RSA Identity Governance and Lifecycle and RSA Authentication Manager Integration: Collections

Lab Exercise Guide

Before You Begin

Objectives

At the end of this lab exercise, you should be able to:

- Connect to the RSA Authentication Manager (AM) host operating system to perform AM-specific configurations
- Create a database user account and RSA AM administrative role
- Install the PostgreSQL driver on the RSA Identity Governance and Lifecycle (IGL) host and import the required root certificate
- Configure IGL with:
  - Attributes for collected user and token account data
  - Applications and data collectors to collect RSA AM user and token information
- Verify success and operation through:
  - Viewing collection results
  - Configuring reviews, reports, and dashboards using the collected RSA Authentication Manager user and token data

Resources

In order to successfully complete these exercises, you can make use of the following references:

- This lab guide
- Accessing RSA Virtual Labs PDF document (Available on the RSA University site ‘Content’ page)
- RSA Identity Governance and Lifecycle online documentation (Available on the RSA Link site)
Contents

BEFORE YOU BEGIN ...........................................................................................................1

OBJECTIVES .....................................................................................................................1

REQUIREMENTS .............................................................................................................1

LOGIN INFORMATION .................................................................................................1

PROBLEMS/QUESTIONS? .............................................................................................1

RESOURCES .................................................................................................................1

LAB GOALS AND METHODOLOGY ..............................................................................4

PREREQUISITES ..............................................................................................................4

LAB ENVIRONMENT .....................................................................................................4

JUMPHost.......................................................................................................................4

RSA IGL 7 Server ...........................................................................................................4

PRESTIGE.COM DC .......................................................................................................5

AM8_PRIMARY ..............................................................................................................5

RSAS.COM AM8-DC .....................................................................................................5

AM8-AMIS ....................................................................................................................5

COPYING TEXT FROM EXTERNAL SOURCES ...............................................................6

INTRODUCTION .............................................................................................................7

COMPONENTS .............................................................................................................7

RSA Identity Governance and Lifecycle (IGL) ...............................................................7

RSA Authentication Manager (AM) ............................................................................7

Authentication Manager Integration Services (AMIS) .............................................7

BUSINESS ISSUE .......................................................................................................7

RSA PRODUCT VERSIONS .........................................................................................8

ACRONYMS .................................................................................................................8

ARCHITECTURE ...........................................................................................................9

INTEGRATION OVERVIEW ...........................................................................................9

WORKSHEET ...............................................................................................................10

PREREQUISITES ..........................................................................................................11

AM AND AMIS INSTALL .............................................................................................11

EXERCISE 1: RSA AUTHENTICATION MANAGER CONFIGURATION ................................12

EXERCISE 1 TASKS .......................................................................................................12

Task 1-1: Enable SSH on AM Server ...........................................................................12

Task 1-2: Establish Connection to AM Command Line Interface ................................13

Task 1-3: Open Database connection through AM Firewall .....................................14

Task 1-4: Create a Read-Only Database User ..............................................................15

EXERCISE 2: IGL CONFIGURATION ........................................................................17

Database Dependencies ...............................................................................................17

EXERCISE 2 TASKS ......................................................................................................17

Task 2-1: 7.0+ Postgresql Driver Update ..................................................................18

Task 2-2: Import Root Certificate ..............................................................................20

EXERCISE 3: CONFIGURING IGL .............................................................................21

Identity and Account Data Collectors .........................................................................21
EXERCISE 3 TASK

Task 3-1: Add IGL Attributes to Collect AM Objects

Task 3-2: Create the IGL Application

Task 3-3: Create the IGL Tokens Application

Task 3-4: Create the IGL Token Expiry Collector

Task 3-5: Create the IGL Token Account Collector

Task 3-6: Create the IGL Token Entitlement Collector

EXERCISE 4: CREATING APPLICATIONS AND COLLECTORS

Task 4-1: Create the RSA Authentication Manager Application

Task 4-2: Create the RSA Authentication Manager Tokens Application

Task 4-3: Create the Available SecurID Tokens Application

Task 4-4: Create the Authentication Manager Account Collector

Task 4-5: Create the Authentication Manager Tokens Account Collector

Task 4-6: Create the Authentication Manager Token Entitlement Collector

NEXT STEPS: GOVERNANCE REVIEWS, REPORTS AND EXPERIMENTS

Review Tokenholders

Review Token Expiry

Privileged User Review

No Login Since... Report

Expiring Token Report

Token Summary Chart

Available vs Assigned Tokens Chart

Tokens Used vs Unused Chart

Token Types Chart

Dashboard

Troubleshooting

Feedback
Lab Goals and Methodology

The RSA Identity Governance Integration with RSA Authentication Manager On-Demand Learning (eLearning) presentation forms a part of the introduction and background to this virtual lab exercise. You should view the eLearning presentation and demonstration prior to beginning this exercise. This exercise can then build on that base knowledge and provide you with the practice of completing the IGL-Authentication Manager integration.

To complete this exercise, follow this guide in the order given. At the completion of the exercise, you are free to experiment with other aspects of the integration (additional reports or reviews, for example) to test your own understanding of the processes presented in this exercise.

Prerequisites

Before beginning this lab exercise, you should have completed the course RSA Identity Governance and Lifecycle Administration or have equivalent experience with the RSA Identity Governance and Lifecycle product. Prior experience or training with RSA Authentication Manager is also recommended.

Lab Environment

This diagram summarizes the preconfigured virtual environment in which you will be working:

JumpHost

The Jumphost serves as a remote desktop for accessing other machines in the environment. The desktop of the Jumphost has shortcuts for browsers IE and Chrome and utilities that you can use as needed for the exercises.

- **WinSCP** allows you to transfer files between the Jumphost and the other virtual machines in your lab environment.

RSA IGL 7 Server

The software for RSA Identity Governance and Lifecycle v7.0 is installed. This machine is joined to the PRESTIGE.COM domain and an initial user account collection from the PRESTIGE.COM Active Directory has been configured and performed.

- RSA IGL Application Credentials: AveksaAdmin/Aveksa123
- OS Credentials root/Av35ka
- OS: SUSE Linux 11 SP3
- Hostname: dvd-acm.prestige.com
- IP Address: 192.168.1.100
PRESTIGE.COM DC
This is a domain controller and DNS server for the PRESTIGE.COM domain with populated Active Directory user accounts.
- Credentials: Administrator/Aveksa123
- OS: Windows 2008 R2
- Hostname: WIN-4H66GL4CGK5.prestige.com
- IP Address: 192.168.1.105

am8_PRIMARY
This is an RSA Authentication Manager v8.2 Primary server.
- Credentials: Security Console: Admin/Password01$; Operations Console: ocadmin/Password01$; OS: rsaadmin/Password01$
- OS: SUSE Linux 11
- Hostname: am8-p.rsas.com
- IP Address [eth0]: 10.101.240.105
  IP Address [eth1]: 192.168.1.102

RSAS.COM am8-dc
This is a domain controller and DNS server for the RSAS.COM domain with populated Active Directory user accounts.
- Credentials: Administrator/Password01$
- OS: SUSE Linux 11
- Hostname: am-dc.rsas.com
- IP Address: 10.101.240.100

am8-AMIS
This is a Windows 2008 machine that hosts the RSA AMIS utility on Apache Tomcat.
- Credentials: Administrator/Password01$
- OS: Windows 2008 R2
- Hostname: am8-webserver.rsas.com
- IP Address: 10.101.240.103
### Copying Text from External Sources

Your environment allows you to copy small sections of text from external sources into your working environment. This is useful, for example, to copy in queries from the Lab Guide, copy URLs, passwords, etc.

To perform the Copy/Paste, follow these steps:

<table>
<thead>
<tr>
<th>Select a text block and copy to your clipboard (right-click &gt; <strong>Copy</strong> or <strong>CTRL+C</strong>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>At the top of your environment screen, select the clipboard icon</td>
</tr>
<tr>
<td>Paste your text into the Copy/paste area</td>
</tr>
<tr>
<td>You can then re-copy the text...</td>
</tr>
<tr>
<td>...and paste it to the desired location within your lab environment</td>
</tr>
</tbody>
</table>
Introduction

Components

Recently, RSA has brought the RSA Identity Governance and Lifecycle and RSA Authentication Manager under the brand name of ‘RSA SecurID Suite’. This also includes additions to the authentication capabilities of RSA Authentication Manager under the product brand name ‘RSA SecurID Access’.

This exercise refers to RSA Authentication Manager as a component but you may find industry-facing references to the RSA SecurID Access product in the future. The specific functions provided by RSA Authentication Manager are the same as they relate to the integration described in this exercise.

The following major components are used in this integration exercise:

RSA Identity Governance and Lifecycle (IGL)

This comprehensive, business-driven identity management and governance platform is intended to help solve both Identity and Access Management needs. The platform includes a user-friendly web interface, which reliably and efficiently allows a business to automate and manage key activities across the identity and access management lifecycle.

RSA Authentication Manager (AM)

RSA Authentication Manager protects resources on your network by requiring users to authenticate using multifactor authentication methods like RSA SecurID tokens, Risk-based authentication (RBA), On-demand authentication (ODA) and mobile device Authentication Tokencodes.

Authentication Manager Integration Services (AMIS)

Authentication Manager Integration Services provides a powerful service tier built to extend critical AM administration and user self-service capabilities. This helps AM to meet a variety of integration needs. AMIS and its ancillary services - Storage and Authentication- are HTTP REST-based web services that provide a simplified API for the automation of administration and end-user functions for management and distribution of all types of RSA SecurID tokens.

Business Issue

This integration can be used to support the following business use cases:

- Offers a unified view of RSA Authentication Manager accounts, groups, roles and RSA SecurID token assignments
- Provides a way to review RSA Authentication Manager token assignments to ensure RSA SecurID tokens are assigned to the right individuals
- Supports RSA Authentication manager and RSA SecurID token AM Reports and Charts that can be used in creating Dashboards for security and business managers
RSA Product Versions

The following RSA product versions were tested in this qualification:

<table>
<thead>
<tr>
<th>RSA Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGL</td>
<td>7.0.0</td>
</tr>
<tr>
<td>AM</td>
<td>8.2.1</td>
</tr>
<tr>
<td>AMIS</td>
<td>8.2 Update – 10-07-2016</td>
</tr>
</tbody>
</table>

Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGL</td>
<td>RSA Identity Governance and Lifecycle (Previously branded as RSA Via Lifecycle and Governance, RSA Identity Management and Governance[IMG], and originating as a product named “Aveksa” before acquisition by RSA)</td>
</tr>
<tr>
<td>ACM</td>
<td>Access Certification Manager (The certification subcomponent of IGL)</td>
</tr>
<tr>
<td>AFX</td>
<td>Access Fulfillment Express (The access request fulfillment subcomponent of IGL)</td>
</tr>
<tr>
<td>AM</td>
<td>Authentication Manager</td>
</tr>
<tr>
<td>AMIS</td>
<td>Authentication Manager Integration Services</td>
</tr>
<tr>
<td>AM Prime</td>
<td>The set of accessory software and utilities that includes AMIS</td>
</tr>
</tbody>
</table>
Architecture

Integration Overview

- Identities are collected from Customer Identity store like HR database, Active Directory etc.
- AM accounts are collected from AM’s PostgreSQL database.
- AM entitlements are collected from AM’s PostgreSQL database.
- AFX provisions AM accounts and entitlements via AMIS REST APIs.
## Worksheet

The following information reflects the on-demand lab environment. In a customer environment, you would want to gather this information to help you as you connect to and deploy components.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM IP address</td>
<td>10.101.240.105</td>
</tr>
<tr>
<td>AM Linux console (OS) userid</td>
<td>rsaadmin</td>
</tr>
<tr>
<td>AM Linux console (OS) password</td>
<td>Password01$</td>
</tr>
<tr>
<td>RSA Security Console UserID</td>
<td>admin</td>
</tr>
<tr>
<td>RSA Security Console Password</td>
<td>Password01$</td>
</tr>
<tr>
<td>RSA Operations Console Userid</td>
<td>ocaadmin</td>
</tr>
<tr>
<td>RSA Operations Console Password</td>
<td>Password01$</td>
</tr>
<tr>
<td>IGL Server IP address</td>
<td>192.168.1.100</td>
</tr>
<tr>
<td>IGL Linux console userid</td>
<td>root</td>
</tr>
<tr>
<td>IGL Linux console password</td>
<td>Av3k5a</td>
</tr>
<tr>
<td>AMIS IP address</td>
<td>10.101.240.103</td>
</tr>
<tr>
<td>AMIS SMB Share</td>
<td>rsa maps to /opt/rsa</td>
</tr>
</tbody>
</table>
**Prerequisites**

**AM and AMIS Install**

In your lab environment, RSA Authentication Manager and Authentication Manager Integrated Services (AMIS) have been installed and initially configured.

In a real-life situation, these components would need to be installed before starting the integration process. For instructions regarding installation and configuration of AM and AMIS, please refer their respective Installation & Configuration guides.

The following have been created/configured in RSA Authentication Manager:

- **Identity Source:** “Company AD” (mapped to Active Directory on the PRESTIGE.COM Domain Controller)
- **Security Domains:** (corresponding to groups within the Company AD Identity Source)
  - Customer Support
  - Development
  - Finance
  - Human Resources
  - Information Technology
  - Manufacturing
  - Marketing
  - Sales
  
  One additional “Bind-Accounts” Security Domain has also been created for use with AMIS bind accounts.

- **An “amis-bind” administrative service account has been created.**
  
  username: **amis-bind**
  password: **Password01$**

**Note:** Not all of the AMIS configurations or RSA Authentication Manager structure is used in this particular lab exercise. This environment supports multiple on-demand lab exercises.
Exercise 1: RSA Authentication Manager Configuration

In typical operation through the RSA Authentication Manager Security Console, administrative roles can be defined granularly to control the level of access to database objects – users, authenticators, etc. Direct access to the data in the AM internal database is limited to a read-only user account that is not subject to the delegated administration roles specified in Authentication Manager. You must control the information retrieved from the database with the application you create to perform the queries. Data that is encrypted in the database is not unencrypted when it is retrieved.

To secure connections to the database using the read-only user account, you must use SSL/TLS to connect to the database. Connections to the database that do not use SSL/TLS will not be allowed and will be rejected. If you want to use an SQL client utility to connect to the database, refer to your SQL client documentation to configure it to use SSL/TLS.

Exercise 1 Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK 1-1: Enable SSH on AM Server</td>
<td>13</td>
</tr>
<tr>
<td><em>Enable access to the RSA Authentication Manager host operating system to allow running command-line utilities.</em></td>
<td></td>
</tr>
<tr>
<td>TASK 1-2: Establish connection to AM Command Line Interface</td>
<td>14</td>
</tr>
<tr>
<td><em>Connect to the RSA Authentication Manager host operating system using PuTTY utility.</em></td>
<td></td>
</tr>
<tr>
<td>TASK 1-3: Open database connection through AM firewall</td>
<td>15</td>
</tr>
<tr>
<td><em>Open the RSA Authentication Manager firewall to allow a connection from the IGL server to the Authentication Manager database.</em></td>
<td></td>
</tr>
<tr>
<td>TASK 1-4: Create a read-only database user</td>
<td>16</td>
</tr>
<tr>
<td><em>Use the rsautil ‘manage-readonly-dbusers’ function to create a read-only database user. This user will be configured in the IGL collector to connect to the RSA Authentication Manager database.</em></td>
<td></td>
</tr>
</tbody>
</table>
Task 1-1: Enable SSH on AM Server

Some configurations must be made at the AM operating system level to allow IGL to connect to AM. In order to access the AM host appliance using SSH, you must explicitly enable the SSH connection. It is disabled by default.

Using a browser on your Jumphost machine, log in to the AM Operations Console by the shortcut on the Jumphost desktop or the URL:

https://am8-p.rsas.com/oc

- Login with username: ocadmin
- Password: Password01$

From the RSA Operations Console:

- Go to Administration > Operating System Access
- Enable SSH by checking the Interface eth0 and Interface eth1 checkboxes
  [Eth0 connects to the RSAS domain network; eth1 to the PRESTIGE domain network. We want to allow access to the OS from either network.]
- Click Save
- Close the browser
Task 1-2: Establish connection to AM Command Line Interface

With operating system access enabled, we need to access the command line interface on the AM host. To do this, we will use “PuTTY”.

From Windows desktop:
- Go to Start > All Programs > PuTTY > PuTTY
- At PuTTY configuration:
  - Hostname: 10.101.240.105 (or am8-p.rsas.com)
  - Port: 22
  - Protocol: SSH
- Click Open

If prompted by a PuTTY Security Alert:
- Click Yes to “trust this host”

Upon successful connection:
- Login using the Linux console credentials:
  - rsaadmin / Password01$
Task 1-3: Open Database connection through AM Firewall

To run the Reporting, you must open up a database connection through the imbedded firewall on the AM8 Appliance:

Within the same PuTTY session, execute:

- `cd /opt/rsa/am/config`
- `./config.sh PostgresDBInstance:allowDBPort ThruFirewall`

**Note:** The step below is described for information only. If you want to undo the connection or stop using the reporting tool, you can turn on the firewall again with the following command:

(from within the `/opt/rsa/am/config` directory)

- `./config.sh PostgresDBInstance:blockDBPort ThruFirewall`
Task 1-4: Create a Read-Only Database User

To query the internal database, you must create a read-only user with the Manage Read-only Database Users utility command `manage-readonly-dbusers`. This user has read-only access to the tables described in the RSA Authentication Manager Database Public Schema. You can only access the database from the client machine(s) whose IP address match those you specify when you create the read-only database user.

The command syntax is: 
```
rsautil manage-readonly-dbusers -a create -o <OC_admin_username> -u <readonly_database_user_name> -i <IP_address_of_client_machine> -n <IP_mask>
```

<table>
<thead>
<tr>
<th>WHERE</th>
<th>IS</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;OC_admin_username&gt;</td>
<td>Operations Console administrator</td>
</tr>
<tr>
<td>&lt;readonly_database_user_name&gt;</td>
<td>User name specified for the read-only user. <strong>Note:</strong> The user name should remain in lowercase. Our tests have determined that the PostgreSQL database does NOT handle mixed case user accounts.</td>
</tr>
<tr>
<td>&lt;IP_address_of_client_machine&gt;</td>
<td>IP address of the users’ client machine (the IGL Server)</td>
</tr>
<tr>
<td>&lt;IP_mask&gt;</td>
<td>If you want connectivity to be limited to a single client machine, specify the exact IP mask with the IP address. If an IP mask is not specified, the default is 255.255.255.255, which will allow connectivity to only the client IP address that was specified in the -i argument. To allow connections from any host with the IP address range 192.168.1.x, set the IP address (-i) argument to 192.168.93.0 and the IP mask (-n) argument to 255.255.255.0.</td>
</tr>
</tbody>
</table>

In our example, we will create a read only user called am8db, password: Password01$

We need to know:  
• The IP address of the IGL server and what netmask to use.  
• The RSA Operations Console userid and password.

From the PuTTY session, execute:

- `cd /opt/rsa/am/utils`
- `./rsautil manage-readonly-dbusers -a create -o ocadmin -u am8db -i 192.168.1.100 -n 255.255.255.0`
  - Operations Console password: Password01$
  - Create and confirm read-only database user password: Password01$

If successful, a trusted root SSL CA certificate is generated.

**Note:** there is usually a delay after the ‘create’ action completes and your CLI prompt returns.
Exercise 2: IGL Configuration

Database Dependencies

We need to Download and Install the PostgreSQL Driver to use IGL collectors for Authentication Manager. Adding the Authentication Manager root certificate to the IGL Java keystore allows trusted communication between RSA AM and IGL.

Exercise 2 Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK 2-1: 7.0+ Postgresql Driver Update</td>
<td>18</td>
</tr>
<tr>
<td>Update the Postgresql driver in IGL from the Postgresql.org website and use the customizeACM utility to add the driver to IGL.</td>
<td></td>
</tr>
<tr>
<td>TASK 2-2: Import Root Certificate</td>
<td>20</td>
</tr>
<tr>
<td>Import the RSA Authentication Manager SSL root certificate into the IGL keystore.</td>
<td></td>
</tr>
</tbody>
</table>
Task 2-1: 7.0+ Postgresql Driver Update

With the introduction of WildFly as our application server you must add the driver to the ear file or else it will be removed upon the next IGL service restart. Each Wildfly restart will extract the current ear file which is why the driver must be added to it for it to persist.

- Download the driver at the following location:
  
  Download [postgresql-9.4-1206-jdbc41.jar](http://jdbc.postgresql.org/download.html) for this lab environment.

  Other Postgresql drivers may be more appropriate for a particular customer environment or Java version that you may be working with.

- Copy `postgresql-9.4-1206-jdbc41.jar` to the RSA IGL Server
  
  (Use WinSCP for the file transfer)

  - Host: **192.168.1.100**
  - Login: **oracle**
  - Password: **secret**

- Location: **/tmp**

- SSH to the IGL Server

  - IP: **192.168.1.100**
  - Login: **oracle**
    
    (If already logged in as Root: **su** to **oracle**)  
  - Password: **secret**

- Verify IGL is running:

  `service aveksa_server status`

Continued...
### Task 2-1: 7.0+ Postgresql Driver Update, continued

- Change directory:
  
  ```
  cd /home/oracle/deploy
  ```

- Run `customizeACM.sh` script to extract .ear file, specifying the location of the .ear file that you want to modify:

  ```
  ./customizeACM.sh -c
  ```

*Note: If you do not specify the path to the .ear file, the script prompts you to use the currently deployed .ear file. If you want to use the currently deployed .ear, enter `yes`. If you do not want to use the currently deployed .ear, enter `no`.

For the purposes of this exercise, enter `yes` to use the most recent archived ear file.

- Add driver to ear file

  ```
  cp /tmp/postgresql-9.4-1206-jdbc41.jar /tmp/customizeACM/aveksa.war/WEB-INF/LocalAgent/common/lib/
  ```

- Change ownership of the PostgreSQL driver

  ```
  chown oracle:oinstall /tmp/customizeACM/aveksa.war/WEB-INF/LocalAgent/common/lib/postgresql-9.4-1206-jdbc41.jar
  ```

- Rebuild the ear file (from within `/home/oracle/deploy`):

  ```
  ./customizeACM.sh -d
  ```

  The script performs the following tasks:

  1. Archives the new .ear file to the following location, appending a time and date stamp to the name: `/home/oracle/archive`.
  2. Deploys the new customized .ear file. (This takes some time to complete.)
Task 2.2: Import Root Certificate

To connect to the PostgreSQL database we must have the root certificate in Java’s keystore.

Use WinSCP to:

- Copy the Root CA cert RSAAMTrustedRootSSLCA.crt from the AM Server /opt/rsa/am/utils/ directory to your local PC.
- Then copy the Root CA cert to the IGL Server /home/oracle directory.

Use PuTTY to:

- SSH to the IGL Server
- Login or su to root (pwd: Av3k5a)
- Change directory: cd /usr/lib64/jvm/java-1.7.0-openjdk-1.7.0/jre/lib/security/

- Import AM 8 Root CA to the cacerts keystore:
  
  keytool -keystore cacerts -importcert -alias AM8RootCA -file /home/oracle/RSAAMTrustedRootSSLCA.crt

  Keystore Password: changeit

- When prompted to trust the certificate, enter yes

- Restart the IGL Server:
  
  service aveksa_server restart
Exercise 3: Configuring IGL

Identity and Account Data Collectors
Identity and Account data collectors extract data that defines users in an organization and their related account information. Collections are performed from target sources such as Active Directory, Databases, or data files. Collection is used after provisioning to confirm an action has been taken (adding an account, enabling a token account, etc.)

Identity and account collections for base information has already been configured in RSA IGL for this lab environment. These collectors collect user data from Active Directory and an “HR Data” file. You can view the configured Identity and Account collectors under the Collectors menu in RSA IGL.

“HR Identity Collector”, “Active Directory IDC”, and “Active Directory ADC” collectors collect and unify identity and account data. The collected user accounts can be viewed under the RSA IGL Users menu.

Active Directory users are also mapped in an RSA Authentication Manager Identity Source and a subset of these users already have RSA SecurID tokens assigned. You can view the RSA AM users in the AM Security Console under the Identity menu and viewing the user list for the Company AD Identity Source (these are the same Active Directory accounts that are collected by RSA IGL) as well as users added directly to the RSA AM Internal Database.

Custom Attributes
Custom attributes extend the database schema to add additional attributes for collections. These custom attributes should be carefully thought out for an implementation as they are global and cannot be removed once they have been added. The tables on the following pages list examples of custom attributes to collect from RSA AM to support the exercises in this guide.

When you are working toward implementation for a Production environment, ensure that you test your custom attributes in a DEV or UAT environment. A full database backup is also highly recommended as a best practice so that you can restore a previous configuration, if needed.

Extending the IGL Schema for Authentication Manager
The schema changes below are optional and depend on what Authentication Manager data will be collected and managed within IGL. Specific attributes are suggested for the purposes of this exercise, but you may wish to experiment with other attributes and modify queries accordingly before collecting accounts.

In a real-life environment, you would probably try to economize the number of custom attribute added to a system. For example, using only one custom “Last Login” attribute to collect the last login date for both user accounts and token accounts. We have purposely duplicated some of the custom attributes to help separate the collected User and Token information for the sake of clarity. And to make it easier to understand what is being collected for each type of account collector.

Account Attributes
Used for Authentication Manager Application – User Account Collection:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Data Type</th>
<th>Database ID</th>
<th>Data Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LastUserLogin</td>
<td>Date</td>
<td>Choose available ‘CAD’ ID</td>
<td>Collected</td>
<td>Date/Time of last login</td>
</tr>
<tr>
<td>UserDisabled</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>True if account is disabled</td>
</tr>
<tr>
<td>UserLocked</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>True if account is locked</td>
</tr>
<tr>
<td>AccountFullName</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>AM Account’s First &amp; Last name</td>
</tr>
<tr>
<td>AccountType</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>AM User Account’s ‘Is Privileged’ flag (Privileged if an admin or User if user with no admin privileges).</td>
</tr>
</tbody>
</table>
Account Attributes, continued

Used for Authentication Manager Tokens Application – Token Account Collection:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Data Type</th>
<th>Database ID</th>
<th>Data Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AssignedUser</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>Token assignee’s user name</td>
</tr>
<tr>
<td>LastTokenLogin</td>
<td>Date</td>
<td>Choose available ‘CAD’ ID</td>
<td>Collected</td>
<td>Date/Time of last token login</td>
</tr>
<tr>
<td>LostStatus</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>Token Enabled/Disabled/Next Tokencode req’d</td>
</tr>
<tr>
<td>TokenExpires</td>
<td>Date</td>
<td>Choose available ‘CAD’ ID</td>
<td>Collected</td>
<td>Token expiration date</td>
</tr>
<tr>
<td>TokenType</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>Type (model) of token – Hardware; Software</td>
</tr>
<tr>
<td>TokenDisabled</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>Token enabled status</td>
</tr>
</tbody>
</table>

Although not used in these exercises, an organization might also want to collect:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Data Type</th>
<th>Database ID</th>
<th>Data Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replacement Status</td>
<td>String</td>
<td></td>
<td>Collected</td>
<td>Replacement token status for software tokens</td>
</tr>
<tr>
<td>Replacement Token Serial</td>
<td>String</td>
<td></td>
<td>Collected</td>
<td>Replacement token serial number for software token</td>
</tr>
</tbody>
</table>

Group Attributes

Used when managing Authentication Manager groups within RSA IGL or user moves to Groups:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Data Type</th>
<th>Database ID</th>
<th>Data Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM_GUID</td>
<td>String</td>
<td>Choose available ‘CAS’ ID</td>
<td>Collected</td>
<td>Used for AM groups GUID</td>
</tr>
</tbody>
</table>

Exercise 3 Task

Task | Page
---|---
TASK 3-1: Add IGL Attributes to Collect AM Objects | 23

Add custom attributes to RSA Identity Governance and Lifecycle to be used when collecting from RSA Authentication Manager.
Task 3-1: Add IGL Attributes to Collect AM Objects

- Login to IGL as **AveksaAdmin**; password **Aveksa123**
- Navigate to **Admin > Attributes**
- Select the **Account** tab and click **Edit**
- Select **Add Attribute**

Add a new attribute by entering a **Name**, selecting a **Data Type**, and selecting an available **Database ID** (‘CASx’, ‘CADx’, etc – depending on the Data Type) **Data Source** will be “Collected” for your attributes.

- At a minimum, add the following attributes for User, Token and Group collection:
  
  [Screen shown at right also has separators added to separate attributes for Users and Tokens]

  **Under Account tab:**
  
  **[AM User Attributes]**
  - LastUserLogin (Date data type)
  - UserDisabled
  - UserLocked
  - AccountFullName
  - AccountType
  
  **[AM Token Attributes]**
  - AssignedUser
  - LastTokenLogin (Date data type)
  - LostStatus
  - TokenExpires (Date data type)
  - TokenType
  - TokenDisabled

  **Under Group tab:**
  - AM_GUID

- Click **OK** to save the attributes
Exercise 4: Creating Applications and Collectors

The Authentication Manager Applications will have the collectors needed to bring user and token account data into RSA Identity Governance.

Exercise 4 Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK 4-1: Create the RSA Authentication Manager Application</td>
<td>25</td>
</tr>
<tr>
<td>Task 4-1: Create the RSA Authentication Manager Application</td>
<td>25</td>
</tr>
<tr>
<td>Application for user collection from RSA Authentication Manager.</td>
<td></td>
</tr>
<tr>
<td>TASK 4-2: Create the RSA Authentication Manager Tokens Application</td>
<td>26</td>
</tr>
<tr>
<td>TASK 4-2: Create the RSA Authentication Manager Tokens Application</td>
<td>26</td>
</tr>
<tr>
<td>Application for user-assigned RSA SecurID tokens collection from RSA Authentication Manager.</td>
<td></td>
</tr>
<tr>
<td>TASK 4-3: Create the Available RSA SecurID Tokens Application</td>
<td>26</td>
</tr>
<tr>
<td>TASK 4-3: Create the Available RSA SecurID Tokens Application</td>
<td>26</td>
</tr>
<tr>
<td>Application for unassigned (available) RSA SecurID tokens collection from RSA Authentication Manager.</td>
<td></td>
</tr>
<tr>
<td>TASK 4-4: Create the Authentication Manager Account Collector</td>
<td>28</td>
</tr>
<tr>
<td>TASK 4-4: Create the Authentication Manager Account Collector</td>
<td>28</td>
</tr>
<tr>
<td>Collects RSA Authentication Manager user accounts.</td>
<td></td>
</tr>
<tr>
<td>TASK 4-5: Create the Authentication Manager Tokens Account Collector</td>
<td>32</td>
</tr>
<tr>
<td>TASK 4-5: Create the Authentication Manager Tokens Account Collector</td>
<td>32</td>
</tr>
<tr>
<td>Collects RSA SecurID token data with user assignments from RSA Authentication Manager.</td>
<td></td>
</tr>
<tr>
<td>TASK 4-6: Create the Authentication Manager Token Entitlement Collector</td>
<td>37</td>
</tr>
<tr>
<td>TASK 4-6: Create the Authentication Manager Token Entitlement Collector</td>
<td>37</td>
</tr>
<tr>
<td>Collects RSA SecurID token data as entitlements that can be used for user account provisioning.</td>
<td></td>
</tr>
</tbody>
</table>

Tip: You can find text files with the Collector queries used in this Lab Guide in the Class Files\Queries\COLLECTOR_QUERIES folder on the Jumphost desktop.
Task 4-1: Create the RSA Authentication Manager Application

<table>
<thead>
<tr>
<th>Login to IGL as AveksaAdmin</th>
<th>Create Authentication Manager Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Navigate to Resources &gt; Applications</td>
<td>□ Select Create Application</td>
</tr>
<tr>
<td>□ Select Other Application</td>
<td>□ Click Next</td>
</tr>
</tbody>
</table>

□ Enter Application Information:
  - Application Raw Name: Authentication Manager
  - Application Name: Authentication Manager
  - Description: RSA Authentication Manager for Multi-factor Authentication.
  - Long Description: RSA Authentication Manager provides multi-factor authentication and capabilities to manage security tokens, users, agents, and resources across physical sites. (this long description is optional)
  - Allow Account Disabling: Yes
  - Allow Account Locking: Yes
  - Leave everything else as default.
□ Click Finish
## Task 4-2: Create the RSA Authentication Manager Tokens Application

<table>
<thead>
<tr>
<th>Login to IGL as AveksaAdmin</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Authentication Manager Application</td>
<td></td>
</tr>
<tr>
<td>□ Navigate to Resources &gt; Applications</td>
<td></td>
</tr>
<tr>
<td>□ Select Create Application</td>
<td></td>
</tr>
<tr>
<td>□ Choose Other Application</td>
<td></td>
</tr>
<tr>
<td>□ Click Next</td>
<td></td>
</tr>
<tr>
<td>□ Enter Application Information:</td>
<td></td>
</tr>
<tr>
<td>▪ Application Raw Name: <strong>Authentication Manager Tokens</strong></td>
<td></td>
</tr>
<tr>
<td>▪ Application Name: <strong>Authentication Manager Tokens</strong></td>
<td></td>
</tr>
<tr>
<td>▪ Allow Account Disabling: <strong>Yes</strong></td>
<td></td>
</tr>
<tr>
<td>▪ Allow Account Locking: <strong>No</strong></td>
<td></td>
</tr>
<tr>
<td>▪ Leave everything else as default.</td>
<td></td>
</tr>
<tr>
<td>□ Click Finish</td>
<td></td>
</tr>
</tbody>
</table>
### Task 4-3: Create the Available SecurID Tokens Application

<table>
<thead>
<tr>
<th>Login to IGL as AveksaAdmin</th>
<th>Create Authentication Manager Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Navigate to Resources &gt; Applications</td>
<td></td>
</tr>
<tr>
<td>□ Select Create Application</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>□ Choose Other Application</th>
<th>□ Click Next</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>□ Enter Application Information:</th>
</tr>
</thead>
<tbody>
<tr>
<td>▪ Application Raw Name: <strong>Available SecurID Tokens</strong></td>
</tr>
<tr>
<td>▪ Application Name: <strong>Available SecurID Tokens</strong></td>
</tr>
<tr>
<td>▪ Allow Account Disabling: <strong>No</strong></td>
</tr>
<tr>
<td>▪ Allow Account Locking: <strong>No</strong></td>
</tr>
<tr>
<td>▪ Leave everything else as default.</td>
</tr>
<tr>
<td>□ Click Finish</td>
</tr>
</tbody>
</table>

---

**Create Application**

- **Application Raw Name**: Available SecurID Tokens
- **Application Name**: Available SecurID Tokens
- **Allow Account Disabling**: No
- **Allow Account Locking**: No
- Leave everything else as default.

**Click Finish**
Task 4-4: Create the Authentication Manager Account Collector

Login to IGL as AveksaAdmin

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>Go to Authentication Manager Application</td>
</tr>
<tr>
<td>☐</td>
<td>Resources &gt; Applications</td>
</tr>
<tr>
<td>☐</td>
<td>Select Authentication Manager</td>
</tr>
<tr>
<td>☐</td>
<td>Click Collectors Tab</td>
</tr>
<tr>
<td>☐</td>
<td>Click Create Account Collector</td>
</tr>
</tbody>
</table>

Configure Collector Description:
- Collector Name: *Authentication Manager ADC*
- Description: *RSA Authentication Manager User Account Collector*
- Data Source Type: Database
- Agent: AveksaAgent
- Status: Active
- Copy from: leave blank
- Schedule: Leave as No

Configure Database Connection:
- DB Type: PostgreSQL
- Driver Class: `org.postgresql.Driver`

**NOTE:** There is a defect in IGL v 7.x that duplicates the jdbc prefix in the default URL. Remove the duplicated jdbc prefix

- User Name: am8db
- Password: Password01$
Task 4-4: Create the Authentication Manager Account Collector, continued

- Select types of account data to collect:
  - Check Accounts, User Account Mappings, and Groups
    (Do not check Sub Groups and Data is not case sensitive)
  - Click Next

- Configure Mapping for account attributes:
  - Accounts Data Query:
    *Remember that text files with the queries used in this Lab Guide are in the Class Files\Queries folder on the Jumphost desktop*

```sql
SELECT prin_data.loginuid as account,
      prin_login.login_date as login_date,
      concat(prin.first_name,' ',prin.last_name) as full_name,
      CASE WHEN prin_data.lockout_flag='true' THEN 1 ELSE 0 END as account_locked,
      CASE WHEN prin_data.enable_flag='true' THEN 0 ELSE 1 END as account_status,
      CASE WHEN prin_data.administrator_flag='true' THEN 'Privileged' ELSE 'User' END as account_type
FROM ims_principal prin
FULL JOIN IMS_Principal_data prin_data on prin.loginuid=prin_data.loginuid
JOIN IMS_Security_domain sec_dom on Prin_data.owner_id=sec_dom.id
JOIN ims_principal_login_date prin_login on prin_data.id=prin_login.principal_id
```

- Account ID/Name: account
- AccountFullName: full_name
- AccountType: account_type
- LastUserLogin: login_date
- UserDisabled: account_status
- UserLocked: account_locked

Continued...

Note that we are using our custom attributes for Last Login, Disabled, Locked, etc. rather than the built-in attributes. Either default or custom attribute types could be used to collect values.
Task 4-4: Create the Authentication Manager Account Collector, continued

- Mapping for User Account attributes:
  - User Account Mappings Data Query:
    ```sql
    SELECT DISTINCT(loginuid), loginuid as account
    FROM rsa_rep.ims_principal_data
    ```
  - User ID: loginuid
  - Account ID/Name: account

- Mapping for group attributes -- Group Data:
  - Groups Data Query:
    ```sql
    SELECT DISTINCT(name) as group, description, 
    CONCAT('ims.',id) as guid
    FROM rsa_rep.ims_group_data
    WHERE identity_src_id = '000000000000000000001000d0011000'
    ```
  - Group ID/Name: group
  - AM GUID: guid
  - (others remain blank)

- Mapping for Account Membership Data:
  - Account Membership Query:
    ```sql
    SELECT act.loginuid as account, grp.name as group
    FROM rsa_rep.ims_principal_data act
    JOIN rsa_rep.ims_principal_group mem ON
    act.id=mem.principal_id
    JOIN rsa_rep.ims_group_data grp ON
    mem.group_id=grp.id
    WHERE grp.identity_src_id = '000000000000000000001000d0011000'
    ```
  - Click Next

  Continued...
Task 4-4: Create the Authentication Manager Account Collector, continued

- **Edit User Resolution Rules:**
  - Click **Add More…**
  - Target Collector: **Users**
  - User Attribute: **User Id**

- **Edit Member Account Resolution Rules:**
  - Click **Add More…**
  - Target Collector: **Authentication Manager ADC**
  - Account Attribute: **Account Name**
  - Click **Finish**

Test the Collector by selecting the collector you just created: **Authentication Manager ADC**

- Click **Test**
- Upon successful test click **Cancel** – A successful test should display an XML-format output
  (If unsuccessful, return to check your query entries and/or perform other troubleshooting as needed)

Collect Account:

- Click **Collect Accounts** (Ignore Circuit Breaker)
- Click **OK** to start collection.

Check status of collection:

- Click **Collection History** Tab
- Click the **Run** number for details

Once status is “Completed”:

- Click breadcrumb on top **Application: Authentication Manager** then click the **Accounts** tab to see what was collected
### Task 4-5: Create the Authentication Manager Tokens Account Collector

**Login to IGL as AveksaAdmin**

- Go to Authentication Manager Application: **Resources > Applications**
  - Click **Authentication Manager Tokens**

- **Click Collectors Tab**
- **Click Create Account Collector**

- **Configure a Collector Description:**
  - **Collector Name:** Authentication Manager Tokens ADC
  - **Description:** Authentication Manager Tokens Account Collector
  - **Data Source Type:** Database
  - **Agent:** AveksaAgent
  - **Status:** Active
  - **Copy from:** leave blank
  - **Schedule:** Leave as No

- **Database Connection:**
  - **DB Type:** PostgreSQL
  - **Driver Class:** org.postgresql.Driver
  - **URL:** jdbc:postgresql://am8-p.rsas.com:7050/db?ssl=true
  - **User Name:** am8db
  - **Password:** Password01$

- **Test the connection**

- **Select types of account data to collect**
  - Check Accounts and User Account Mappings

---

*Continued...*
Task 4-5: Create the Authentication Manager Tokens Account Collector, continued

- Mapping for account attributes:
  - Accounts Data Query:
    
    ```sql
    SELECT serial_number as account, last_login_date as LastLogin,
    CASE WHEN is_enabled = 'false' THEN 'YES' ELSE 'no'
    END as disabled,
    token_shutdown_date as deathdate,
    CASE WHEN token_type = '0' THEN 'Std Card' WHEN
    token_type = '1' THEN 'PIN Pad' WHEN token_type =
    '2' THEN 'SID 700' WHEN token_type = '4' THEN
    device.family_key WHEN token_type = '9' THEN 'SID
    800' END AS tokenModel,
    CASE WHEN is_token_lost = 'true' THEN 'LOST' END as
    loststatus, loginuid
    FROM ims_principal_data usr
    JOIN am_token tkn on usr.id=tkn.principal_id
    LEFT JOIN am_sw_token_device_types device on
    tkn.sw_token_device_type_id=device.id
    JOIN am_token_oob tkn_oob on
    tkn.id=tkn_oob.am_token_id
    ```

  - Account ID/Name: account
  - AssignedUser: loginuid
  - LastTokenLogin: LastLogin
  - LostStatus: loststatus
  - TokenDisabled: disabled
  - TokenExpires: deathdate
  - TokenType: tokenModel

Continued...
Task 4-5: Create the Authentication Manager Tokens Account Collector, continued

- Mapping for user account mapping attributes:
  - User Account Mappings Data Query:
    ```sql
    select distinct tkn.serial_number as Account, usr.loginuid as LoginID
    from am_token tkn
    join ims_principal_data usr on tkn.principal_id=usr.id
    ```
    - User ID: loginID
    - Account ID/Name: account

- Edit User Resolution Rules
  - Click Add More…
  - Target Collector: Users
  - User Attribute: User Id
  - Click Finish

- Click the collector you just created: Authentication Manager Tokens Account Collector

- Test Collector:
  - Click Test
  - Upon successful test click Cancel
  (If unsuccessful, return to check your query entries and/or perform other troubleshooting as needed)

Continued…
Task 4-5: Create the Authentication Manager Tokens Account Collector, continued

Collect Account
- Click **Collect Accounts**
- Click **OK** to start collection.

Check status of collection
- Click **Collection History** Tab
- Click the **Run** number for details

Once status is “Completed”
- Click breadcrumb on top Application: **Authentication Manager Tokens** then click the **Accounts** tab to see what was collected

You should see approximately 40 accounts of RSA SecurID tokens with user assignments

Example:

<table>
<thead>
<tr>
<th>Account</th>
<th>Business Source</th>
<th>LastTokenLogin</th>
<th>LastStatus</th>
<th>TokenDisabled</th>
<th>TokenExpires</th>
<th>assignedUser</th>
<th>TokenType</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001212520484</td>
<td>Authentication Manager Tokens</td>
<td>2017-01-30 21:21:34</td>
<td>-</td>
<td>no</td>
<td>1/2/2017 12:00 AM</td>
<td>guest</td>
<td>700</td>
</tr>
<tr>
<td>0001212520479</td>
<td>Authentication Manager Tokens</td>
<td>2017-01-30 21:21:34</td>
<td>-</td>
<td>no</td>
<td>1/2/2017 12:00 AM</td>
<td>guest</td>
<td>700</td>
</tr>
<tr>
<td>0001212520478</td>
<td>Authentication Manager Tokens</td>
<td>2017-01-30 21:21:34</td>
<td>-</td>
<td>no</td>
<td>1/2/2017 12:00 AM</td>
<td>guest</td>
<td>700</td>
</tr>
<tr>
<td>0001212520477</td>
<td>Authentication Manager Tokens</td>
<td>2017-01-30 21:21:34</td>
<td>-</td>
<td>no</td>
<td>1/2/2017 12:00 AM</td>
<td>guest</td>
<td>700</td>
</tr>
<tr>
<td>0001212520476</td>
<td>Authentication Manager Tokens</td>
<td>2017-01-30 21:21:34</td>
<td>-</td>
<td>no</td>
<td>1/2/2017 12:00 AM</td>
<td>guest</td>
<td>700</td>
</tr>
</tbody>
</table>

- Adjust the Table Options to display the specific custom attributes you created for the Tokens collection (such as ‘AssignedUser’ to see who a token was assigned to)

Notice that all of the Token Accounts that you have collected are tokens with assigned users.

It might be useful to also collect unassigned tokens for the purposes of reporting and token inventory. For example, to determine how many users you would be able to assign tokens to.

You can do this by modifying the Accounts Data Query for your Authentication Manager Tokens ADC collector.

Continued...
Task 4-5: Create the Authentication Manager Tokens Account Collector, continued

- Navigate to Resources > Applications and select the Authentication Manager Tokens application
- Select the Collectors tab
- Select the Authentication Manager Tokens ADC collector
- Select Edit and Next to the fourth page: Account Data

- In the Accounts Data Query field, replace the existing query with the query below:

  This particular query is in the Class Files\Additional_Query\Additional_AM_Token_Acct_Collector_Query.txt file

```sql
SELECT serial_number as account, last_login_date as LastLogin,
CASE WHEN is_enabled = 'false' THEN 'YES' ELSE 'no' END as disabled,
token_shutdown_date as deathdate,
CASE WHEN token_type = '0' THEN 'Std Card' WHEN token_type = '1' THEN 'PIN Pad' WHEN token_type = '2'
THEN 'SID 700' WHEN token_type = '4' THEN device.family_key WHEN token_type = '9' THEN 'SID 800' END
AS tokenModel,
CASE WHEN is_token_lost = 'true' THEN 'LOST' END as loststatus, loginuid
FROM ims_principal_data usr
JOIN am_token tkn on usr.id=tkn.principal_id
LEFT JOIN am_sw_token_device_types device on tkn.sw_token_device_type_id=device.id
JOIN am_token_oob tkn_oob on tkn.id=tkn_oob.am_token_id
UNION
SELECT serial_number as account, null as LastLogin, 'YES' as disabled,
token_shutdown_date as deathdate,
CASE WHEN token_type = '0' THEN 'Std Card' WHEN token_type = '1' THEN 'PIN Pad' WHEN token_type = '2'
THEN 'SID 700' WHEN token_type = '4' THEN 'SW Token' WHEN token_type = '9' THEN 'SID 800' END
AS tokenModel, null as LostStatus, null as loginuid
FROM am_token
```

What the addition to the original query will do is collect token accounts that do not have assigned users and insert null data into the fields that would have user data if the user was assigned.

The remaining settings and queries remain unchanged.

- Click Next through the next two pages and click the Finish button
- Select Test to ensure valid settings, then select Collect Accounts to collect token accounts with the updated query

- Select the Accounts tab to view the latest collection. You should now see approximately 120 accounts added to your original collection – consisting of a mix of assigned and unassigned RSA SecurID tokens
### Task 4-6: Create the Authentication Manager Token Entitlement Collector

<table>
<thead>
<tr>
<th>Login to IGL as <strong>AveksaAdmin</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to Authentication Manager Application: <strong>Resources &gt; Applications</strong></td>
</tr>
<tr>
<td>□ Click <strong>Available SecurID Tokens</strong></td>
</tr>
<tr>
<td>□ Click <strong>Collectors Tab</strong></td>
</tr>
<tr>
<td>□ Click <strong>Create Entitlement Collector</strong></td>
</tr>
</tbody>
</table>

#### Collector Description:
- **Collector Name:** Available SecurID Tokens EDC
- **Description:** Entitlement Collector for available SecurID tokens
- **Data Source Type:** Database
- **Agent:** AveksaAgent
- **Status:** Active
- **Copy from:** leave blank
- **Schedule:** Leave as No

#### Database Connection:
- **DB Type:** PostgreSQL
- **Driver Class:** org.postgresql.Driver
- **URL:** jdbc:postgresql://am8-p.rsas.com:7050/db?ssl=true
- **User Name:** am8db
- **Password:** Password01$
Task 4-6: Create the Authentication Manager Token Entitlement Collector, continued

- Select types of entitlement data to collect:
  - Check **Accounts** under resource-action entitlements
  - Leave the remaining checkboxes **UN** checked

- Define General Column Names:
  - **User Reference ID/Name**: **Account**
  - **Resource Fully Qualified Name**: **TokenType**
  - **Action ID/Name**: **Serial**

- Mapping for resource attributes:
  - **Resource Data Query**:
    ```sql
    SELECT
    DISTINCT CASE
    WHEN tkn.token_type = '0' THEN 'Std Card'
    WHEN tkn.token_type = '1' THEN 'PIN Pad'
    WHEN tkn.token_type = '2' THEN 'SID 700'
    WHEN tkn.token_type = '4' THEN 'Soft Token - ' || device.family_key
    WHEN tkn.token_type = '9' THEN 'SID 800'
    END AS TokenType
    FROM am_token tkn
    left outer join am_sw_token_device_types device on
    tkn.sw_token_device_type_id=device.id
    UNION
    SELECT 'Soft Token' as TokenType
    ```
  - **Resource ID/Name**: **TokenType**
  - **Resource Fully Qualified Name**: **TokenType**

Continued...
Task 4-6: Create the Authentication Manager Token Entitlement Collector, continued

- **Mapping for resource-action based entitlements -- Resource Entitlement Data:**
  - Resource Entitlements Query:
    ```sql
    SELECT
tkn.serial_number as Serial,
    CASE
    WHEN tkn.token_type = '0' THEN 'Std Card'
    WHEN tkn.token_type = '1' THEN 'PIN Pad'
    WHEN tkn.token_type = '2' THEN 'SID 700'
    WHEN tkn.token_type = '4' THEN 'Soft Token - '
    || device.family_key
    WHEN tkn.token_type = '9' THEN 'SID 800'
    END AS TokenType
    FROM am_token tkn
    left outer join am_sw_token_device_types device on
    tkn.sw_token_device_type_id=device.id
    WHERE tkn.principal_id is null
    UNION
    SELECT 'Next Available' as Serial, 'Soft Token' as TokenType
    ```
  - Resource Fully Qualified Name: **TokenType**
  - ActionID/Name: **Serial**

- **Mapping for resource-action based entitlements -- Account Data:**
  - Ents. for Accounts Query:
    ```sql
    SELECT " as Account, " as TokenType, " as Serial
    FROM am_token
    WHERE 1=0
    ```
  - Entitled Account: **Account**
  - Resource Fully Qualified Name: **TokenType**
  - ActionID/Name: **Serial**

- Leave **Entitlement Relationship Attributes** as is – it is not used.

*Continued...*
Task 4-6: Create the Authentication Manager Token Entitlement Collector, continued

- Account Evaluation
  - Associated account collector: **Authentication Manager Tokens ADC**
  - Account value evaluates to: **Account Name**

- Account Evaluation Processing Order: no change is needed. Click **Finish**

- Select the collector you just created: **Authentication Manager Entitlement Collector**

- Test Connector:
  - Click **Test**
  - Upon successful test click **Close**
  (If unsuccessful, return to check your query entries and/or perform other troubleshooting as needed)

**Collect Entitlements:**

- Click **Collect Entitlements**
- Click **OK** to start collection.

**Check status of collection:**

- Click **Collection History** Tab
- Click the **Run** number for details

*Continued...*
Once status is “Completed”:

- Navigate to Resources > Applications: Available SecurID Tokens
- Click the What Access tab to see what was collected

**Example:**

<table>
<thead>
<tr>
<th>Entitlement name</th>
<th>Entitlement Collector</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010122408520</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408521</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408522</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408523</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408524</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408525</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408526</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408527</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408528</td>
<td>Available SecurID Tokens ADC</td>
</tr>
<tr>
<td>2010122408529</td>
<td>Available SecurID Tokens ADC</td>
</tr>
</tbody>
</table>
Next Steps: Governance Reviews, Reports and Experiments

Now that you have been guided through collections for RSA Authentication Manager user accounts and tokens as accounts, you can use your experience and expertise in RSA Identity Governance to utilize this information in Reviews and Reports. Since this lab environment is essentially the same structure as that used in the RSA Identity Governance and Lifecycle Administration course, you can refer to the lab guide for that course as well as user documents if you need further assistance to create and perform reviews and reports.

You have full administrative access to the RSA Identity Governance platform, RSA Authentication Manager, and source data in the Prestige.com Active Directory. Feel free to use this environment for further experimentation.

Some suggestions and high-level information are provided in this section of the Lab Guide. More detailed instructions are contained in the Appendices to this Lab Guide if you are unfamiliar with creating Reviews, Reports, or Charts.

Reviews

You can structure and run Reviews in RSA Identity Governance on various aspects of RSA SecurID tokenholders – based on information collected from the user and token accounts. You may want to experiment with one or more of these reviews:

- Account Access and Ownership review to review if users with assigned RSA SecurID tokens should legitimately have an assigned token. This supports a Business Driven Security model to ensure that the security level provided by multi-factor authentication is appropriate for the user(s) under review.

- Account Access and Ownership review based on token expiry date. A review definition could include a custom review state to notify an RSA Authentication Manager administrator to begin a token replacement process. This helps ensure that RSA SecurID tokenholders have uninterrupted secure access to the resources that they need to perform their work.

- Account Access and Ownership review to confirm that user accounts designated as ‘Privileged’ (AM users with administrator privileges) should have an administrative role. This helps ensure that user accounts do not have over-reaching or inappropriate privilege in the security platform.
Review Tokenholders

This review is conducted to review users with assigned tokens. Any user that should not have an RSA SecurID token assigned can be identified, notification made to the appropriate person and corrective action taken.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Log in to RSA IGL as AveksaAdmin/Aveksa123 and select Reviews &gt; Definition</td>
</tr>
<tr>
<td>2</td>
<td>Select the Create Review Definition button</td>
</tr>
<tr>
<td>3</td>
<td>Under Review Type, select the Account Access and Ownership Review radio button</td>
</tr>
</tbody>
</table>
| 4    | Name this review Tokenholder Review  
Description: Review RSA SecurID token assignments  
Review State: Active  
Review Instructions: For each user who has an assigned RSA SecurID token, select ‘KEEP’ to remain assigned or ‘REMOVE’ to have the user’s token unassigned.  
The remaining settings can stay at their default values. |
| 5    | On the next Account Selection page, select All accounts including deleted accounts  
For the purposes of this particular exercise, we also want to filter accounts and look only at accounts with a number as their name (the token serial number). Otherwise, we will list all user accounts in the review.  
Select the Filter accounts checkbox click the All link and as the filter criteria, set Name contains 000 (three zeros). All of the RSA SecurID token serial numbers in this lab environment contain at least three leading zeros. This will include only the accounts with token numbers. |
| 6    | On the Contents page, you can leave the selections at their default settings |
| 7    | On the Reviewers page, for the purposes of this exercise, select Selected reviewers review all accounts and assign the responsibility to AveksaAdmin |
| 8    | On the next two – Monitors and Display Options- pages, you do not need to make any changes |
| 9    | On the States page, enter a value of KEEP for the Maintain display and a value of REMOVE for the Revoke display.  
Feel free to experiment with additional states – perhaps ‘Disable’ or adding a Custom State that might apply to this type of review |
| 10   | On the Configuration page, you do not need to make any changes |
| 11   | Click Finish then select the review you defined and Run it |

Continued…
### Review Tokenholders, continued

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 12   | As AveksaAdmin, go to the Home page and select, then Perform the review
(If you selected review by Supervisor, you will need to log in as the supervisor of user with an assigned token to perform the review as that supervisor) Depending on what information you have collected, you will at least see all assigned tokens. By adjusting the table display options, you can view other attributes such as username ("AssignedUser") and LastTokenLogin to assist your review. You can also return to the Review Definition and experiment with different Filter Account parameters. |

### Review Token Expiry

This review is conducted to review users with tokens expired or near expiration. This review includes a custom state to notify a security administrator to take action to replace the token.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Create Review Definition</td>
</tr>
</tbody>
</table>
| 2    | Type: Account Access and Ownership
Name: Token Expiry Review
Description: Review expiring RSA SecurID tokens
Review State: Active
Review Instructions: For each token, select ‘Maintain’ to take no action, ‘Revoke’ to unassign the token from the user, or ‘Replace’ to notify security to replace the token.
The remaining settings can stay at their default values for this page.
Other settings can be the same as the previous exercise except for the two pages below (Steps 3 and 4). |
| 3    | Select All Active Accounts and filter parameters as: TokenExpires before <approx. 30 days from today's date> |
| 4    | Forward to the Reviewers page and select AveksaAdmin as the reviewer |
| 5    | Forward to the States page and add a Custom state: Replace that Generates change request |
| 6    | Finish and Run the review.
As AveksaAdmin or other user as you selected, access the review and select Perform to see the expired or expiring tokens and their users. (Again, you may need to adjust your displayed Table Options)

Hint: If you do not see any results (a review will not run if there are no results), make sure your token account collector has mapped the Expiration Date correctly. All tokens have expiration dates associated with their account data ("TokenExpires" attribute).
Privileged User Review

This review is conducted to review user accounts that have a ‘Privileged’ type – some level of administrative privilege in RSA Authentication Manager.

The definition outlined in task is simplistic but you may embellish it with additional review states or notification settings, as you wish.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create Review Definition as in the previous task</td>
</tr>
<tr>
<td>2</td>
<td>Virtually all of the parameters and settings are the same as the previous reviews. On the Account Selection page, configure an account filter as: account_type = Privileged (because ‘Privileged’ is a freeform text entry, make sure the spelling is correct – this filter requires an exact match)</td>
</tr>
<tr>
<td>3</td>
<td>Run and perform the review as AveksaAdmin to view the results. In your lab environment, many of the ‘Privileged’ accounts are service accounts. If you wish to experiment, log in to the RSA Authentication Manager Security Console and assign some administrative role to an ordinary user. This review will show any user that has the elevated privilege.</td>
</tr>
</tbody>
</table>
Reports

You can structure and run Reports on collected data relating to RSA SecurID tokenholders. You may want to experiment with one or more of these reports:

- Account Report to list users who have not logged in since a certain date. [Query all accounts; filter on Last Login Date]
- Custom Report to list RSA SecurID tokens and the users where a token has expired or will expire in the next 30 days. [Filter on: $CADx < 30$ days from current date where $CADx$ is the Database ID assigned to the expiration date]

No Login Since... Report

This report lists users who have not logged in since a certain date. This report could be used to identify users who could have their tokens re-assigned to an active user, flagged for follow-up to see if the issued token has been lost, or otherwise investigated to see why the user has become inactive.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Reports &gt; Tabular</td>
</tr>
<tr>
<td>2</td>
<td>Select the Create Report button</td>
</tr>
<tr>
<td>3</td>
<td>Give the report a name of your choosing&lt;br&gt; Select Account as the Report Type&lt;br&gt; Select Accounts from the Template pull-down list&lt;br&gt; Select Application: Authentication Manager Tokens from the Scope pull-down list</td>
</tr>
<tr>
<td>4</td>
<td>Under the Query tab, you can leave the FROM Clause as $V_{AVR_ACCOUNTS}$ to retrieve all accounts</td>
</tr>
<tr>
<td>5</td>
<td>Under the Columns and Display Attributes tabs, you can make any selections you wish or leave the default settings&lt;br&gt; Some columns that might make sense are:&lt;br&gt; - Account Name (this is the token Serial Number for the Auth. Manager Tokens application)&lt;br&gt; - Account.AssignedUser&lt;br&gt; - Account.LastTokenLogin&lt;br&gt; - Account.LostStatus (to see if a token has not been used for login because it’s lost)</td>
</tr>
<tr>
<td>6</td>
<td>Under the Filter Criteria tab, select the following: &lt;br&gt; $\text{Attribute} \quad \text{Operator} \quad \text{Value}$&lt;br&gt; $Account_AssignedUser \quad \text{IS NOT NULL} \quad \text{AND}$&lt;br&gt; $Account_LastTokenLogin &lt; \text{&lt;select current date or other past date of interest&gt;}$&lt;br&gt; The Preview button will display data from your filter selection.</td>
</tr>
<tr>
<td>7</td>
<td>You can add Grouping and Sorting and Schedule/Email selections if you wish.&lt;br&gt; When you are satisfied with your report, select OK to save it.</td>
</tr>
</tbody>
</table>
Expiring Token Report

This report lists tokens that have expired or about to expire.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Reports &gt; Tabular then Create Report</td>
</tr>
<tr>
<td>2</td>
<td>Name the report and let the Type be Account; Template: Accounts; Scope: Application: Authentication Manager Tokens</td>
</tr>
<tr>
<td>3</td>
<td>The query will be V_AVR_ACCOUNTS to query all token accounts but we will filter the result by date in a moment.</td>
</tr>
<tr>
<td>5</td>
<td>For Filter Criteria, filter on Account.TokenExpires &lt; [30 days from today’s date]</td>
</tr>
<tr>
<td>6</td>
<td>Preview your report to verify that you are seeing the correct data</td>
</tr>
<tr>
<td></td>
<td>Select OK to save the report</td>
</tr>
</tbody>
</table>
Charts

You can create a variety of charts that draw upon the data you have collected from RSA Authentication Manager. Some suggestions for charts are:

- Token Summary chart to display an overall summary of RSA SecurID tokens: Unassigned, Never used, Deleted, Never Reviewed and Disabled.
- Available vs Assigned Tokens Chart to display a comparison of the number of available tokens with the accounts that have an RSA SecurID token assigned.
- Used vs Unused Tokens chart to display a comparison of assigned tokens that have been used to log in versus those tokens that have never been used.
- Token Types chart to display the various types of hardware and/or software tokens that have been collected from RSA Authentication Manager.

**Tip:** that the lengthy queries used for these charts can be copied from files in the Class Files\Queries folder on the desktop of the Jumphost machine.

Before creating charts, you will need to know the Account Data Collector ID for your RSA SecurID token collector. The Account Data Collector ID appears in queries as “adc_id”.

You can view the value for your environment by opening your token account collector and noting the **Oid** value in the browser URL: **Oid=22** in this illustration) Typically, this value will be “22” but if it is different for your environment, be sure to substitute the correct value in the queries for the charts described in this section.
Token Summary Chart

This chart report displays an overall summary of RSA SecurID tokens: Unassigned, Never used, Deleted, Never Reviewed and Disabled. This chart might be used in a dashboard displayed for a security administrator or similar role.

*Remember that you can copy/paste this query into your environment or access the query text files in the JumpHost Class Files\Queries\CHART_DATA_QUERIES folder*

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select <strong>Reports &gt; Charts</strong> then <strong>Create Chart</strong></td>
</tr>
<tr>
<td>2</td>
<td>Name the report and let the Type be <strong>Custom</strong></td>
</tr>
<tr>
<td>3</td>
<td>Name the report and let the Type be <strong>Single Series Chart</strong></td>
</tr>
</tbody>
</table>

For the query, enter:

```sql
(SELECT -- total tokens
  'Total All SecurID Tokens' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id = 22
UNION all

SELECT -- total assigned tokens
  'Assigned Tokens' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id=22 AND CAS5 is not null
UNION all

SELECT -- total assigned but Never logged in
  'Tokens Never Logged In' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id = 22 AND CAD2 is null AND CAS5 is not null
UNION all

SELECT -- total expired tokens
  'Tokens Expired' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id = 22 AND CAD3 < current_timestamp
UNION all

SELECT -- total disabled
  'Tokens Disabled' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id = 22 AND CAS5 is not null AND CAS8 = 'YES'
UNION all

SELECT -- total lost tokens
  'Lost Tokens' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
WHERE adc_id=22 AND CAS6 = 'LOST'
UNION all

SELECT -- Available - not assigned and unexpired
  'Total Tokens Available' as Summary, Count(*) as Total
FROM avuser.t_av_accounts
where adc_id=22 AND CAS5 is not null AND CAD3 > current_timestamp)
```

*Continued…*

Substitute the highlighted `adc_id=`, `CAS` and `CAD` values in the query with the identifier for your environment if different.
**Token Summary Chart, continued**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>This query counts all tokens from RSA AM (assigned, available and expired); Tokens that are assigned (AssignedUser CAS5 has a value); Tokens that have never been used for login (LastTokenLogin date is null); Tokens that have expiration dates prior to the current date; Disabled tokens (Token Disabled attribute = “YES”); and Lost tokens (LostStatus =“LOST”). The last lines total the tokens that are available (not assigned and not expired) from the overall population.</td>
</tr>
</tbody>
</table>
| 6 | Remaining settings for the chart can stay at their defaults  
You can decide how you want this chart to display (Column, Pie, 3D, etc.) |
| 7 | Select **OK** to save your chart  
*Hint: No value in the Total column should be ‘0’. If you have a ‘0’ in this column, double-check the query that all adc_id and CAD/CAS values are correct.* |
Available vs Assigned Tokens Chart

This chart report displays the number of available tokens (unexpired tokens that could be assigned to users) compared with the number of accounts that have an RSA SecurID token assigned.

This chart might be used in a dashboard displayed for a security administrator or Operations role that needs to know if sufficient tokens are available for users.

*Remember that you can copy/paste this query into your environment or access the query text files in the Jumphost **Class Files**\Queries folder*

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select <strong>Reports &gt; Charts</strong> then <strong>Create Chart</strong></td>
</tr>
<tr>
<td>2</td>
<td>Name the report and let the Type be <strong>Single Series Chart</strong></td>
</tr>
</tbody>
</table>
| 3    | For a query, enter:  
  
  ```sql
  (SELECT -- total token accounts which are not assigned and unexpired  
    'Total Available' as Status,  
    Count(*) as total  
  FROM avuser.t_av_accounts  
  where adc_id=22 AND CAD3 > current_timestamp  
  AND CAS5 is null  
  union all  
  SELECT -- total accounts which are assigned  
    'Total Assigned' as Status,  
    Count(*) as total  
  FROM avuser.t_av_accounts  
  where adc_id=22 AND CAS5 is not null)
  ```
  
  Substitute the highlighted `adc_id=` and `CAS` attribute ID value in the query if different than those values shown |
| 4    | This query displays the total number of available token accounts (tokens that are not expired \(\text{CAD3} > \text{today}\) and not assigned to a user\[CAS5 not null\]) vs the count of tokens that have a user assignment (\(\text{CAS5 AssignedUser attribute is populated}\)).  
  In your environment, if TokenExpires is an attribute identifier other than \(\text{CAD3}\) or if \(\text{AssignedUser}\) was assigned to an attribute other than \(\text{CAS5}\), use your identifier(s) in this query. |
| 5    | Remaining settings can stay at their defaults  
  You can decide how you want this chart to display (Column, Pie, 3D, etc.) |
| 6    | Select **OK** to save your chart |
Tokens Used vs Unused Chart

This chart report displays a comparison of tokens assigned to users that have been used to log in versus those tokens that have never been used.

*Remember that you can copy/paste this query into your environment or access the query text files in the Jumphost Class Files\Queries folder*

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Reports &gt; Charts then Create Chart</td>
</tr>
<tr>
<td>2</td>
<td>Name the report and let the Type be Single Series Chart</td>
</tr>
<tr>
<td>3</td>
<td>For a query, enter: (SELECT -- total assigned tokens which have logged in vs those never logged in 'Never Logged In' as Status, Count(<em>) as total FROM avuser.t_av_accounts WHERE CAS5 is not null and CAD2 is null) UNION all (SELECT -- total tokens logged In 'Logged In' as Status, Count(</em>) as total FROM avuser.t_av_accounts WHERE CAS5 is not null and CAD2 is not null)</td>
</tr>
<tr>
<td>4</td>
<td>Remaining settings can stay at their defaults</td>
</tr>
<tr>
<td>5</td>
<td>Select OK to save your chart</td>
</tr>
</tbody>
</table>

In this query, CAS5 counts the token assignments (where AssignedUser not null) and where CAD2 is null (LastTokenLogin has no login date) – therefore, never has never been used.

In your environment, if AssignedUser was assigned to an attribute identifier other than CAS5 or if LastTokenLogin was assigned to an attribute other than CAD2, use your identifier in this query.
Token Types Chart

This chart report displays a summary of the types of all tokens collected.

Remember that you can copy/paste this query into your environment or access the query text files in the Jumphost Class Files\Queries folder.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Reports &gt; Charts then Create Chart</td>
</tr>
<tr>
<td>2</td>
<td>Name the report and let the Type be Single Series Chart</td>
</tr>
</tbody>
</table>
| 3    | For a query, enter:  
  (SELECT CAS7 as Token_Type, Count(*) as Total  
  FROM avuser.t_av_accounts  
  WHERE adc_id=22  
  GROUP BY CAS7)  
  In your environment, if the TokenType attribute was assigned to an attribute identifier other than CAS8, use your identifier in this query. Also remember to use the correct adc_id value for your implementation.  
  Note that a Software Token that has been assigned to a user has a further sub-type (“Generic AES 128” or “Desktop PC” in the population used in this exercise environment) whereas an unassigned Software Token has a non-specific type of “SW Token”. |
| 4    | Remaining settings can stay at their defaults  
  You can decide how you want this chart to display (Column, Pie, 3D, etc.) |
| 5    | Select OK to save your chart |

If you wanted to enhance this chart – for example, to eliminate expired tokens for an up-to-date inventory of token types – modify the query accordingly.

Feel free to experiment with other reports and charts based on the samples we have described or using your own design.
Dashboards

With reports and charts available, you can assemble a custom dashboard to display summary information for the RSA SecurID token data that you collect.

Dashboard

The following is one suggestion for arranging a report and charts as a dashboard.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Admin &gt; Dashboards then Create Dashboard</td>
</tr>
<tr>
<td>2</td>
<td>Name the dashboard and create a Proportional Layout of your choosing to incorporate the reports and charts that you have created. A suggested layout is shown below.</td>
</tr>
</tbody>
</table>
Troubleshooting

If you see this error:

```
java.sql.SQLException: No suitable driver found for jdbc:postgresql://<host_IP>:7050/db?ssl=true
```

This usually means the URL is mal formed.

Check to see that the default double prefix has been removed.

```
jdbc:jdbc:postgresql://<host_IP>:7050/db?ssl=true
```

Check to make sure there are no extra dashes, colons, etc.

Check to make sure the prefix is not capitalized.

**Example:** `jdbc:postgresql://10.101.240.10:7050/db?ssl=true`

Make sure the postgresql driver has been deployed using the ‘customizeACM’ function (see Exercise 2)

If you see this error:

```
Org.postgresql.util.PSQLException:FATAL:no pg_hba.conf entry for host "<host_IP>", user :testuser", database"db",SSL on
```

This usually means the IP address or IP range and/or the subnet mask are not correct or missing from the database user entry on Auth Manager.

This is needed to allow RSA IGL to connect to Auth Manager

Update the IP address and/or the IP mask of the client machine used to connect to the database using the read-only user on the Auth Manager server:

1. Connect to the Appliance using an SSH client.
   a. Log on as rsaadmin using the operating system password specified during Quick Setup.
   b. Type `cd /opt/rsa/am/utils`, and press Enter.
2. Type the following command:

   ```
   ./rsautil manage-readonly-dbusers -a update -o <OC_username> -u <database_user_name> -i <IP_address> -n <IP_mask>
   ```

   where:
   - `OC_username` is the Operations Console administrator user name;
   - `database_user_name` is the user name you specify for the read-only user;
   - `IP_address` is the new IP address of the users' client machine. (Do not specify this argument if you do not want to update the IP address but just want to update the IP mask.);
   - `IP_mask` is the new IP mask. (Do not specify this argument if you do not want to update the IP mask but just want to update the IP address.)
3. Press Enter.
4. When prompted, enter the Operations Console Admin password (Password01$ for this environment).
5. Press Enter.

**Example:**

```bash
./rsautil manage-readonly-dbusers -a update -o ocadmin --u testuser -i 10.10.100.10 -n 255.255.255.0
```
Feedback

After you complete this lab exercise, please click the following link to provide your feedback on this experience: http://survey.walkerinfo.com/RSAelearn. You can also follow the shortcut (“COURSE EVALUATION”) on the Jumphost desktop to access this survey.

Thank You!