prevents IdentityIQ database to be updated with the changes (if any) from the Managed system for all update operations. If this parameter is set to true then **disableHotpathForChgPwd** and **disableHotpathForManageAcntOp** are ignored and the hotpath will be disabled.

  Default value is false (hotpath is enabled).

  To disable the hotpath for all update operations, set the value to true as

  follows:

  ```
  <entry key="disableHotpathForUpdate" value="true"/>
  ```
- **disableHotpathForChgPwd:** Disables hotpath for Change Password operation only. If a hotpath is disabled, the connector executes the query and sends the status immediately to improve the performance. By disabling the hotpath, the latest information from Connector is not updated in IdentityIQ database.

  Default value is false (hotpath is enabled).

  To disable the hotpath for Change password, set the value to true as follows:

  ```
  <entry key="disableHotpathForChgPwd" value="true"/>
  ```
- **disableHotpathForManageAcntOp:** Disables hotpath for Enable, Disable and Unlock operations. This prevents IdentityIQ database to be updated with the changes (if any) from the Managed system for Enable, Disable and Unlock operations.

  Default value is false (hotpath is enabled).

  To disable the hotpath for Enable, Disable, and Unlock operations, set the value to true as follows:

  ```
  <entry key="disableHotpathForManageAcntOp" value="true"/>
  ```
- **useMSAdminDefined**: Define Managed System Administrator in provisioning plan which can be used by the Connector as a Managed System Administrator when a transaction is initiated.

  Default vale is N (functionality is disabled).

  To enable this functionality, set the value to Yes (Y) as follows:

  ```
  <entry key=”useMSAdminDefined” value=”Y”/>
  ```
- **splAccountAttributes**: Set the attributes which are not specified in schema but are expected to be sent/received by IdentityIQ. For example, **RU_LOCKED**. The field is mapped to be locked, hence this field is not required in schema.

```
<entry key="splAccountAttributes">
      <value>
        <Map>
          <entry key="RU_LOCKED" value="false"/>
          <entry key="RU_SUSPENDED" value="false"/>
        </Map>
      </value>
</entry>
```

- **splConnectionAttributes**:
  - (**For CA-ACF2 and CA-Top Secret**) Add a attribute in the following tag which is a Account-Group connection attribute on Connector. The value can be read for the Account in IdentityIQ.

```
<entry key="splConnectionAttributes">
    <value>
       <Map>
          <entry key="CONNECTION_ATTRIBUTE_1" value="false"/>
          <entry key="CONNECTION_ATTRIBUTE_2" value="false"/>
       </Map>
    </value>
</entry>
```

  - (**For RACF only**) For account or group connection, to set the group as the connection owner for that group, add the following attribute manually to the RACF-Full.xml file when upgrading IdentityIQ from any previous version to IdentityIQ version 7.2:

```
<entry key="splConnectionAttributes">
  <value>
    <Map>
       <entry key="OWNER" value="true"/>
    </Map>
  </value>
</entry>
```

    Note: One connection/group is mandatory to create a RACF account. One of the requested groups would be marked as the Default Group for that account.

- **UserAdminMap**: USER_ADMIN field for account is not in readable format and has values like 1,2,3 and so on. To make it readable a map can be added in Application definition as follows:

```
<entry key="UserAdminMap">
   <value>
     <Map>
       <entry key="1" value="User"/>
       <entry key="2" value="Auditor"/>
       <entry key="3" value="Administrator"/>
       <entry key="4" value="Both"/>
       <entry key="5" value="Ignore"/>
     </Map>
   </value>
</entry>
```

  Run the aggregation task to view the changes in IdentityIQ.

  Note: While setting the value of USER_ADMIN field from IdentityIQ, user must enter the value as defined in the MAP. For example, to set USER_ADMIN field to user in above map the value must be 'User' and not 1.

- **limitNumberOftransactions**: To limit the number of transactions that can be initiated from IdentityIQ, add limitNumberOftransactions parameter in the application debug page as follows:

  ```
  <entry key="limitNumberOftransactions" value="true"/>
  ```

Target Aggregation configuration parameters

For implementing the target aggregation, the following configuration parameters must be updated:

- **splAceAttributes:** The list that contains keywords to synchronize resource entity.
- **splResourceAttributes:** The list that contains keywords to synchronize ACE entity.

  Note: It is recommended not to modify the list of keywords defined in splAceAttributes and splResourceAttributes.
  If the POS attribute not found error is displayed on Agent for deleting the permissions, perform the procedure specified in “Adding POS keyword for ACF2 application” on page 32.

- **splTargetPermissionsInterestingKwds**: The keywords mentioned in **splTargetPermissionsInterestingKwds** key are subset of keywords defined in **splAceAttributes** key that is used to form the permissions string. These keywords are not sent to Mainframe Connector while keyword synchronization.
- **fieldDelimiter** and **listDelimiter:** The delimiters for field separator and list separator can be modified in the respective application registry file. The default values for field and list delimiter are **#** and **;** respectively.

  For more information on modifying the delimiters, see “Modify the delimiter” on page 22.

For more information on the above parameters, see “Implementing Target Aggregation” on page 19.

Configuring Connector Gateway

This section describes the installation and uninstallation procedures of Connector Gateway.

Installation

This section describes the installation procedures for Windows and UNIX operating systems.

For Microsoft Windows

1. Download the `ConnectorGateway-releaseVersion.zip` file from IdentityIQ home directory to a temporary directory.

   Note: The Connector Gateway and the IdentityIQ version must be same.

2. Unzip the `ConnectorGateway-releaseVersion.zip` file in the temporary directory.

The following files are extracted:

- `init.xml`
- `install.bat`
- `log4j.properties`
- `ConnectorGateway.jar`
- `ConnectorGateway32.exe`
- `ConnectorGateway64.exe`
- `README_UNIX`
- `README_WIN`

3. Edit the `install.bat` file and set `JDK_HOME` to the root directory of your JDK installation.
4. Run the following `install.bat` file, to install Connector Gateway on 32-bit or 64-bit:

   - (*32-bit*): `install.bat 32`
   - (*64-bit*): `install.bat 64`

   And verify if `IdentityIQConnectorGateway_<service name>` service has been installed, where `<service name>` is the name of the service to which the Connector Gateway is connected during the installation of Gateway Connector.
5. Edit `init.xml` and set the host and port details for the Connector and the listening port for the Connector Gateway in the Server tag.

   The parameters of the `init.xml` file are as follows:

Parameters	Description
hostname	Hostname of the Connector.
port	Port number of the Connector. For example, the port number in the above sample file of `init.xml` is **8260**.
<port>	(Server tag) Listening port number of the Connector Gateway. For example, the listening port number under the server tag in the above sample file of `init.xml` is **2450**.
agent_connection_restart_timeout	(*Optional*) Add this parameter as follows under the `<server>` tag of the `init.xml` file of Connector Gateway to define Connector Gateway - Connector connection timeout interval if the Connector does not respond to any transaction: `<agent_connection_restart_timeout>10</agent_connection_restart_timeout>` For example, the value of this parameter is set to 10 in the above sample file of `init.xml`.

Parameters	Description
send_keep_alive_to_all	(*Optional*) Connector Gateway does not send a keep alive message to all the Active IdentityIQ/Mainframe Connectors connections except one configured for interceptions. If user wants Connector Gateway to send keep alive message to all the active IdentityIQ/Agent based mainframe connectors connections, add the following parameter to the `init.xml` file of Connector Gateway: `<send_keep_alive_to_all>true</send_keep_alive_to_all>` For example, the value of this parameter is set to **true** in the above sample file of `init.xml`.

Note: Restart Connector Gateway after `init.xml` file is modified.

6. Start the `IdentityIQConnectorGateway_<service name>` service.

For UNIX

1. Download the `ConnectorGateway-releaseVersion.zip` file from IdentityIQ home directory to a temporary directory.
2. Unzip the `ConnectorGateway-releaseVersion.zip` file in the temporary directory.
   The following files are extracted:
   - `init.xml`
   - `install.bat`
   - `log4j.properties`
   - `ConnectorGateway.jar`
   - `ConnectorGateway32.exe`
   - `ConnectorGateway64.exe`
   - `README_UNIX`
   - `README_WIN`
3. Add the `bin` directory of `Java Development Kit` to the system `PATH` variable.
4. Edit `init.xml` and set the host and port details for the Agent and the listening port for the gateway in the Server tag.
5. Run the following to start the Connector Gateway:
   `java -jar ConnectorGateway.jar`

Uninstallation

This section describes the uninstallation procedures for Windows and UNIX operating systems.

For Microsoft Windows

1. Stop the `IdentityIQConnectorGateway_<service name>` Windows service.

2. Run following command from command prompt, to uninstall Connector Gateway installed on 32-bit or 64 bit Java run:
   - (*32-bit*): `ConnectorGateway32.exe //DS//"IdentityIQConnectorGateway_<service name>"`
   - (*64-bit*): `ConnectorGateway64.exe //DS//"IdentityIQConnectorGateway_<service name>"`
3. Manually remove the installation directory.

For UNIX

1. Stop the `IdentityIQConnectorGateway` service.
2. Manually remove the installation directory.

Chapter 3: Configuring IdentityIQ Gateway Connectors

The following topics are discussed in this chapter:

Configuration

Configuring the Provisioning Engine involves defining an Application in IdentityIQ for each of the target applications. When creating an Application within IdentityIQ, the following information is required:

- Application type
- Connector Gateway details
- Connector details
- Encryption details

Note: Version of IdentityIQ and Connector Gateway must be same.

Application type

- Select the type of application that IdentityIQ is managing.
- See the “Applications supported and associated connectors” on page 6 for the list of supported applications and their respective application types.

Connector Gateway details

Each application that IdentityIQ is managing must have a dedicated Connector Gateway. When defining an application following of the Connector Gateway are required:

Attributes	Description
Connector Gateway host	Hostname/IP of the computer where the Connector Gateway for the targeted Application is installed.
Connector Gateway port	Port on which the Connector Gateway is listening.

Connector details

A single Connector manages one or more instances of an associated application type. Each instance of the application managed by a Connector requires a separate configuration to create a configuration set unique to that instance of the application. The information below is required to establish this differentiation:

Note: The values of the attributes referenced below are case sensitive and if provided incorrectly, will cause requests from IdentityIQ to fail.

Attributes	Description
MSCS Name	Name for the managed system configuration set that is defined when installing the Connector Manager and Connector.
Username	User account used when authenticating to the Target Application through the Connector.
Password	Password for the user account specified in Username. **Note: Special type of user as authentication user is supported in application configuration for Mainframe Connectors. For such type of users defined, password field must be kept blank. The special type of users are as follows:** • **Protected for RACF** • **Restricted for ACF2 and** • **With no Password (NOPW) for CA-Top Secret**
Enable TLS	Select this checkbox if the connection is over Transport Layer Security (TLS). For more information, see “Enabling TLS Communication” on page 23.

Encryption details

The communication between IdentityIQ and the Connector can be secured by using either DES or Triple DES encryption algorithm. For more information on enabling secured communication, see “Enabling secured communication between IdentityIQ and Gateway Connectors” on page 22. The following information is required to enable secured communication:

- **Encryption**: the Encryption Algorithm used to secure the communication channel.
- **Encryption Key File**: fully qualified file name of the encryption key file on the IdentityIQ server. For more information on enabling secured communication, see “Enabling secured communication between IdentityIQ and Gateway Connectors” on page 22.

Click Test Connection to check if entered values are correct.

Note: When aggregation is in progress, do not perform the test connection.

Testing the Application

The **Test Connection** button in the Application definition web page can be used to test basic connectivity between IdentityIQ and the managed application. Additional tests can be run using the IdentityIQ integration console.

1. Launch the console by using the following IdentityIQ script in the `WEB-INF/bin` directory of the IdentityIQ installation to run IdentityIQ integration:

```
iiq integration
```

2. When the command prompt appears enter the following:
   ```
   list
   ```
   This lists the names of all Applications created in system.
3. From the console enter the following:
   ```
   use applicationName
   ```
   This makes the Application active for further console commands.
4. Enter the following:
   ```
   ping
   ```
   The following response is displayed:
   ```
   Response: Connection test successful
   ```
   To verify the requests that are transmitting over the network, and how they are being processed, enable the `log4j.properties` file as mentioned in “8 - Enable traces”.

Target Aggregation implementation in Gateway Connectors

Users can use Target aggregation feature to aggregate User, Group permissions of each mainframe based application users and group which has permissions on Mainframe resources (such as datasets , CICS...). In addition, it is also possible to revoke these permissions from IdentityIQ so the relevant Gateway Connectors will drop the relevant permission. The **Target Aggregation** functionality is supported only for the following Gateway Connectors:

- CA-Top Secret
- CA-ACF2
- RACF

Implementing Target Aggregation

To implement Target Aggregation Users must perform the following steps:

1. Synchronize keywords
2. Unstructured Target Configuration on Mainframe application
3. Execute a task to aggregate resources
4. Run certification process to revoke permissions

1 - Synchronize keywords

To aggregate the permissions, keywords for resource and ACE must be synchronized. Users must perform Test Connection for Mainframe application. Following are the three set of keywords maintained for Target Aggregation:

- splAceAttributes
- splResourceAttributes
- splTargetPermissionsInterestingKwds

Users can add/remove these keywords from the list by modifying and importing the following respective application registry file in IdentityIQ for Mainframe connector:

- RACF-Full.xml
- ACF2-Full.xml
- TopSecret-Full.xml

If any keyword must be added/deleted in the permissions list then adding/deleting the keyword in **splTargetPermissionsInterestingKwds** tag and running the target aggregation will populate the value of the keyword on record.

For example, if the **splTargetPermissionsInterestingKwds** contains the following list of keywords as in `RACF-Full.xml` file:

```
<entry key="splTargetPermissionsInterestingKwds">
  <value>
    <Map>
       <entry key="ACCESS" value="false"/>
       <entry key="INFO.ACC_COUNT" value="false"/>
       <entry key="WHEN.APPCPORT" value="false"/>
       <entry key="WHEN.CONSOLE" value="false"/>
       <entry key="WHEN.JESINPUT" value="false"/>
       <entry key="WHEN.PROGRAM" value="false"/>
       <entry key="WHEN.SYSID" value="false"/>
       <entry key="WHEN.TERMINAL" value="false"/>
    </Map>
  </value>
</entry>
```

And if the user does not want IdentityIQ to display the fields starting with WHEN, then user must modify the list to the following in the application debug page:

```
<entry key="splTargetPermissionsInterestingKwds">
   <value>
      <Map>
        <entry key="ACCESS" value="false"/>
        <entry key="INFO.ACC_COUNT" value="false"/>
      </Map>
   </value>
</entry>
```

Note: Users can revoke only the permissions aggregated in IdentityIQ. In above scenario the permissions ACCESS and INFO.ACC_COUNT can only be revoked from IdentityIQ.

2 - Unstructured Target Configuration on Mainframe application

1. Configure unstructured targets for the application you want to execute target aggregation.

Provide the following details on Unstructured Target Configuration:

- **Name**: Provide name to the created unstructured configuration.
  For example, `RACFUnstructuredTargetConfig`.

- **Correlation Rule**: Create a correlation rule to correlate the permissions and entities in IdentityIQ.

The following sample code can be used for correlation:

```
import sailpoint.api.Correlator;
import sailpoint.tools.xml.XMLObjectFactory;
      private String ATTR_USER_OBJECT= "USER_ID";
      private String ATTR_GROUP_OBJECT= "GROUP_ID";
      Map returnMap = new HashMap();
  if ( isGroup ) {
         returnMap.put(Correlator.RULE_RETURN_GROUP_ATTRIBUTE,"nativeIdentity");
          returnMap.put(Correlator.RULE_RETURN_GROUP_ATTRIBUTE_VALUE, nativeId);
      } else {
      returnMap.put(Correlator.RULE_RETURN_LINK_IDENTITY, nativeId);
}
    System.out.println("Incomming ["+nativeId+"] correlated
["+XMLObjectFactory.getInstance().toXml(returnMap)+"]");
      return returnMap;
```

2. Depending on the type of the mainframe connector, select **Target Source type** and other attributes:

   - For application type **RACF Full**:

     Target Source type: **PE2 RACFCollector**

Attributes	Description
RACF Target Resource Name	Provide RACF Target Resource name.
Target Resource Type	Provide RACF Target Resource Type.
Volume	Provide RACF Target Resource volume.
Resource Get Type	Specify ID or PREFIX.
Generic Type	Specify Yes (Generic) or No (non-Generic).

   - For application type **Top Secret Full**:

     Target Source type: **PE2 TSSCollector**

Attributes	Description
TSS Target Resource Name	Provide TSS Target Resource name.
Target Resource Type	Provide TSS Target Resource Type.
Resource Get Type	Specify ID or PREFIX.

   - For application type **CA-ACF2**:

     Target Source type: **PE2 ACF2Collector**

Attributes	Description
ACF2 Target Resource Name	Provide ACF2 Target Resource name.

Attributes	Description
Target Resource Type	Provide ACF2 Target Resource Type.
ACF2 ruleset Type	None (specific) or PREFIX or NEXTKEYS.

Modify the delimiter

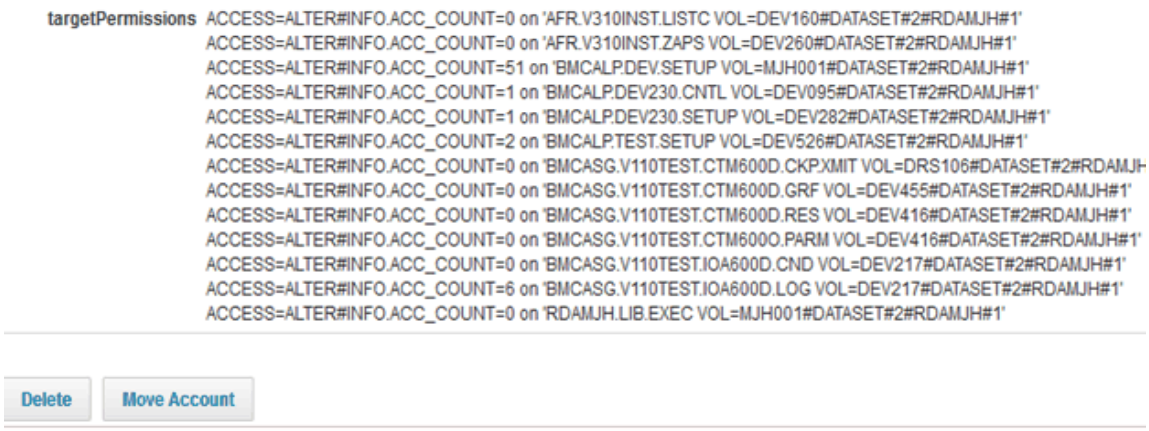
The resources and permissions will be updated in the respective account as a string in delimited format. Customer can define the delimiters for field separator and list separator by modifying the following keys in respective application registry file in IdentityIQ for Mainframe connector:

```
<entry key="fieldDelimiter" value="#"/>
<entry key="listDelimiter" value=";"/>
```

The default values for field and list delimiter are **#** and **;** respectively.

3 - Execute a task to aggregate resources

To aggregate permissions create New Target Aggregation task. Enter the name for the task and select Mainframe application to scan targets from the list and the Unstructured Target configured for the application. When the task gets completed successfully you can view the permissions on the Account property as displayed in the following figure:



4 - Run certification process to revoke permission

To revoke permission through certification run **Refresh Identity Cube** and **Refresh Entitlement Correlation** tasks before running certification and follow the certification process.

Enabling secured communication between IdentityIQ and Gateway Connectors

This section provides the steps used to enable secured communication between IdentityIQ and Gateway Connectors. To secure the communication channel between IdentityIQ and Gateway Connectors, IdentityIQ uses either DES or Triple DES for encrypting the exchanged data. Regardless of the type of encryption used, an

encryption key file is required. You can use one common encryption key file for all the Gateway Connectors or use separate key files for each Gateway Connector.

On IdentityIQ computer

To create an encryption key file, perform the following steps for each key file required:

1. From the command prompt change the directory to *IIQFolder*\WEB-INF\bin\<*OS_Flavor*>
2. Run the following command:
   `kgen <encryption_key_file_name>`

   Note: The complete *fileName* (including the path and extension) is specified while creating an IdentityIQ Application.

On computer where Connector is installed

Perform the following steps for each Gateway Connector.

1. Copy the encryption key file created above to `<olprefs>.<olvers>.ENCREXT` file.
   where:
   - `<olprefs>` is the value set in the OLPREFS parameter in member INSTALLD in the Connector INSTALL library.
   - `<olvers>` is the value set in the OLVERS parameter in member INSTALLD in the Connector INSTALL library.
2. Edit the `<ilprefs>.<ilvers).PARM CTSPARM` member and modify the following parameters:
   - FOREVERY ENCR_EXT_ACT: set the value to **Y**
   - FOREVERY FILE_ENCREXT: set the value to **ENCREXT**

   where:
   - `<ilprefs>` is the value set in the ILPREFS parameter in member INSTALLD in the Connector INSTALL library.
   - `<ilvers>` is the value set in the ILVERS parameter in member INSTALLD in the Connector INSTALL library.
3. Restart the Connector and the Connector Gateway.

Enabling TLS Communication

The Transport Layer Security (TLS) protocol provides encryption, authentication and integrity to all applications.

This section describes the steps for enabling TLS 1.2 between IdentityIQ and Connector Gateway.

Implementing Secured Communication to Mainframe Agent

Secured communication to Mainframe Agent must be implemented using one of the following methods:

- **Mainframe Agent SSL**: Communication must be implemented on a port defined to Mainframe Agent as secured.

For more information, see “Implementing Mainframe Agent TLS”.

- **AT-TLS policy**: Communication must be implemented on a port defined to Mainframe as non-secured. The TLS processing is done by TCPIP and is transparent to Mainframe Agent.

  For more information, see “Implementing AT-TLS policy for Mainframe Agent communication”.

The secured communication is implemented using server authentication.

Common implementation procedure

1. A valid server certificate with its associated server private key must be defined. This certificate must be signed by a trusted Certificate Authority's (CA).
2. The server certificate and the CA certificate must be connected to a key ring.
3. The CA certificate must be exported to a file, transferred (using FTP with ASCII mode) to the client and installed there to be used for certificate verification by the TLS handshake process.

   Note: For testing purposes, a local CA can be defined for signing the server certificate.

Implementing Mainframe Agent TLS

For detailed information about implementing Mainframe Agent TLS, see “Setting up for SSL/TLS” chapter of *z/OS IBM Tivoli Directory Server Administration and Use for z/OS* IBM manual.

Note: Mainframe Agent must be granted with permission to access the key ring containing the Mainframe Agent certificate and the CA certificate.

Implementing AT-TLS policy for Mainframe Agent communication

For detailed information about implementing AT-TLS policy, see “Application Transparent Transport Layer Security data protection” chapter of *z/OS Communications Server IP Configuration Guide*.

The required policy attributes for AT-TLS policy are:

- Local Port Range – ports defined in Mainframe Agent as non-secured
- Direction = Inbound
- TLS Enabled = On
- TLS v1.1 = On
- TLS v1.2 = On
- Handshake Role = Server
- Client Authorization Type = PassThru
- Application Controlled = Off
- Secondary Map = Off
- The name of the certificate created for the secured communication and the name of the key ring to which the server certificate and the CA certificate are connected, should be specified.

  Note: TCPIP must be granted permission to access the key ring to which the Mainframe Agent certificate and the CA certificate are connected.

Sample file for AT-TLS policy

```
# RULE for LDAP GLDSRV
################################################
TTLSRule LDAP
{
```

```
  LocalAddr ALL
  RemoteAddr ALL
  LocalPortRange 389
  Direction Inbound
  Priority 255 # highest priority rule
  Userid GLDSRV
  TTLSGroupActionRef GrpAct_LDAP
  TTLSEnvironmentActionRef GrpEnv_LDAP
  TTLSConnectionActionRef GrpCon_LDAP
}

TTLSGroupAction GrpAct_LDAP
{
  TTLSEnabled On
  Trace 7
}

TTLSEnvironmentAction GrpEnv_LDAP
{
  Trace 7
  HandshakeRole Server
  EnvironmentUserInstance 0
  TTLSKeyringParmsRef PrmKeyRing_LDAP
  TTLSEnvironmentAdvancedParmsRef PrmEnvAdv_LDAP
}

TTLSEnvironmentAdvancedParms PrmEnvAdv_LDAP
{
  TLSv1.1 On
  TLSv1.2 On
  ClientAuthType PassThru
}

TTLSConnectionAction GrpCon_LDAP
{
  HandshakeRole Server
  TTLSCipherParmsRef PrmCipher_LDAP
  TTLSConnectionAdvancedParmsRef PrmConAdv_LDAP
  CtraceClearText Off
  Trace 7
}
TTLSConnectionAdvancedParms PrmConAdv_LDAP
{
  ApplicationControlled Off
  CertificateLabel GLDSRV
  SecondaryMap Off
}
TTLSCipherParms PrmCipher_LDAP
{
# supported cipher suites - we used a wide list, that should be decreased according
# to specific needs
V3CipherSuites      TLS_DH_DSS_WITH_DES_CBC_SHA
V3CipherSuites      TLS_DH_RSA_WITH_DES_CBC_SHA
V3CipherSuites      TLS_NULL_WITH_NULL_NULL
V3CipherSuites      TLS_RSA_WITH_NULL_MD5
V3CipherSuites      TLS_RSA_WITH_NULL_SHA
V3CipherSuites      TLS_RSA_EXPORT_WITH_RC4_40_MD5
V3CipherSuites      TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
```

```
V3CipherSuites      TLS_RSA_WITH_DES_CBC_SHA
V3CipherSuites      TLS_DHE_DSS_WITH_DES_CBC_SHA
V3CipherSuites      TLS_DHE_RSA_WITH_DES_CBC_SHA
V3CipherSuites       TLS_RSA_WITH_AES_256_CBC_SHA256
V3CipherSuites       TLS_RSA_WITH_AES_256_CBC_SHA
V3CipherSuites       TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
V3CipherSuites       TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_DHE_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_DHE_DSS_WITH_AES_128_CBC_SHA
V3CipherSuites       TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
V3CipherSuites       TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites       TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites       TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
}
TTLSKeyringParms PrmKeyRing_LDAP
{
  Keyring GLDRING

}
```

Enabling TLS between Mainframe Agent and Connector Gateway

1. Setup AT-TLS on Mainframe computer where Agent is installed. The truststore must be exported from Mainframe Agent and imported in Connector Gateway.
   Export server certificate and copy the exported **.cer** file to the Java client computer (IdentityIQ computer).
2. Import truststore certificate exported in the above step into truststore of Connector Gateway using the following command:
   `keytool -keystore cacerts -importcert -alias "`*`agentCert`*`" -file `*`agentFile`*
   where

Variables	Description
agentCert	It is an Alias to Mainframe generated certificate that is used to store imported certificate on truststore.

Variables	Description
agentFile	It is an Mainframe exported file to be used for TLS communication which consists Public/Private key combination to be used while encrypted communication.

Note: Ensure that the Agent file is in the truststore that is, cacerts lib and replace it with actual name.

3. Use the following command to set the **enableCGtoAgentTLS** parameter to **true** and **truststore** path.
   `java -jar ConnectorGateway.jar -enableCGtoAgentTLS "true" -trustStore "truststorePath"`

Enabling TLS between IdentityIQ and Connector Gateway

1. Generate keystore for Connector Gateway.

- **Configuring the Keystore**: Keystore file and keystore.jks, contains the Application Server's certificate, including its private key. The keystore file is protected with a password, initially **changeit**. If **keystore.jks** is not available it can be created by using the following command:

  `keytool -genkey -noprompt -trustcacerts -keyalg RSA -alias ${cert.alias} -dname "domainName" -keypass ${key.pass} -keystore ${keystore.file} -storepass "keystorePass"`

- **Creating certificate**: Create a self-signed certificate in a keystore of type JKS. This will create a certificate using algorithm stored in keystore and it will generate public and private key combination.

  `keytool -genkey -alias CGKey -keypass password -storepass password -keystore keystore.jks`

- **Exporting and importing the Certificate:** Use the Java keytool, as described in the following sections to export and import the certificate into Java clients.

  - Generate the *server.cer* certificate:
    a. `keytool -export -alias CGKey -storepass password -file server.cer -keystore keystore.jks`

       Note: At the prompt enter valid *keystorePassword*.

       The Java keytool outputs the certificate file to current directory.

  - To import *server.cer* certificate into truststore cacerts:
    Note: Provide your keystore password to import certificate. The Java default is 'changeit'.
    a. Locate the certificate file that you exported from Authentication Manager, and copy it to the target host.
    b. Import the certificate to the local *cacerts* keystore. Enter the following command:

       `keytool -import -v -trustcacerts -alias CGKey -file server.cer -keystore cacerts.jks -keypass password`

       The Java keytool displays a confirmation that the certificate was added to the keystore.

2. Set the **enableCGtoPE2TLS** parameter to true in the `init.xml` file as follows:
   `java -jar ConnectorGateway.jar -enableCGtoPE2TLS "true" -keyStore "keystorePath" -keyStorePass "keystorePassword" -keyPass "keyPassword"`
   where

Variables	Description
keystorePath	It is a keystore.jks location where it has been generated using keytool utility, specifically on the Connector Gateway machine.
keystorePassword	The keystore.jks file is protected with password and this password enables to access keystore.jks once authenticated.
keyPassword	Key password is for the private/secret key password, in case keystore password fails then keytool uses keypassword to recover keys.

3. Import the truststore generated in Step <$elemparanumonlyb on IdentityIQ computer.
4. Define a value for the JAVA_OPTS environment variable `-Djavax.net.ssl.trustStore` to the full path of truststore.

Settings for configuring Password Interceptor and Online Interceptor

Password Interception or Online Interceptor is not enabled by default. There are few steps that need to be performed in order to enable it on Gateway Connector and IdentityIQ.

Note: The value of hostName in ServiceDefinitions of SMListener and ResourceEvent are case sensitive.

To Enable Password Interception or Online Interceptor on IdentityIQ, add the following attributes in Service definition:

1. To open the ServiceDefinitions navigate to **IdentityIQ debug page => ServiceDefinition => SMListener** and add the attribute and hostname as follows:

- Add the following attribute in SMListener:

```
<Attributes>
  <Map>
    <entry key="applications" value="Application name"/>
    <entry key="retryInterval" value="5" />
  </Map>
</Attributes>
```

Note: To configure multiple applications for password interceptor, add multiple names in application value as follows:
`<entry key="applications" value="App1,App2"/>`

- Add the hostname as follows:

```
<ServiceDefinition created="1347280738108" hosts="hostName"
id="2c9095d439b02f0b0139b030973c00e9" modified="1347285339111"
name="SMListener">
```

2. To open the ServiceDefinitions navigate to **IdentityIQ debug page => ServiceDefinition => ResourceEvent** and add the hostname as follows:

```
<ServiceDefinition created="1347280738124" hosts="hostName"
id="2c9095d439b02f0b0139b030974c00ea" modified="1347285354945"
name="ResourceEvent">
```

In the preceding command lines, the value of *hostName* in ServiceDefinitions of SMListener and ResourceEvent must be the name/IP of host where IdentityIQ Server is running. User must define only one IdentityIQ host if there are multiple instances of IdentityIQ Servers running. This will start **SMListener** Service on one server where the interceptors sent from Agent would be processed.

Note: If the application that is configured to receive password changes was before upgrading IdentityIQ to version 6.0 then perform the following additional steps:

a. Navigate to the application definition page (application that is configured to receive delta changes).
b. Re-enter the password.
c. Save the application.
d. Perform a **Test Connection**.

When IdentityIQ starts it will start a thread per application for which password interception is enabled. This thread is responsible for receiving the delta changes from that application. If any change is made to the above attributes or if you enable or disable encryption for the application that is configured to receive password changes, the IdentityIQ server must be restarted for the changes to take effect.

Password Interceptor and Online Interception are supported for the following Gateway Connectors:

- **CA-ACF2**: To enable Password Interceptor or Online Interceptor on CA-ACF2, refer to the *SailPoint IdentityIQ Connector for CA-ACF2 Administration Guide*.
- **CA-Top Secret (TSS)**: To enable Password Interceptor or Online Interceptor on TSS, refer to the *SailPoint IdentityIQ Connector for CA-Top Secret Administration Guide*.
- **RACF**: To enable Password Interceptor or Online Interceptor on RACF, refer to the *SailPoint IdentityIQ Connector for RACF Administration Guide*.

Change Interception

Under default operation, when an account is modified on the remote system, change interception in IdentityIQ can automatically receive only the items/attributes that were changed and then merge that change into the existing IdentityIQ identity. These items/attributes were not processed through the aggregation procedures set up for the application (such as buildMap rule, customization rule and so on).

If the processes associated with aggregation are important for these changes, you can enable the complete aggregation of single accounts, groups, or both when a change is detected, so that these additional procedures are applied to the account when a change is detected. This can be achieved by specifying the following attributes in the ServiceDefinition for the ResourceEvent:

- **getEntireAccount**
- **getEntireGroup**

The above attributes will trigger the aggregation of the accounts, groups, or both for the object on which the change is detected and ensures that any new change on accounts or groups are processed through the aggregation procedures set up on IdentityIQ for the respective applications.

Note: The “getEntireAccount” and “getEntireGroup” attributes accept boolean values.

Following is an example, with the attribute values enabled:

```
<ServiceDefinition name="ResourceEvent">
  <Attributes>
    <Map>
      <entry key="correlateEntitlements" value="true"/>
```

```
        <entry key="getEntireAccount" value="true"/>
        <entry key="getEntireGroup" value="true"/>
        <entry key="noOptimizeReaggregation" value="true"/>
        <entry key="promoteAttributes" value="true"/>
      </Map>
    </Attributes>
  </ServiceDefinition>
```

Appendix A: Troubleshooting

This appendix describes problems that may be encountered while setting and configuring the Gateway Connectors and the resolutions for these problems.

1 - Third party monitoring tool (Port Monitoring software) usage and it's implications

Resolution: When using Port Monitoring software to validate the Connector Gateway connections, use one of the following methods to enable this service to work properly:

- **Using a port monitoring solution**: Ensure that the port monitoring software is located on a server other than the IdentityIQ and/or the Connector Gateway server. The proper method of using the software is to establish a connection in the same method (as follows) as used by IdentityIQ:

  Connect first on the defined port +1, then on the defined port 2471, (for example, the port defined in `init.xml` is 2470) as this will create a valid socket connection.
- **Checking the status of Connector Gateway**: This method involves the use of two telnet sessions in the same manner as described below:

  Connect first on the defined port +1, then on the defined port 2471, (for example, the port defined in `init.xml` is 2470) as this will create a valid socket connection.

  This should be tested to ensure that this does not interfere with active connections within your environment.
- **Writing a log parser**: Write a log parser to check for socket information within the logs. If the socket connections are not performing, that for Connector Gateway to determine the proper status, enable the DEBUG tracing. Following are the other methods to use log parser method:

  - Check the log output to see when the connection information was last updated (stale connection check), and if there are no updates from the Connector Gateway within time variable **X** (to be defined locally as it pertains to the environment), then Service. With this the Connector Gateway would be Online.

  - Write a log parser that will validate the socket connection pairs for the transactions that are valid and are initiated from IP address of IdentityIQ host only. When using this method, ensure that the connection is misdirected, unclean or invalid.

Ensure that other unwanted traffic does not interfere with the normal communications in your network by the use of Firewalls and Routing to isolate traffic to and from only trusted and intended source machines and services.

SailPoint recommends to test the above alternative methods in your environment using the proper Change Management methodology as specified below:

- Develop
- Test
- Promote to production after proper testing is completed

2 - During aggregation if user performs Test connection the Test connection transaction timeouts

During aggregation if user performs Test connection the Test connection transaction timeouts.

Resolution: During aggregation if user performs Test connection, the Test connection transaction timeouts as the Connector is busy in processing Aggregation. To resolve this issue, configure Multiple Transaction Servers on

Connector to perform more than one transaction at a time. For more information on the configuration changes, see the procedures documented in the respective *Connector Administration Guide*.

Note: You can set 2 or 3 multiple transaction servers to handle more concurrent operations depending on the load.

3 - Transactions taking more time than 10 minutes.

Transactions taking more time.

Resolution: If the transaction is expected to take more than 10 minutes, perform the following:

1. Set **smReadTimeout** parameter in Configuration => System Configuration as follows:
   `<entry key="smReadTimeout" value="10"/>`
   Timeout value is in minutes and can be defined to read message between IdentityIQ and the Connector. The default value is 10 minutes.

4 - Why Target Permissions are not getting aggregated in IdentityIQ?

Target Permissions are not getting aggregated in IdentityIQ.

Resolution: If Target Permissions are not getting aggregated in IdentityIQ, then:

- Verify if the application name defined in unstructured target is correct.
- Verify if the correlation rule is pasted (by removing the extra end lines if it is copied from document) correctly.

5 - Administrator or Password is invalid.

While doing Test Connection or performing any provisioning operations on RACF managed system, the transaction fails with the following error message:

```
Administrator or Password is invalid
```

Resolution: Ensure that the application configuration credentials are correct and application administrative user is enable or active on the managed system.

6 - POS attribute not found error is displayed on Agent.

When deleting the permissions the following POS attribute error is displayed on Agent:

```
Mandatory keyword POS missing in ACE ADDINFO
```

Resolution: Perform the following:

Adding POS keyword for ACF2 application

1. Navigate to application debug page and open ACF2 application for which the error is observed.
2. Add the POS keyword in **splAceAttributes** and **splTargetPermissionsInterestingKwds** as follows and save the page:
   `<entry key="POS" value="false"/>`

3. Perform the following change in IdentityIQ Application Schema to update IdentityIQ that the POS keyword is added:
    - Navigate to **Application ==> ACF2** application and modify any property (for example, description of any keyword).
    - Save the application.
4. Perform Test Connection to update the keyword list on Agent.
5. Run Target Aggregation and Certification. Now try deleting a permission.

7 - Special character(s) not working with Gateway Connectors.

Special character(s) not working with Gateway Connectors in passwords.

Resolution: If any special character(s) is not working with Gateway Connectors in passwords, perform the following:

1. Change the character set to IBM1047 for Mainframe based applications as follows:
   `<entry key="IBMcharacterSet" value="`**IBM1047**`"/>`
2. Set the value of the <`characterSet`> in `init.xml` file of Gateway Connectors to the same value as that of the characterSet in the above step:
   `<characterSet>`**IBM1047**`</characterSet>`
3. Restart Connector Gateway.

8 - Enable traces

Resolution:

- On IdentityIQ:
    a. Enable `log4j` tracing on sm package within the Application Server by adding the following line in the `log4j.properties` file:

       ```
       log4j.logger.sailpoint.connector.sm=debug
       ```
    b. Save and reload the `log4j.properties` file.
- On Connector Gateway:
    a. Edit `log4j.properties` file on Connector Gateway by setting the following line:

       ```
       log4j.rootLogger = ALL, commonLog
       ```
    b. Restart Connector Gateway.

9 - When DELAYED-DELETE is enabled on Agent and not on IdentityIQ an error message appears

When DELAYED-DELETE is enabled on Agent and not on IdentityIQ, the following error message is displayed:

```
PreScript Message: 2017/08/29 23:57:38 CTS3016I R DELAYED-DELETE option must be
active. 2017/08/29 23:57:38 CTS3016I R Invalid script return code ERROR. Assumed
FATAL(4) 2017/08/29 23:57:38 CTS3016I R SCRIPT CTSRQ.V400I.TS1.USER.CLIST(UNDFLNK3)
HAS FAILED 2017/08/29 23:57:38 CTS1433I R Pre-script failed with rc=FATAL ActualMsg:
PostScript Message:
```

Resolution: Enable **DELAYED-DELETE** option in application debug page.

10 – (*Only for CA-ACF2*) An error message appears when creating an account with lower case characters.

When a user tries to create an account in CA-ACF2 with lower case characters, the following error message appears:

```
Provisioning account failed for operation Create with error: Identity attribute
[USER_ID] was not found.
```

Resolution: Enter the USER_ID in upper case or add the following code in the application debug page:

1. Open the application in debug page and search for **PlanInitializerScript** tag.
2. Add the following code for Create account operation to convert the USER_ID entered in lower case to upper case.

```
{{    if (op == AccountRequest.Operation.Create)
      {
        String userId = req.getNativeIdentity();
        if (userId != null)
        {
            req.setNativeIdentity(userId.toUpperCase());
        }
      }
}}
```

Index