



SECRETS MANAGER INTEGRATION - TECHNICAL DOCUMENTATION

Soroco

Prospero Automation Platform

2019.04

May 17, 2019

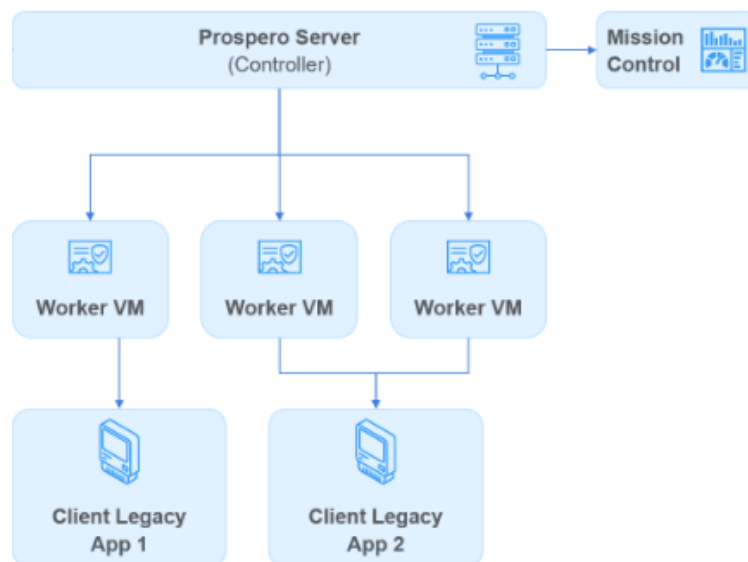
SOROCO SOLUTION OVERVIEW

Prospero is an industry-leading automation platform for building, running, and monitoring complex software automation systems within the enterprise. The platform includes three complementary components: an expressive software development kit for building automation, a server for runtime execution, and a suite of dashboards for performance monitoring, troubleshooting, and maintenance.

The remainder of this document focuses on the integration of CyberArk Application Access Manager (SECRETS MANAGER) as a third-party credential management provider to the Prospero automation platform. It is also possible to fetch credentials using the CyberArk REST endpoints. Fetching credentials using the REST endpoints are covered here as well.

KEY BENEFITS

The Prospero platform is responsible for the end-to-end execution of a Soroco automation system. The Prospero platform features a distributed architecture – with the central Prospero *server* scheduling the execution of automation logic across one or more *workers*.

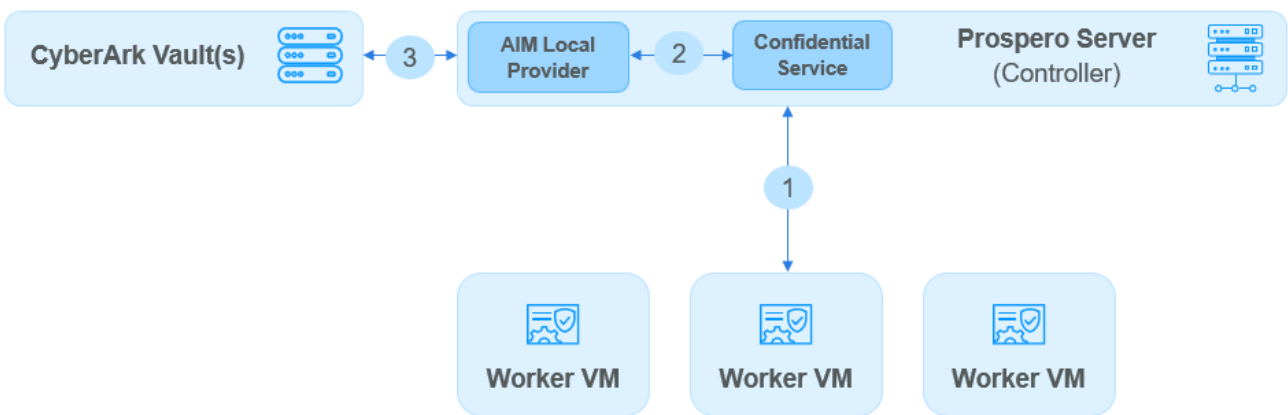


Before a worker can perform an action inside a legacy application or system, the worker typically requires secure access to application/system credentials. While the Prospero platform already includes native credential management, several Soroco customers have specifically requested CyberArk-backed credential management. The integration of CyberArk **Application Access Manager** (SECRETS MANAGER) with the Prospero Server allows Soroco to offer this functionality to our customers.

DIAGRAM & DESCRIPTION OF INTEGRATION

I. SECRETS MANAGER Integration with the Prospero Platform

The *Confidential* service deployed within the Prospero server offers automation developers a centralized credential management solution. The architecture of the service is also extensible, allowing automation developers the ability to *delegate* credential management to external providers. Integration of the CyberArk SECRETS MANAGER system takes full advantage of this architecture.



The CyberArk local provider is installed on the Prospero server. When a worker requests a credential [1], the Confidential service delegates the request to the SECRETS MANAGER Local Provider [2]. The AAMSecrets Manager local provider then communicates with the CyberArk service to fetch the requested credentials [3]. The following section describes credential retrieval in further detail.

II. Credential Retrieval Flow

1. Workers periodically retrieve credentials by invoking a REST endpoint exposed by the Confidential service. The request might appear like this:

```
confidential.getitem('secretcred')
```

where `secretcred` is a reference to a *CyberArk object* containing the desired credentials.

2. The Confidential service receives the request for `secretcred`. Since the Confidential service has been configured for delegation to CyberArk, the SECRETS MANAGER local provider is

executed. The Confidential service could also be configured to invoke the CyberArk REST endpoint.

3. The SECRETS MANAGER local provider forwards the credential request to the CyberArk service using the following information:
 - a. A pre-configured APPID
 - b. The requested CyberArk object: `secretcred`
 - c. A derived CyberArk safe name. The safe name matches the automation system name.

SECRETS MANAGER INSTALLATION

The SECRETS MANAGER client executable needs to be installed on the Prospero Linux server as a standard executable. This installation does not require special setup, and the latest Credential Provider and ASCP Implementation guide should be followed for installation.

The Central Credential Provider (CCP) is also supported. Refer to the Central Credential Provider implementation guide for installation/configuration.

SECRETS MANAGER CONFIGURATION

DEFINING THE APPLICATION ID (APPID) AND AUTHENTICATION DETAILS

1. Click on **Applications > +Add Application**. Create an application named **Soroco application**. Specify the following information:

The screenshot shows the 'Add Application' dialog box in the CyberArk PVWA interface. The dialog is titled 'Add Application' and contains the following fields and options:

- Name:** Soroco application
- Description:** The Soroco application. The Soroco application will added as a member to soroco-safe.
- Business owner:**
 - First Name:** John
 - Last Name:** Graham
 - Email:** john.graham@soroco.com
 - Phone:** 617-555-5555
- Location:** \
- Access Permitted:** From: [dropdown] To: [dropdown]
- Expiration Date:** [calendar icon]
- Disabled:** [checkbox]

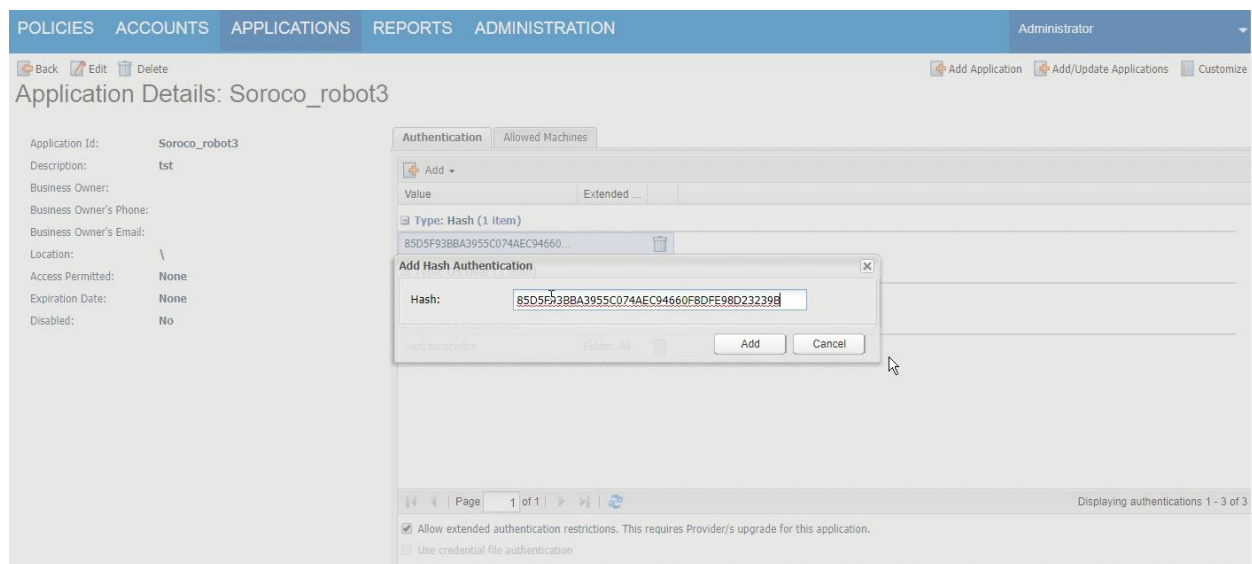
The 'Add' and 'Cancel' buttons are located at the bottom right of the dialog. The background shows the 'Applications List' page with a search bar and a table header 'Application Id'.

2. Click **Add**. The application is displayed in the Application Details page. By default, no authentication options have been configured. Check the **Allow extended authentication restrictions** checkbox. This enables you to specify an unlimited number of machines and the Windows domain OS users for a single application.

Note: CyberArk PVWA also supports a rich set of authentication options, including:

- OS User
 - We tested this configuration with the value: **soroco**
 - This is the default user account running the Confidential service.
- Application path
 - We tested this configuration with the value: **/opt/soroco/bin**
 - This the path executing the call the to the SECRETS MANAGER client.
- Application hash
 - We tested this configuration by computing the hash of the **waitress-serve** command which launches the Confidential service.

```
/opt/CARKaim/bin/aimgetappinfo gethash -FilePath
/opt/soroco/bin/waitress-serve
>>> 85D5F93BBA3955C074AEC94660F8D23239B
```



- IP address
 - We tested with the machines IP address

The following trusted CLI shells must be added:

```
/opt/soroco/embedded/bin/runsv
/usr/bin/bash
/opt/soroco/bin/python3
```

Follow the procedure described on the Credential Provider and ASCP implementation guide.

- Make sure the SECRETS MANAGER Credential Provider user is given the following permissions to retrieve credentials from the required safes: “List accounts”, “Retrieve accounts”, “View Safe Members”
- Make sure the **Soroco Application** has “Retrieve accounts” permission from the required safe.

CONFIGURING THE CONFIDENTIAL SERVICE TO INVOKE SECRETS MANAGER

1. On the Prospero server, configure the Confidential service to delegate to the credential requests to CyberArk. Edit the `/opt/soroco/config/confidential.yaml` file, and specify CyberArk as the back-end for the secret management service (SMS):

/opt/soroco/config/confidential.yaml file:

```
sms: confidential_service.backend.cyberark.sms.SMS
```

2. On the Prospero server, configure the executable location of the SECRETS MANAGER local provider. Edit `/opt/soroco/config/confidential.yaml`, and specify the desired application id and executable path of the AAMSecrets Managercal provider:

/opt/soroco/config/confidential/cyberark/cyberark.yaml

```
APPID: Soroco application
FOLDER: Root
BINARY_PATH: /opt/CARKAIM/sdk/clipasswordsdk
```

3. On the Prospero server, you can test the retrieval of your inserted credential with the following snippet of python code. This will validate whether your end-to-end setup is working properly.

```
$ python3

>>> from confidential import Confidential
>>> c = confidential(cert=('automation-system.crt', 'automation-
system.key'))
```

```
>>> c.getitem('secretcred')
'password123'
>>>
```

If you see your previously configured password get retrieved, congratulations! CyberArk is now configured as the credential management provider for the Prospero automation platform!

REST CLIENT CONFIGURATION

As previously mentioned, it is also possible to fetch CyberArk credentials through the Central Credential Provider. Credential retrieval using the REST endpoints makes it easier to enforce some of the authentication options on the Application modal dialog. To fetch credentials using REST, complete the following steps:

1. Convert your CCP certificates to PEM format. PEM format is required by Python and other libraries. For example, the certificates RootCA.P12 and client.p12 files can be converted using the following command:

```
openssl pkcs12 -in client.p12 -nocerts -out cyberark.key -nodes
openssl pkcs12 -in client.p12 -nokeys -out cyberark.crt -nodes

>> cyberark.key
>> cyberark.crt

openssl pkcs12 -in RootCA.P12 -nokeys -out root.crt -nodes

>> root.crt
```

2. Test connectivity using curl:

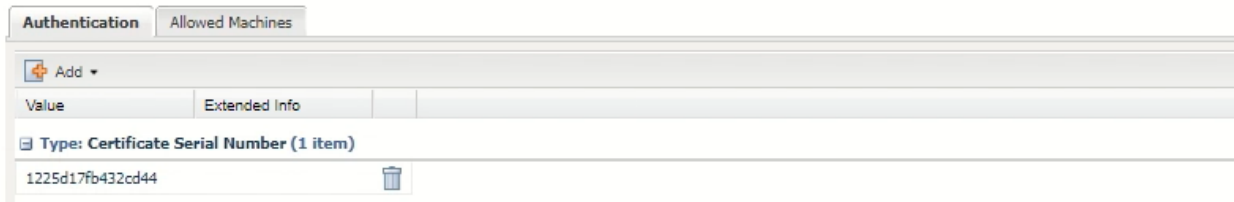
```
Curl 'https://services-uscentral.skytap.com:17285/AIMWebService/api/Accounts?AppID=Soroco
application&Query=Safe=SAFEName;Folder=root;Object=Operating+System-
WinDesktoplocal-10.0.1.20-sv_account' --cacert root.crt --cert cyberark.crt -
key cyberark.key

>>
```

The desired credential is returned:

```
json response:
{
    // credential returned inside content block
}
```

Note: This was tested using the **Certificate Serial Number** authentication option:



A certificate serial number can be obtained using:

```
openssl x509 -in cyberark.cert -noout -text
```

CONFIGURING THE CONFIDENTIAL SERVICE TO INVOKE THE CYBERARK REST ENDPOINT

With the ability to fetch credentials using the curl command, configure the Confidential service to call the REST endpoint. Complete the following steps:

1. Copy the generated certificates from the previous section over to the Confidential service:

```
cp root.crt /opt/soroco/certificates/cyberark-root.crt
cp cyberark.crt /opt/soroco/certificates
cp cyberark.key /opt/soroco/certificates
```

2. Specify the CyberArk REST Service as the back-end for the secret management service (SMS):

/opt/soroco/config/confidential.yaml file:

```
sms: confidential_service.backend.cyberarkrest.sms.SMS
```

3. Modify the cyberarkrest backend configuration file:

/opt/soroco/config/confidential/cyberarkrest/cyberarkrest.yaml

```
APPID: Soroco application
FOLDER: Root
URL: <ULR>
TLS_CACERT: /opt/soroco/certificates/cyberark-root.crt
TLS_KEY: /opt/soroco/certificates/cyberark.key
TLS_CERT: /opt/soroco/certificates/cyberark.crt
```

4. On the Prospero server, you can test the retrieval of your inserted credential with the following snippet of python code. This will validate whether your call to the REST Service is working properly.

```
$ python3

>>> from confidential import Confidential
>>> c = confidential()
>>> c.getitem('secretcred')
'password123'
>>>
```

PARTNER PRODUCT INSTALLATION & INTEGRATION CONFIGURATION

Contact Soroco for automation platform setup and configuration.

PARTNER CONTACT INFO

Primary Contact	Name	Bence Magyar
	Email	bence.magyar@soroco.com