Step-by-Step Guide on using HIA Connectors

By Rafael Ulhoa

Table of Contents

[Version Tracking 3](#_Toc190952337)

[Acronyms 4](#_Toc190952338)

[Step by Step Guide 5](#_Toc190952339)

[Generating a Plugin Key 6](#_Toc190952340)

[Setup a Connector 8](#_Toc190952341)

[Cloud Connectors 8](#_Toc190952342)

[On-Premises Connectors 10](#_Toc190952343)

[Create Automation Policies 20](#_Toc190952344)

[Generic Inicial Steps 20](#_Toc190952345)

[Configuring Actions: Connector Specific 21](#_Toc190952346)

[Generic Final Steps 28](#_Toc190952347)

[Advisories 31](#_Toc190952348)

[Generic Information 31](#_Toc190952349)

[How to delete a plugin credential 31](#_Toc190952350)

[Connector failing to connect due to Self Signed Certificate 32](#_Toc190952351)

[Connector Specific 34](#_Toc190952352)

[TSSA: How HIA interprets job executions 34](#_Toc190952353)

[Innovation Studio: Example Workflow that runs a Control-M job 36](#_Toc190952354)

[Generic Rest Connector: How to locate the configured Endpoint URL 40](#_Toc190952355)

# Version Tracking

|  |  |
| --- | --- |
| Change date | Changes |
| 2022-06-30 | * Guide created |
| 2022-07-05 | * Added “Table of Contents” and organized subsections of the guide to make navigation easier * Created “Acronyms” section * Created “Advisories” section * Migrated “Linux Disk Cleanup” job’s description from step #27 to the “Advisories” section and elaborated more on the behavior between HIA and TSSA * Elaborated a bit more on step #23 on how a policy filter works |
| 2022-07-07 | * Added advisory on how to delete a plugin credential * Updated step #19 on configuring the plugin’s credential   + We must create the folder structure the plugin uses before configuring the credential. Rewrote the step to do this with a simple command.   + Fixed the URL. It should be “https” and not “http” |
| 2022-10-19 | * Tweaked document name for clarity |
| 2022-12-30 | * Added an advisory on how to resolve the self-signed certificate error that prevents connectors from connecting. |
| 2023-02-14 | * Reorganized document to be a broader HIA guide and not just focused on using the TSSA connector; * Included instructions on how to use the Generic REST connector * Included instructions on how to use the Innovation Studio connector |

# Acronyms

***BMC TrueSight Server Automation = TSSA***

* Descriptions: TrueSight Server Automation, one of BMC's digital enterprise automation solutions, allows you to quickly and securely provision, configure, patch, and maintain physical, virtual, and cloud servers.
* In previous versions it was called ***BladeLogic Server Automation (BSA)***

***BMC Helix Intelligent Automation = HIA***

* HIA is a subcomponent of ***BMC Helix Operations Manage*** (***BHOM)***. Typically pronounced as *Be-home*). BHOM manages events and HIA enables the execution of automations for those events.
* Description: BMC Helix Intelligent Automation is an automation broker that enables organizations to connect with automation tools of their choice and define policies to trigger remediation actions. It acts as a Manager of Managers for connecting with third-party automation tools, listens to the incoming events from data sources such as BMC Helix Operations Management and BMC Helix AIOps and enables creation of event-driven automation policies to remediate events.

***BMC Helix Control-M = CTM***

# Step by Step Guide

To use HIA on-prem connector you need to:

1. **Generate a Plugin Key**: this is what on-premises connectors use to authenticate with HIA;
2. **Setup a Connector**: connectors act as brokers that enables HIA to talk with other applications so that automations can be run. There are two types of connectors:
   1. **Cloud Connector**: These run directly in the Helix SaaS tenant. Examples include the ones for AWS and Innovation Studio.
   2. **On-Premises Connector**: All choices use the same installer called remote-restapi-plugin, with the difference being on how it is configured in the remote server.
3. **Create automation policies:** HIA talks through specific connectors to applications to do automation run calls.

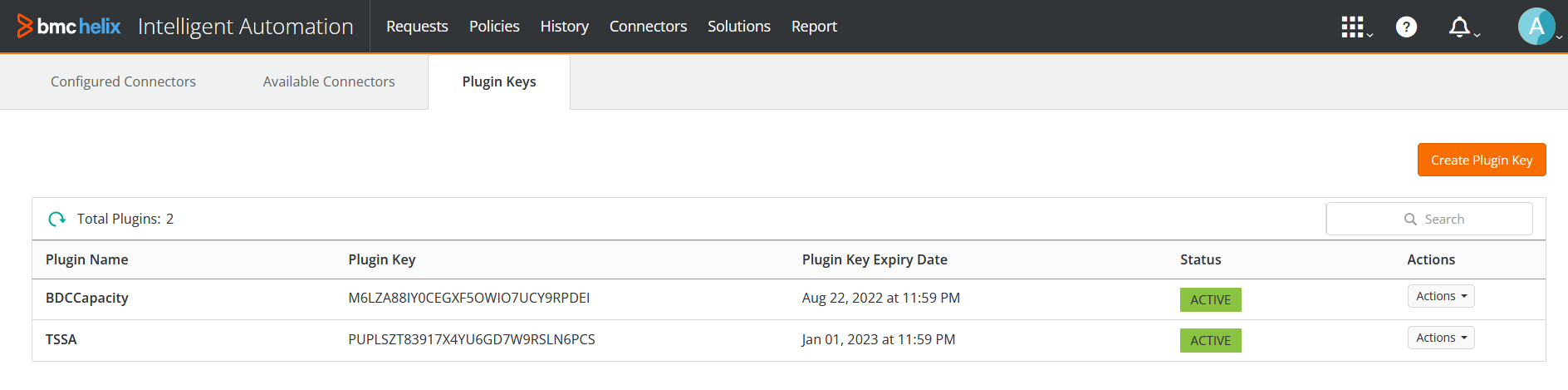
## Generating a Plugin Key

This is only needed if you’re using an on-premises connector.

It’s a recommended practice to use a unique Plugin Key per server you’ll deploy a connector to, so that you can selectively revoke a key if needed.

Steps:

**Login to Helix and go to “Intelligent Automation”->Connectors->”Plugin Keys”**



**Click on “Create Plugin Key”**

Define the Key Expiry

* This is a demo environment, so feel free to give yourself plenty of time (it can be months or years)
* You can update this anytime you want later.

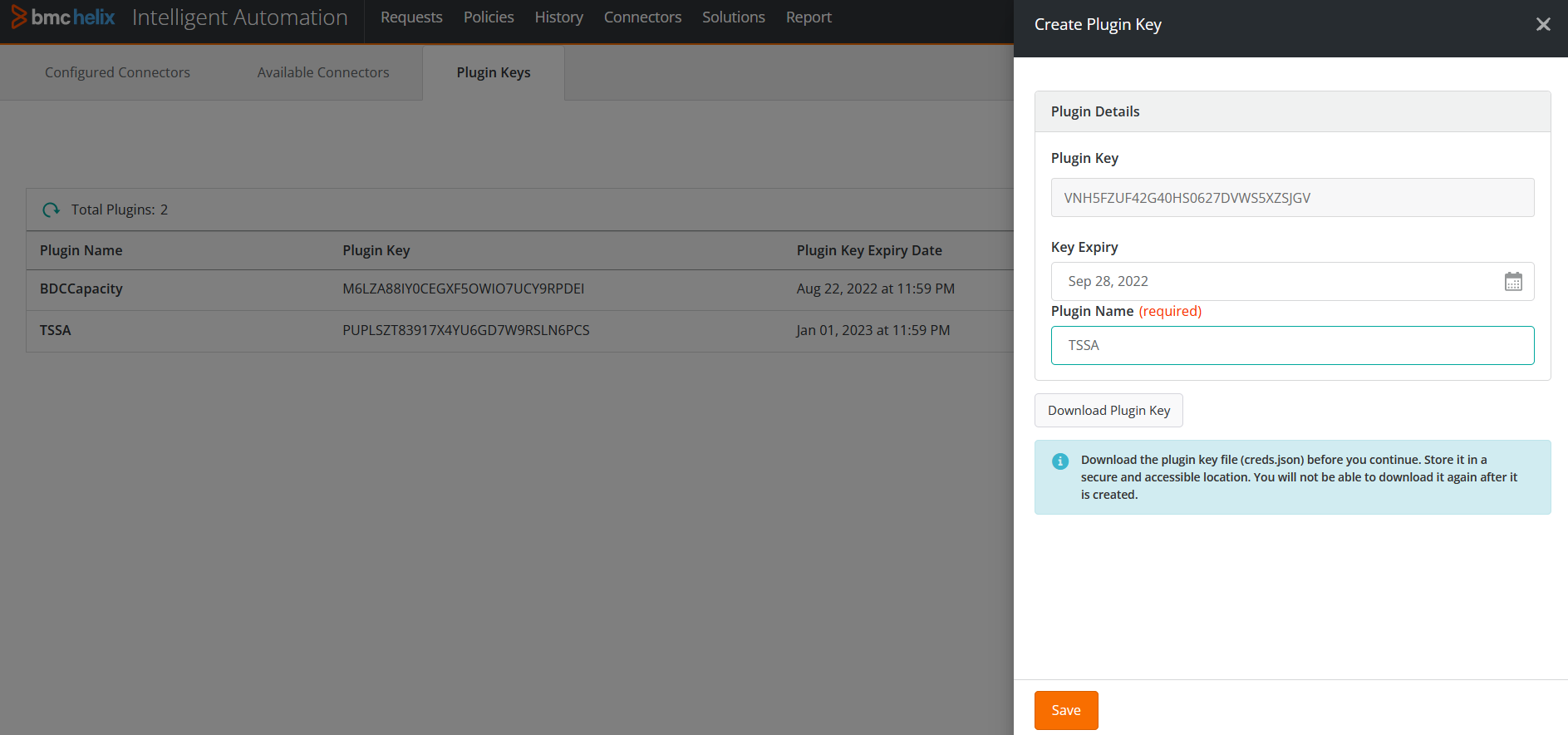
Give it a name.

* A meaningful name helps you understand why the key was generated.
* You can update this anytime you want later.

Download the Plugin Key.

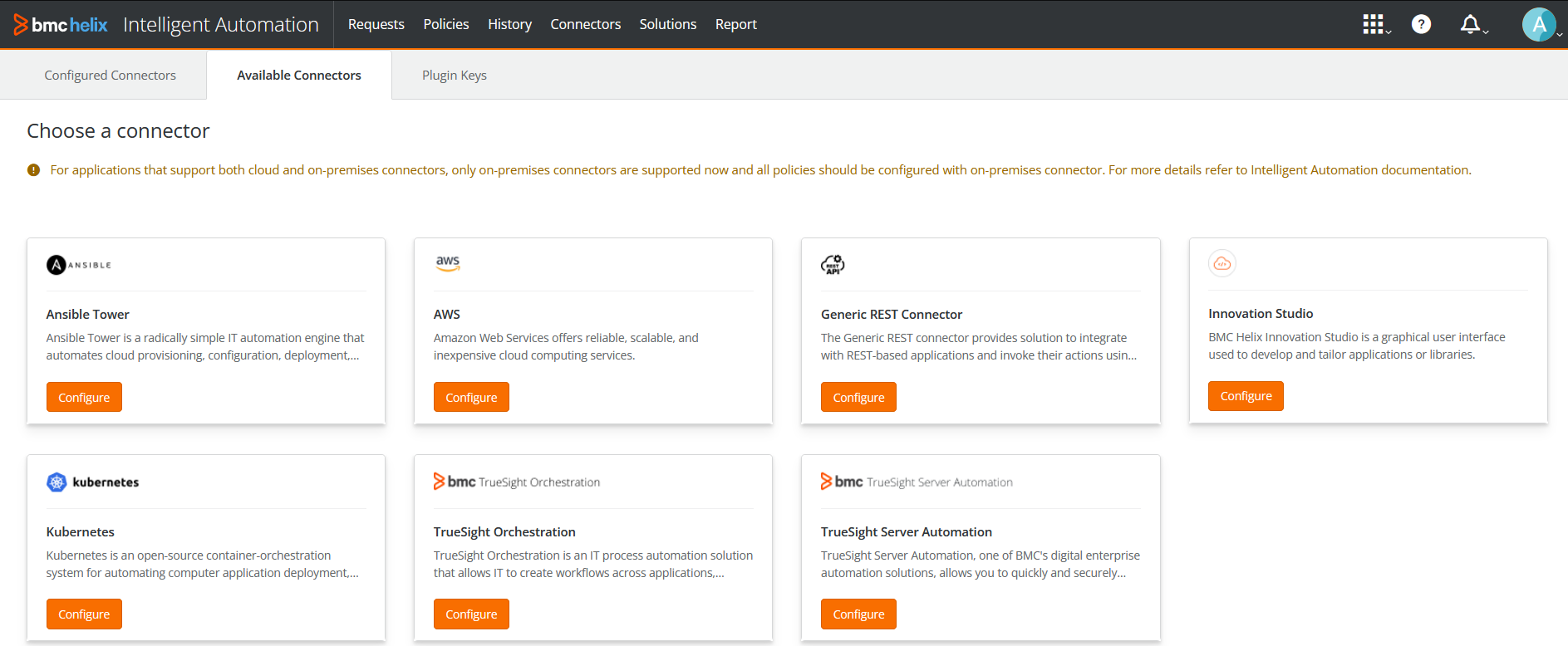
* The file will be called creds.json
* This is your one and only chance to do this. If you don’t, you’ll need to make another key.
  + Save this file somewhere safe so that you can reuse it as needed in the future

Click “Save” once you’re done.



## Setup a Connector

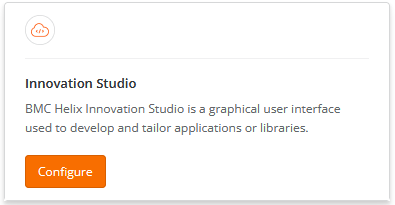
**In HIA open the “Available Connectors” tab**



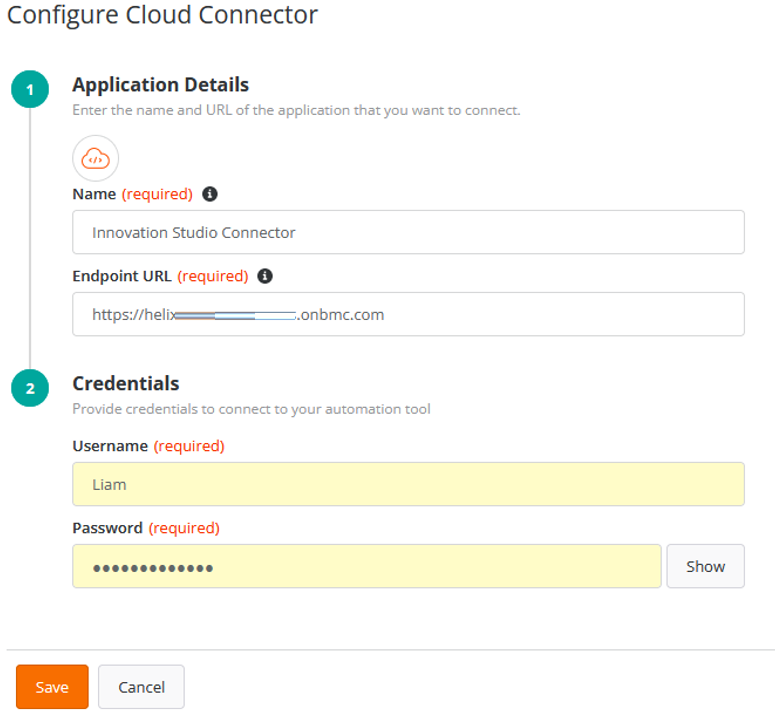
### Cloud Connectors

#### Deploying an Innovation Studio Connector

**Click on the configure button for “Innovation Studio”**

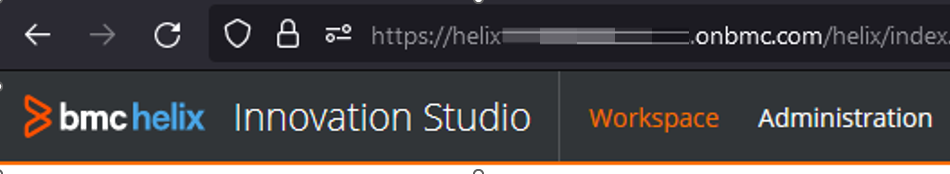


**Fill-out the parameters and click Save**



Guidance:

* Name: Give it a meaningful name
* Endpoint URL: Add Innovation Studio’s domain.
  + If you don’t know what the URL is, go the Helix Portal and open Innovation Studio, and get the domain from the URL that opens:

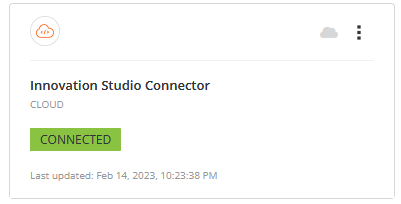


* Username and Password: What HIA will use to authenticate to Innovation Studio
  + This determines what automations that HIA will be able to locate
  + For production environments this would typically be a user only used by HIA.
  + On our demo and PoV environments the user Liam has access to everything in Innovation Studio.

**Verify that the connector is running successfully**

In HIA open the *Connectors->”Configured Connectors”* page.

An Innovation Studio connector should have appeared and have a green “connected” label

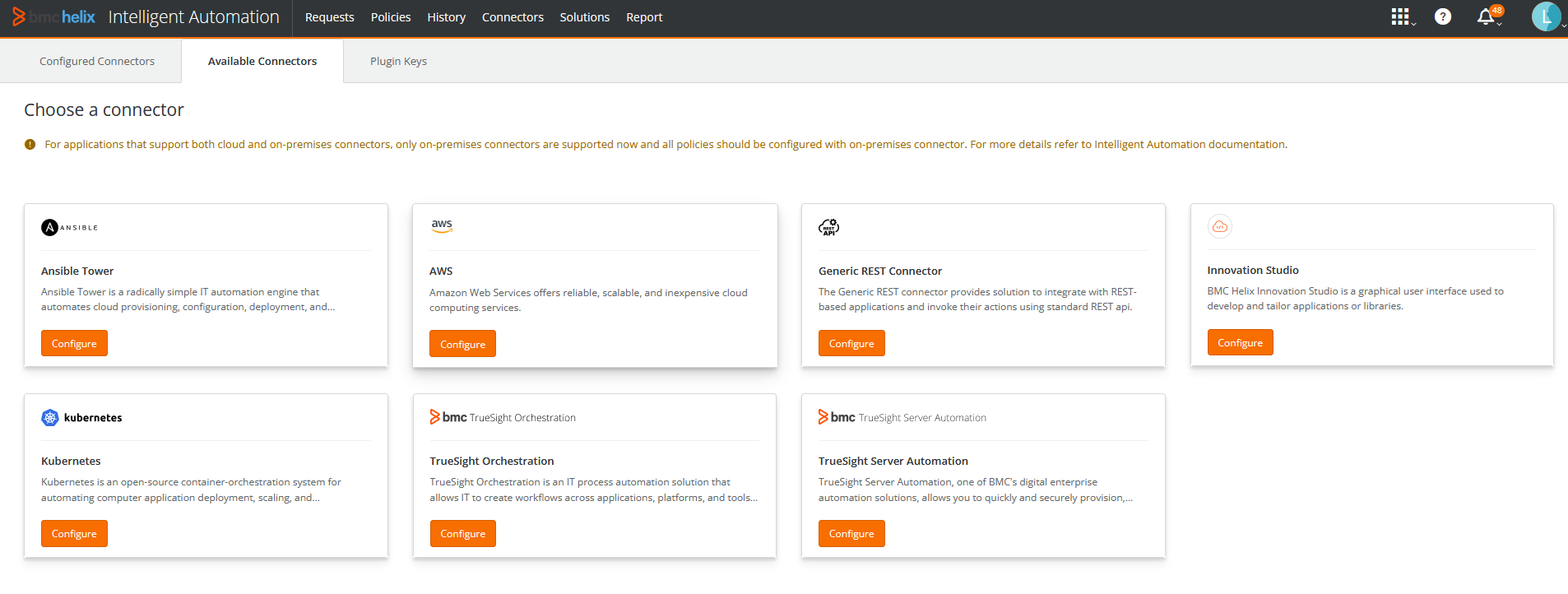


### On-Premises Connectors

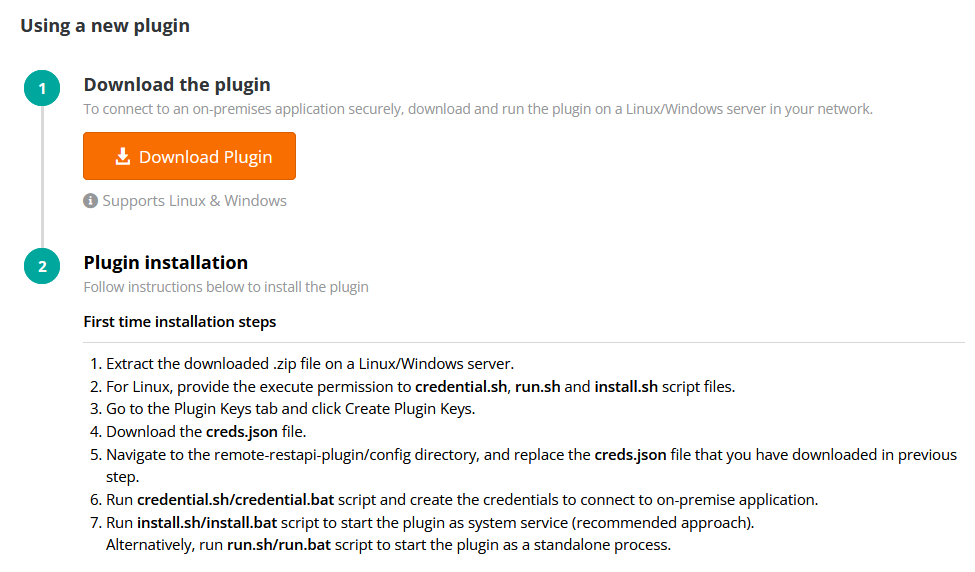
#### Generic Initial Steps

**On the Connectors->Available Connectors tab, click “configure” on the desired on-prem connector**

(it’s pretty much the same binary for all the options)



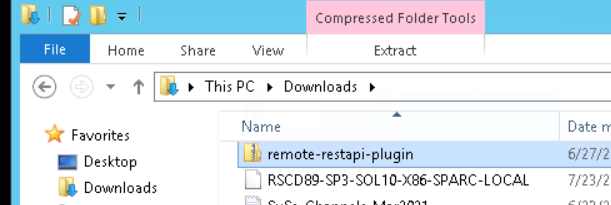
**And then download the plugin:**



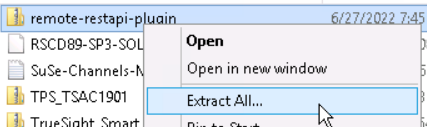
**The zip file needs to be sent to the server that will execute the connector, have its contents extracted.**

#### Deploying a TrueSight Server Automation (TSSA) connector

1. **Copy the remote-restapi-plugin.zip to the server.**

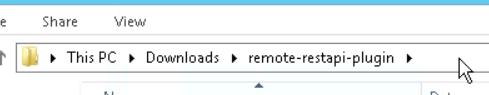


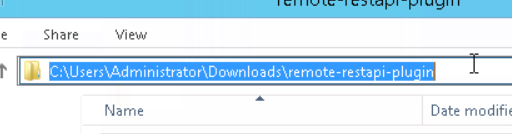
1. **Extract the zip by right clicking and using the “Extract all…” option**

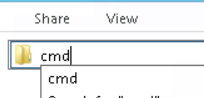


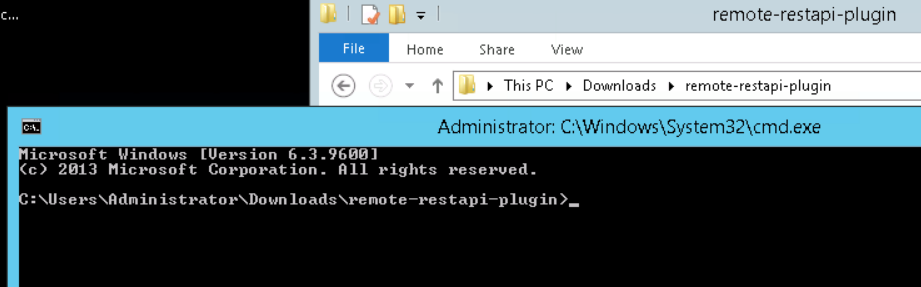
1. **Once the extraction finishes, open the remote-restapi-plugin folder and go to the “config” folder**
2. **Copy the creds.json file, belonging to the Plugin Key you wish to use, to the config folder overwriting the existing creds.json file.**
3. **Go back to the remote-restapi-plugin folder**
4. **Click in the address bar, type cmd, and press enter.**

This will open a command line in the current folder





<press enter>



1. **Configure the plugin to be able to communicate with TSSA:**

We’ll start by creating the folder tree that the plugin will use to store its credential database. Execute in the command line:

mkdir C:\ProgramData\BMC\ia\_remote\_plugins\database

Obs: The ProgramData folder is a hidden folder.

Next, we’ll configure a plugin credential so that it knows how to connect to TSSA:

credential.bat create -n tssa -t TSSA -i https://<IP or FQDN>:9843 -p tssa -a apikey

Very important that you type “https” and not “http” in the URL above.

Replace <IP or FQDN> with the IP or FQDN of you your TSSA installation.

example:

A screen shot of a computer

AI-generated content may be incorrect.

Once you press enter it’ll ask for the information below. Type the answers EXACTLY as defined in TSSA because TSSA is case-sensitive:

username: BLAdmin

password: <the defined BLAdmin password>

role\_name: BLAdmins

1. **Execute the following command:**

install.bat

This will install the plugin, set it to run as a windows service, and to start running.

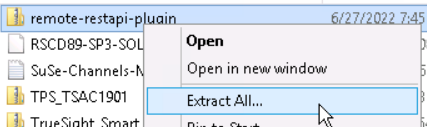
If you don’t have sufficient privileges to manage windows services, you may alternatively use run.bat to execute the connector but you’ll need to leave the command line active, as closing it will abort the connecter.

#### Deploying a Generic Rest Connector

Note that when HIA makes automation calls using REST, it makes the request and then moves on, acting as a stateless automation broker. That means that it won’t keep checking for any additional updates and will assume that the automation ran successfully if the API endpoint responded with a code “200 OK” status message.

If there’s any interest in the automation updating the event after running, sufficient information (e.g. event id) needs to be sent via the automation policy to the REST endpoint, and the automation must be setup to send BHOM updates using REST API calls.

1. **Extract the installer by right clicking and using the “Extract all…” option**



1. **Once the extraction finishes, open the remote-restapi-plugin folder and go to the “config” folder**
2. **Copy the creds.json file, belonging to Plugin Key you wish to use, to the config folder overwriting the existing creds.json file.**

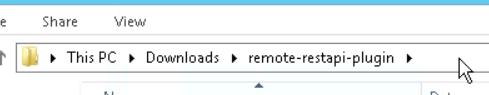
##### Configuring with Control-M

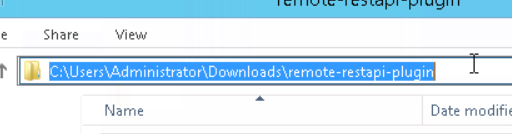
To connect to Control-M, you’ll the following information from the Control-M team:

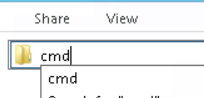
* URL to reach the automation-api (also called Endpoint URL)
  + Exemple: https://<IP or FQDN>:443/automation-api
* API Key that will be used for authentication:
  + Example: UFJER0ZQOmFjMa(...)rSUkwWUl1NW9BU1ZVK0pRRm1ka1NrPQ==

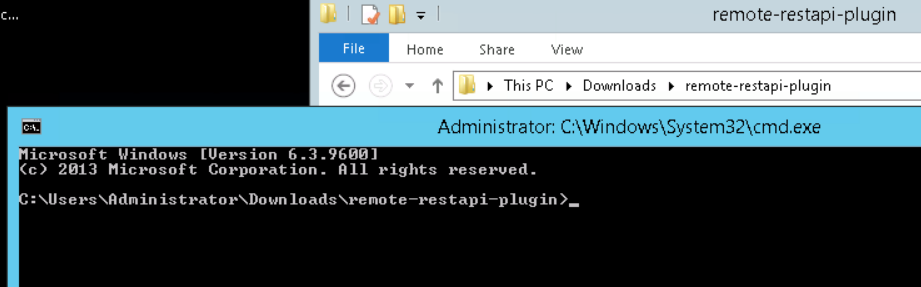
1. **Go back to the remote-restapi-plugin folder**
2. **Click in the address bar, type cmd, and press enter.**

This will open a command line in the current folder





<press enter>



1. **Configure the plugin to be able to communicate with Control-M:**

We’ll start by creating the folder tree that the plugin will use to store its credential database. So execute in the command line:

mkdir C:\ProgramData\BMC\ia\_remote\_plugins\database

Obs: The ProgramData folder is a hidden folder.

Next, we’ll configure a plugin credential so that it knows how to connect to control-m:

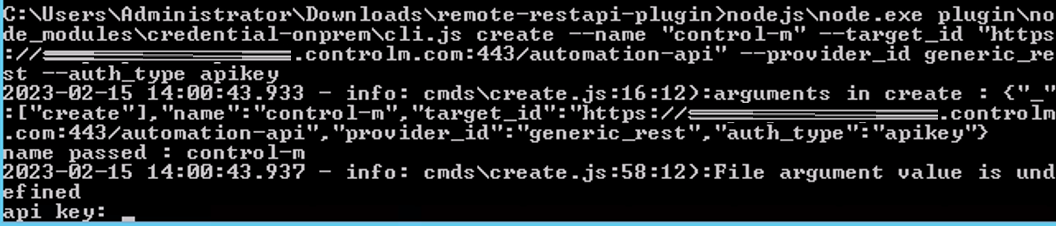
*Obs: Type the values EXACTLY as shown because they are occasionally case-sensitive:*

credential.bat create --name "control-m" --target\_id "[https://<IP or FQDN>:443/automation-api](https://se-preprod-aapi.us1.controlm.com:443/automation-api)" --provider\_id generic\_rest --auth\_type apikey

* Adjust the “target\_id” parameter to use the URL for the Control-M that you want to connect to
  + This will be the Endpoint URL that the connector will always append to the start of automation calls.
* Change the “name” to whatever helps you identify the connector instance locally.
  + Note that when the connectors connects to HIA, it will register with a random name and ignore this parameter.
* Very important that you type “https” and not “http” in the URL above.

Once you press enter it’ll ask for the information below.

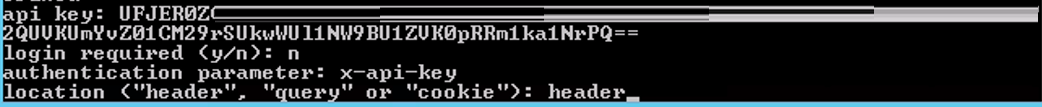
Api key: <use the value you got from the control-m team>



Next, answer “n” for the “login required” question:

For the authentication parameter insert: x-api-key

Type header for the location.



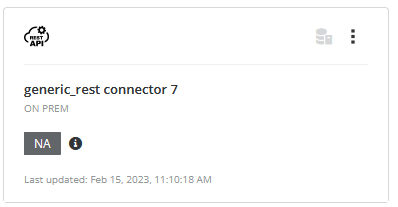
**Type and execute the following command:**

This will install the plugin, set it to run as a windows service, and to start running

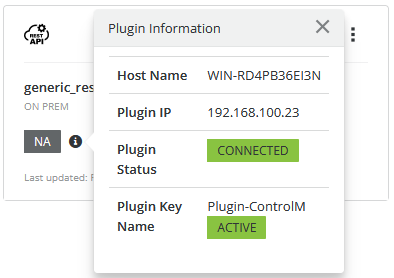
install.bat

**Login to Helix and go to “Intelligent Automation”->Connectors->”Configured Connectors”**

Unlike other connectors, REST APIs ones don’t auto validate if they were correctly configured with the target system, will always show itself with the “NA” status.

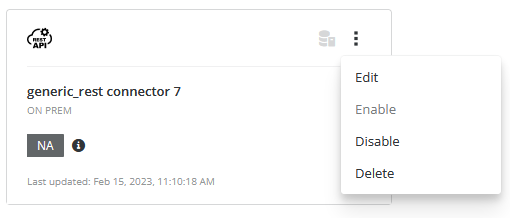


It only checks if the connecter is connected with HIA and ready for use.

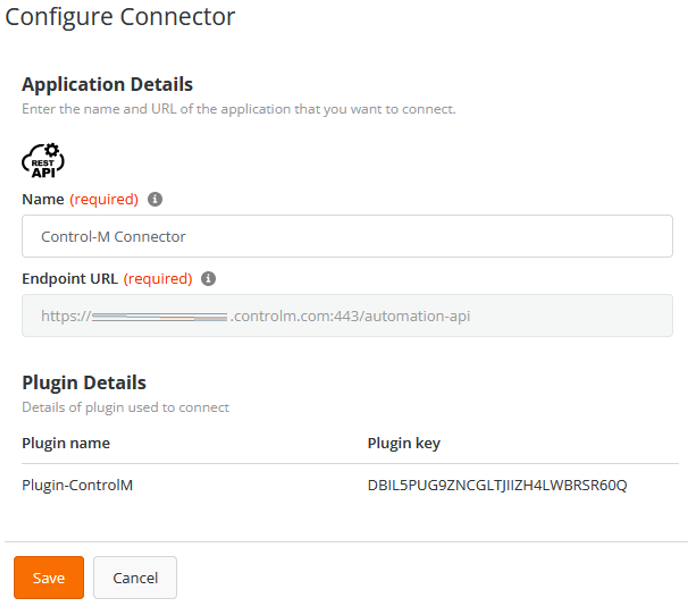


**Rename the Connector to something meaningful**

Click the 3 dots and select edit



Type a new name and click save

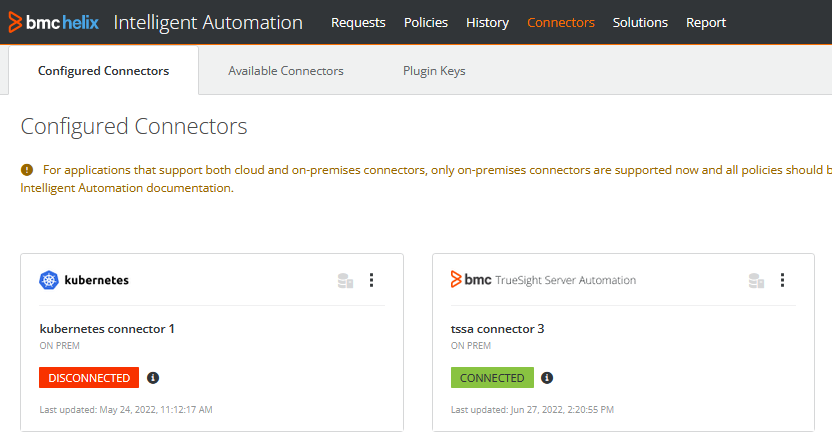


#### Generic Final Steps

##### Verify that the connector is running successfully

1. **Login to Helix and go to “Intelligent Automation”->Connectors->”Configured Connectors”**

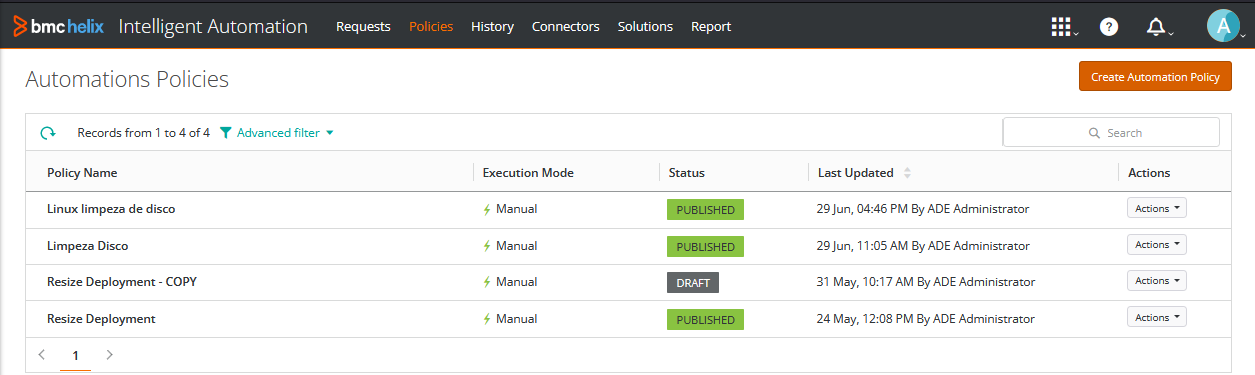
The configured connector should have appeared and have a green “connected” label



## Create Automation Policies

### Generic Inicial Steps

**Click on the “Policies” tab and then “Create Automation Policy”**

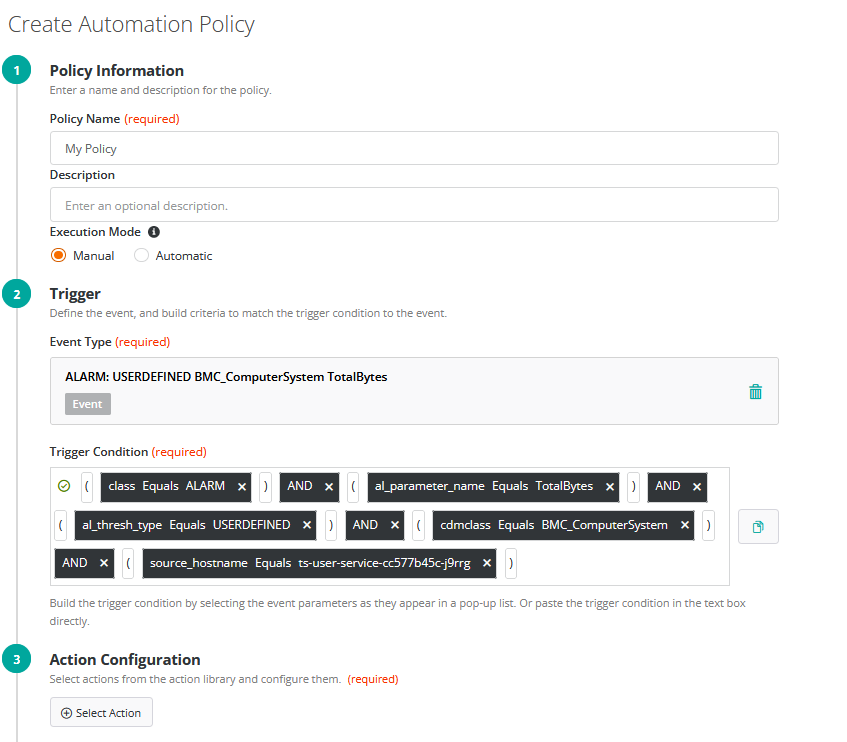


**Configure the policy by defining a name and a trigger**

The trigger is simply a filter of what events are applicable for this automation.

In other words, if the filter matches for an event, then operators can run this automation for the event.

You can define the filter as generic or as specific as you need, and the images below have a random example.



**Click on the “Select Action” to continue the creation process**

### Configuring Actions: Connector Specific

There are 3 steps in defining an action:

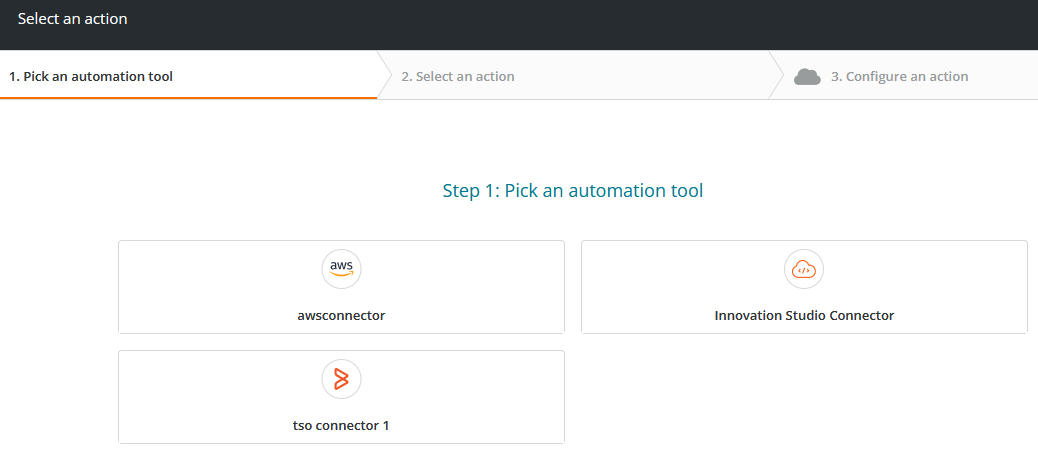
* Choosing a connector: What system will be called;
* Choosing an action: Which automation needs to be executed;
* Configuring the action: Values so that the automation know what to do.
  + In this step you’ll give the action a meaningful name (to make it easier to recognize in HIA) and define what parameters to pass on.
  + In this step there will be a side menu on the right with the list of event attributes you can use as variables. Search and click on whichever one you want to use, and the page will copy the variable name to memory, and then use “ctrl-v” to paste it wherever you need it.
    - See images below for an example:

|  |
| --- |
|  |
|  |
|  |

**Choose the desired connector by clicking on it.**

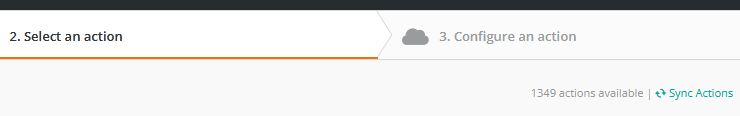
#### Innovation Studio workflows

**These steps are for using a previously configured Innovation Studio connector**



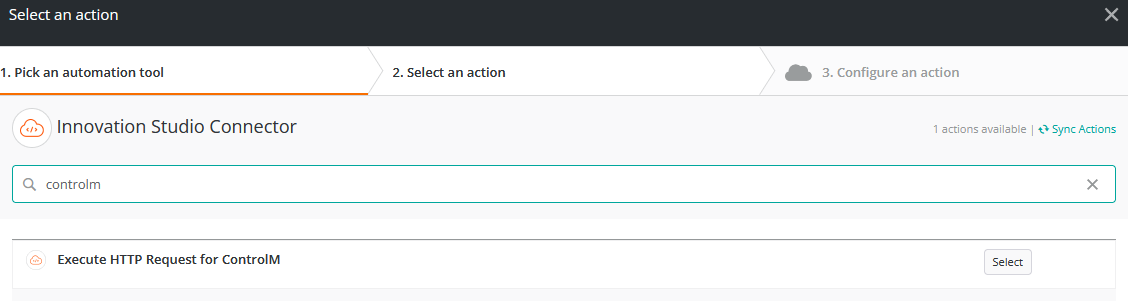
**Once the connect is selected, click the tiny blue “Sync Actions” link in the top right corner**

This will sync Intelligent Automation with the list of automations created in Innovation Studio.

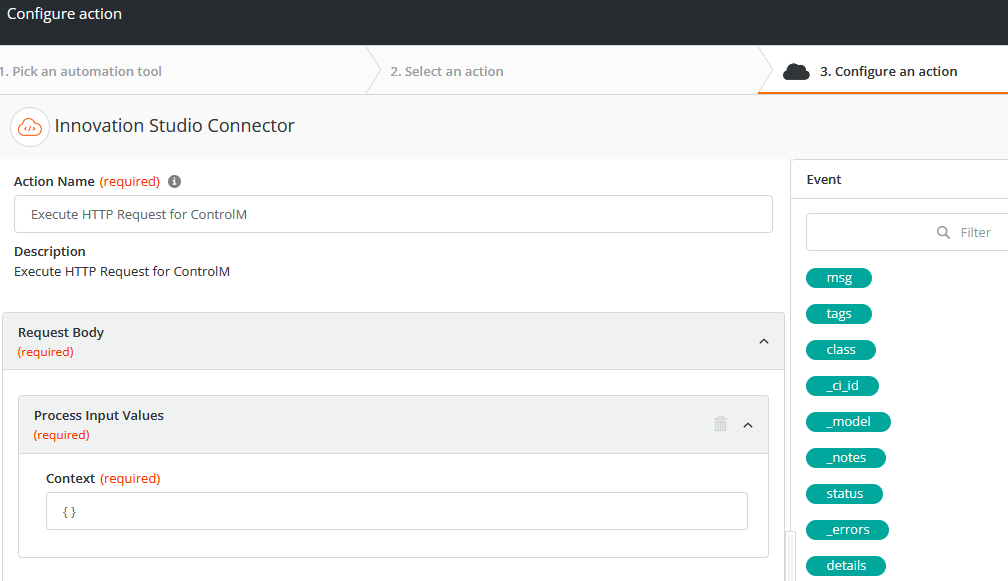


**Once it finishes loading, search for the desired automation and click “Select”.**

*The following screenshots use the* Innovation Studio: Example Workflow that runs a Control-M job*as the selected action.*



**Fill-in any values as needed.**

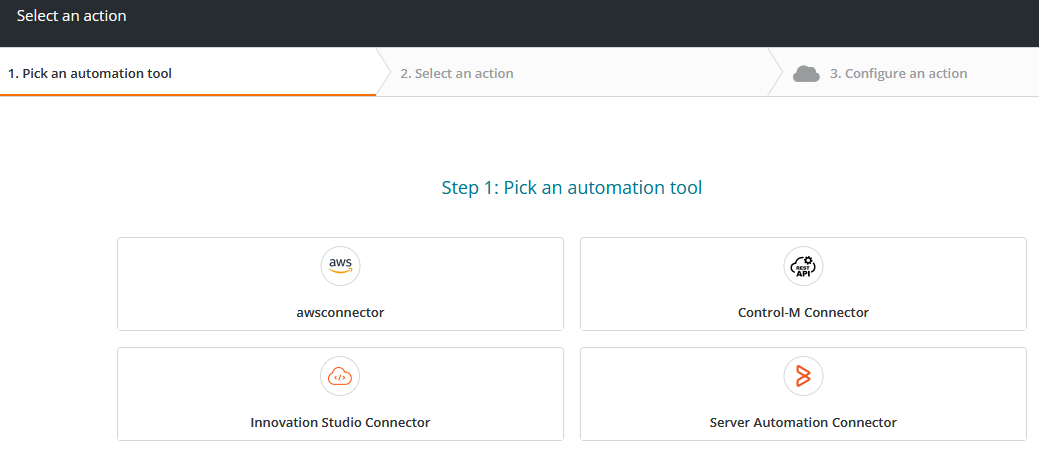


**Click “done”**

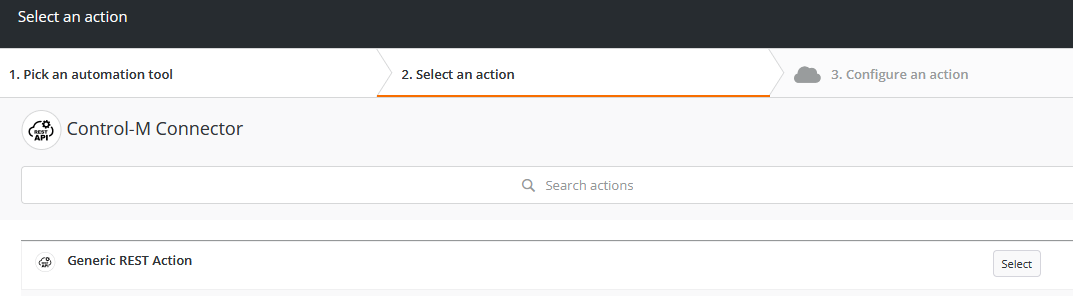
#### Generic Rest Connector

**These steps are for using a previously configured Generic Rest connector**

In the image below the Generic Rest Connector is called “Control-M Connector”



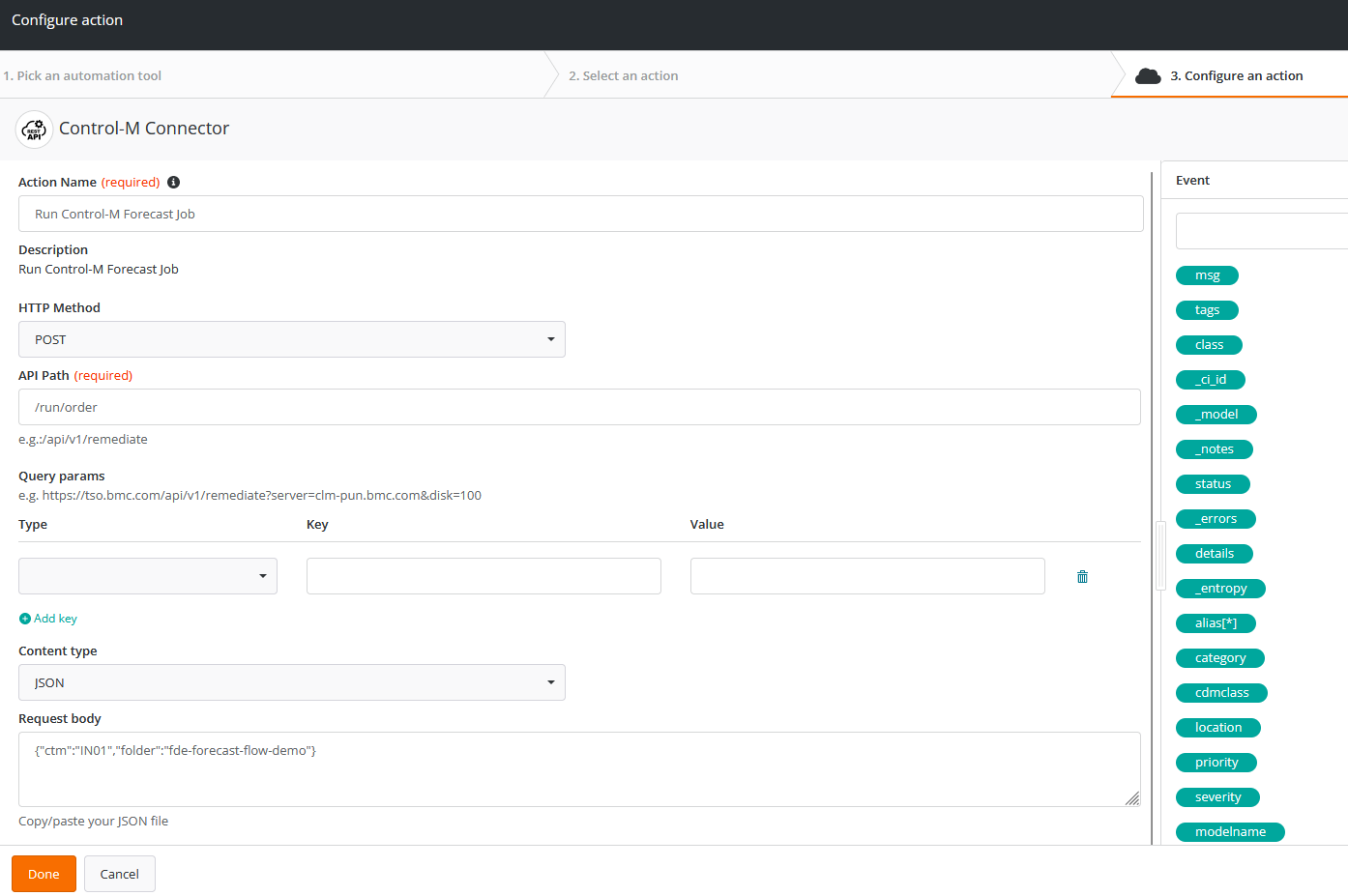
**Once the connect is selected, you’ll be presented with a list containing a single action. Click the “select” button for it.**



**Give the action a meaningful name & configure the parameters to make the appropriate REST API Call.**

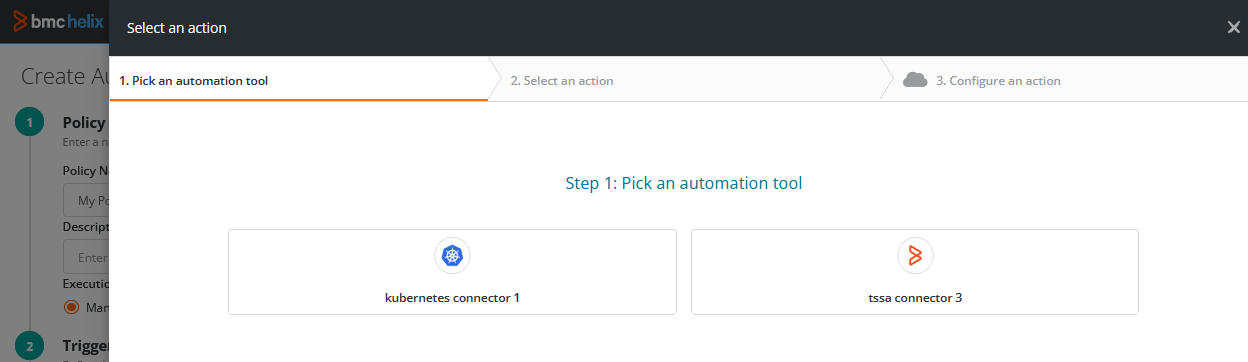
In the print below is an example of how HIA can call Helix Control-M to run a job.

Note that the API Path is appended to the end of whatever was configured as the connector’s Endpoint Url. Generic Rest Connector: How to locate the configured Endpoint URL.



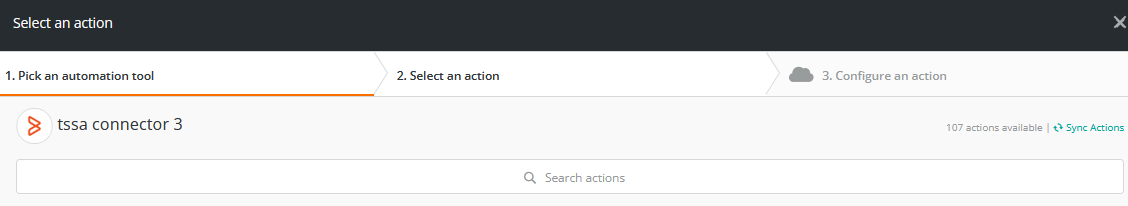
#### TSSA automations

**These steps are for using a previously configured TSSA connector**

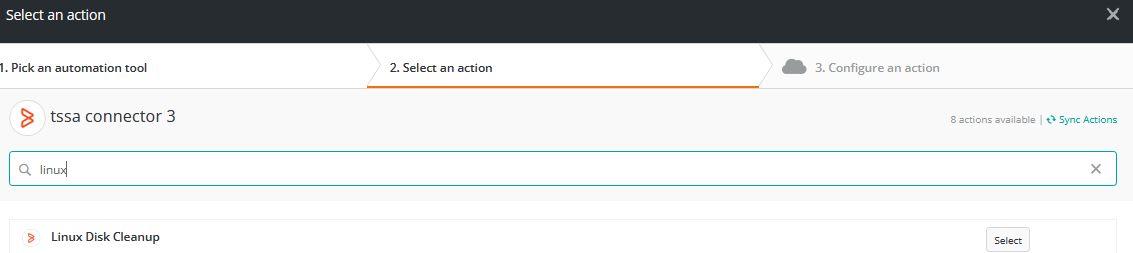


**Once the connect is selected, click the tiny blue “Sync Actions” link in the top right corner**

This will sync Intelligent Automation with the list of automation jobs already configured in TSSA.



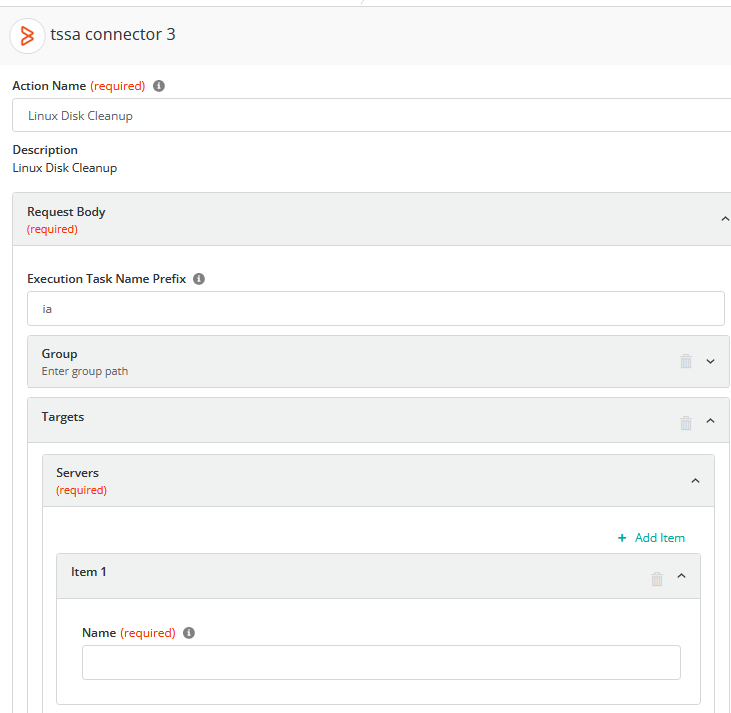
**Once it finishes loading, search for “Linux Disk Cleanup” and click “Select”**



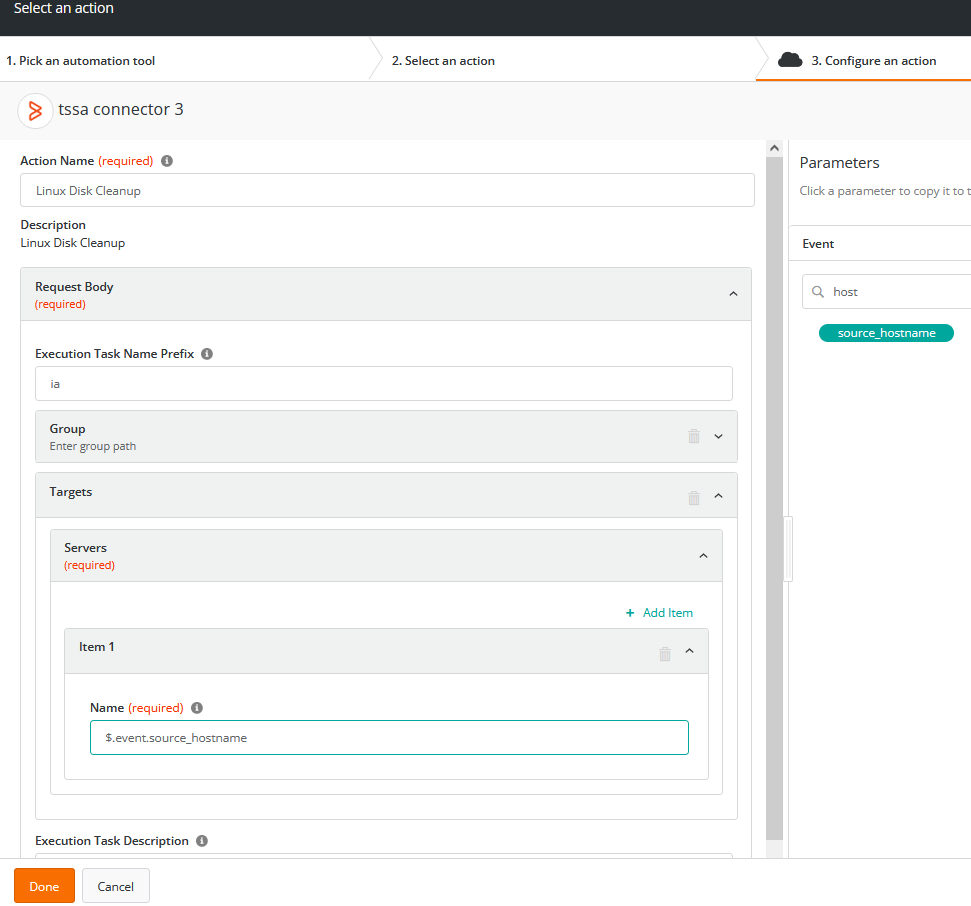
**Expand the menus to locate the field Targets->”Item 1”->Name**

This is the name of the target host.

In the following steps we’ll configure this field to receive the value of the event’s hostname.



**In the menu to the right search for “source\_hostname”. Click it to copy the variable memory to memory, then click in the “Name” field you found in step #27 and type “Ctrl-V”**

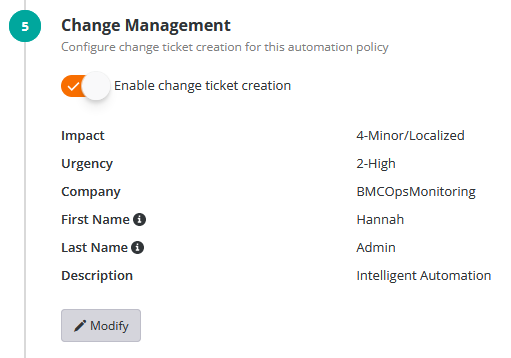


**Click “done”**

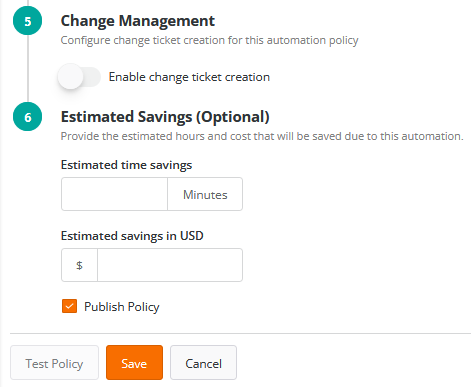
### Generic Final Steps

**Enable or Disable “change management” if you want to demonstrate its usage**

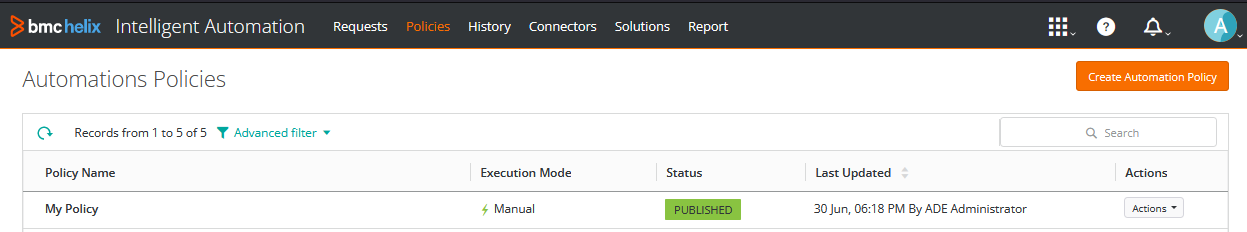
Example of how it can be configured:



1. **Select “Publish Policy” and click save**



1. **Your policy is now ready to be used for automations in BHOM!**



# Advisories

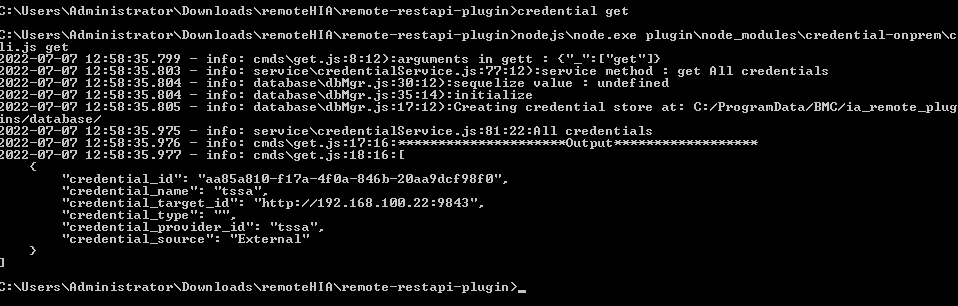
## Generic Information

### How to delete a plugin credential

To delete the credential, execute:

credential.bat get

Copy the “credential\_id” value

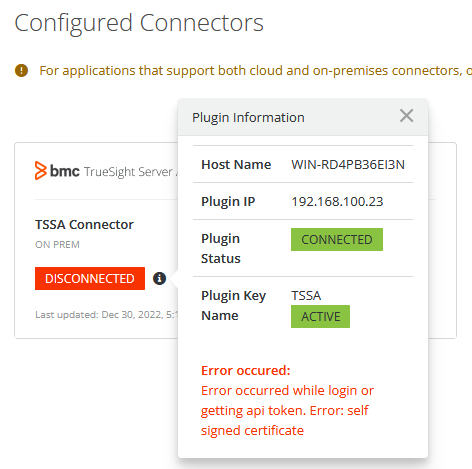


And then execute:

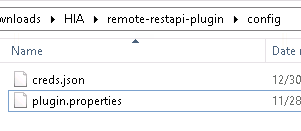
credential.bat delete -i <insert the “credential\_id” value here>

### Connector failing to connect due to Self Signed Certificate

If HIA reports that the connector is failing to connect to your application due to a self-signed certificate error (see image below):

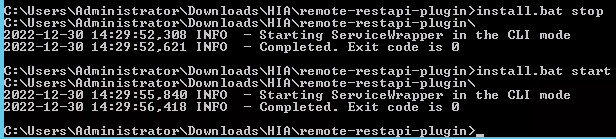


You’ll need to edit the plugin.properties configuration file located in “remote-restapi-plugin/config” folder and change the parameter config.providerCertValidation from “true” to “false”:





And then use the “install.bat” command to restart the connector:



## Connector Specific

### TSSA: How HIA interprets job executions

Note that when you run a TSSA automation, TSSA expects to receive a hostname of a server that it is currently managing (has an agent installed and added to TSSA’s list of servers).

Otherwise, the job execution will either fail or run the job successfully with warnings that the target wasn’t found and nothing was actually done, depending on how the job was setup in TSSA.

HIA will interpret any TSSA’s status of “Successful run but with warnings of target not found” as simply being “Successful” and you will not find the warnings in HIA.

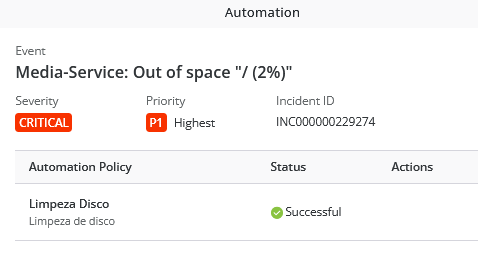
#### Example: “Linux Disk Cleanup” job in TSSA

This job simply runs a shell script that deletes all files in the /tmp folder (temporary files) of a Linux server.

Because of how HIA interprets job results and doesn’t handle warnings, and how the “Linux Disk Cleanup” job is configured in TSSA, you’re unlikely to ever get a FAILED message in HIA unless there’s a communication problem between Helix and TSSA.

So, it doesn’t matter if you run the job providing TSSA with a hostname that it manages or not, the job will always end up with a “Successful” status in HIA.

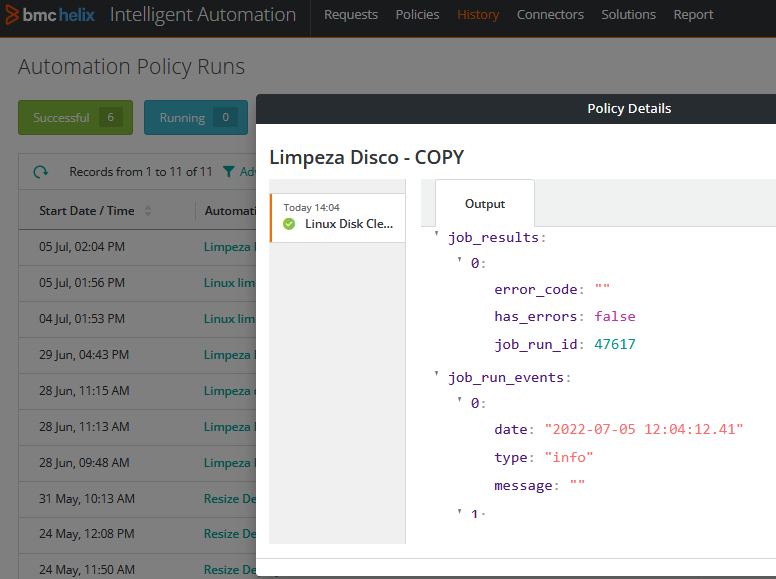
For a simple Helix automation execution demo, in which you won’t go into the details and logs of the automation engine (TSSA in this case) and just want to demonstrate in Helix a green successful execution result in a few seconds, this is perfect!



Now, if you do provide the “Linux Disk Cleanup” job with a valid TSSA managed hostname as a target, then it will actually do something. And the following list of hostnames can be used as valid targets, since TSSA is provisioned in BDC with them already configured: *buildserver, bl-rhwww, rhwww64* (they’re all the same VM). You can run the job as many times you want on these servers.

To know through HIA if the job actually did something (found a target), currently (@2022-07-05), the only way is to check HIA’s history page and search the policy’s run output. If in the output you find the “job\_results” and “job\_run\_events” sections, then you’ll know that the job found a valid target.

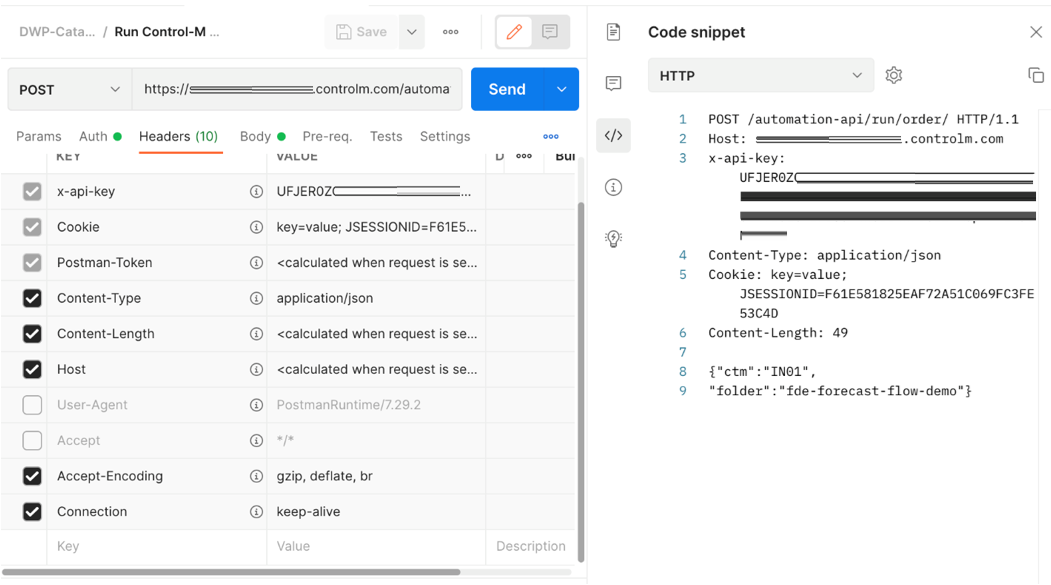
Example:



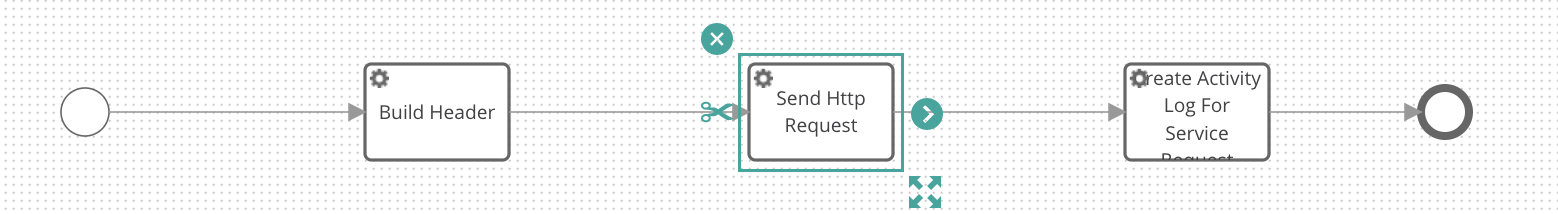
### Innovation Studio: Example Workflow that runs a Control-M job

**CURL example to run a control-M job via CLI:** curl -X POST -H "x-api-key: "**<API-KEY>**"" --header "Content-Type: application/json" --header "Accept: application/json" -d "{  \"ctm\": \"IN01\",  \"folder\": \"fde-forecast-flow-demo\"}]}" https://<IP or FQDN>.controlm.com/automation-api/run/order

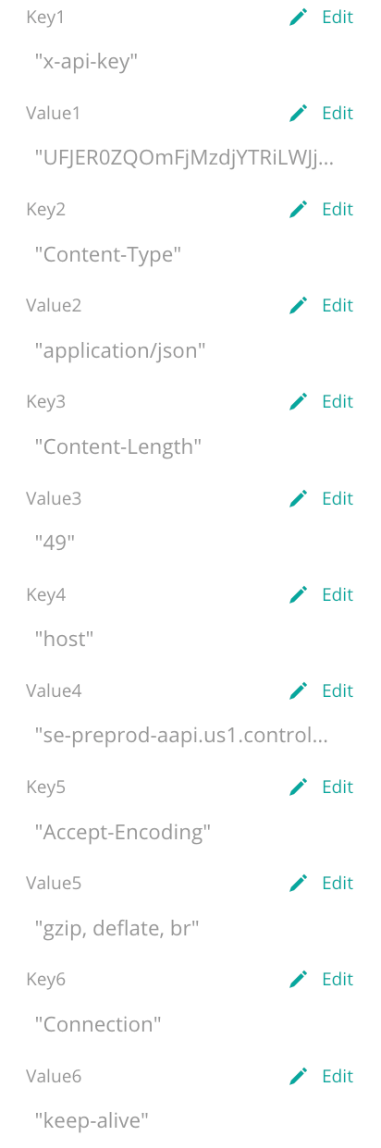
**How the same call looks like in Postman:**



**How the workflow can be constructed in DWP-C:**

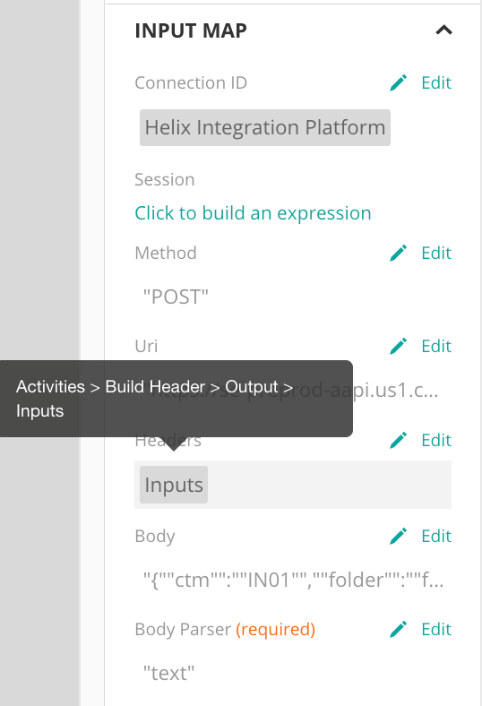


How the “Build Headers” step was configured in its Build Input Set:





How the “Sent HTTP Request Step” was configured:



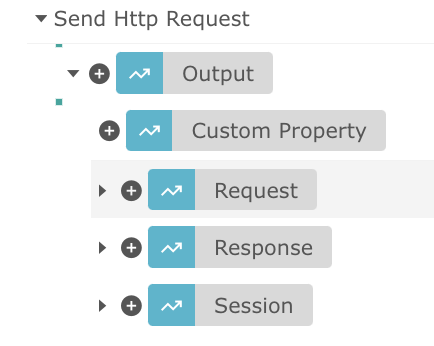
**Uri =** "https://<IP or FQDN>.controlm.com/automation-api/run/order"

**Body =** "{""ctm"":""IN01"",""folder"":""fde-forecast-flow-demo""}"

All quotes are double quotes. DWP does not recognize \” or other ways to putting quotes within quotes.

The body example above has a hardcoded job name and ideally should be changed to a variable, but can be used hardcoded so that HIA can pass on whatever it wants for demonstration purposes.

What the execution outputs look like in DWP:



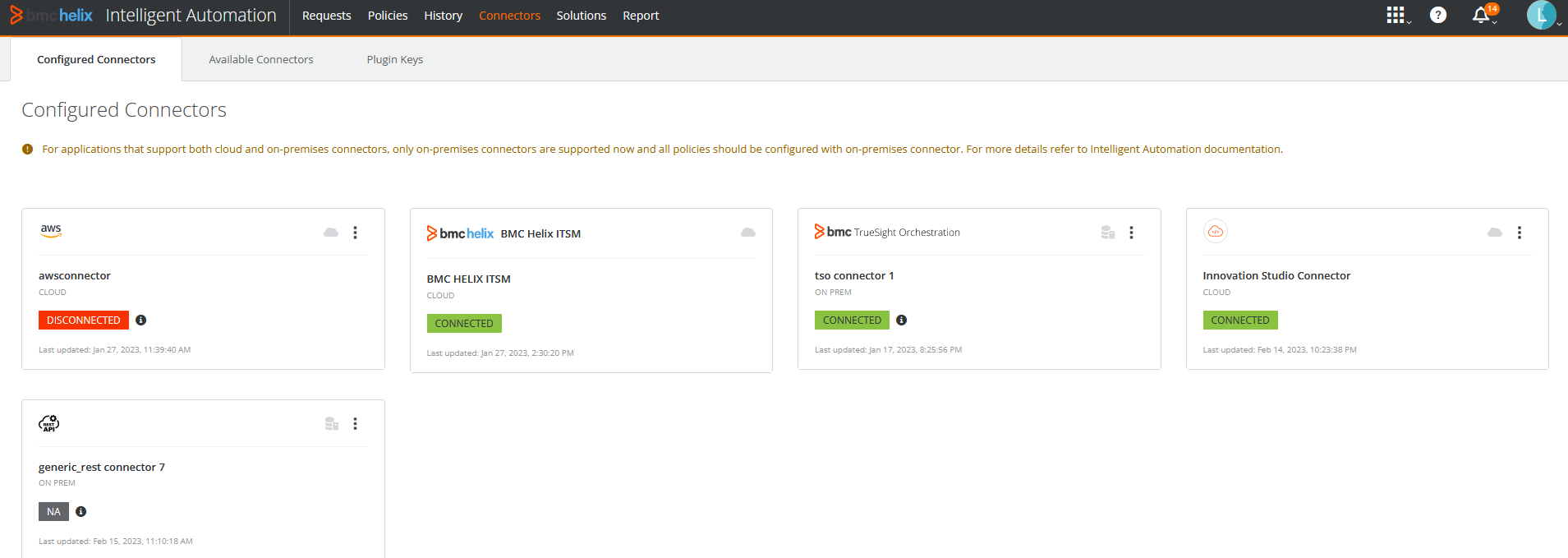
**REQUEST:** {"method":"POST","uri":"https://**<IP or FQDN>**.controlm.com/automation-api/run/order","headers":[{"key":"x-api-key","value":"**<API-KEY>**"},{"key":"Content-Type","value":"application/json"},{"key":"Content-Length","value":"49"},{"key":"host","value":"[<IP or FQDN>.controlm.com](http://se-preprod-aapi.us1.controlm.com/)"},{"key":"Accept-Encoding","value":"gzip, deflate, br"},{"key":"Connection","value":"keep-alive"},{"key":"Accept","value":"\*/\*"}],"body":{"ctm":"IN01","folder":"fde-forecast-flow-demo"}}  
  
**RESPONSE:** {"status":200,"statusReason":"OK","headers":[{"key":"x-amzn-Remapped-Date","value":"Wed, 07 Sep 2022 13:46:38 GMT"},{"key":"x-amz-apigw-id","value":"YF4NnFC0vHcF1bw="},{"key":"vary","value":"accept-encoding,origin,accept-encoding"},{"key":"X-Content-Type-Options","value":"nosniff"},{"key":"Connection","value":"keep-alive"},{"key":"Pragma","value":"no-cache"},{"key":"x-amzn-Remapped-Server","value":"Web Server"},{"key":"Date","value":"Wed, 07 Sep 2022 13:46:38 GMT"},{"key":"X-Frame-Options","value":"SAMEORIGIN"},{"key":"Referrer-Policy","value":"no-referrer"},{"key":"Strict-Transport-Security","value":"max-age=31536000; includeSubDomains; preload"},{"key":"Cache-Control","value":"no-cache, no-store, private, must-revalidate, max-age=0, no-transform"},{"key":"x-amzn-Remapped-Connection","value":"keep-alive"},{"key":"hstsMaxAgeSeconds","value":"31536000"},{"key":"Content-Security-Policy","value":"object-src 'none'; base-uri 'self';plugin-types application/x-java-applet; sandbox allow-scripts allow-popups allow-forms allow-top-navigation allow-presentation allow-same-origin allow-downloads; report-uri /csp-violation-report-endpoint/; form-action 'self'; frame-ancestors 'none';"},{"key":"Content-Encoding","value":"gzip"},{"key":"x-amzn-RequestId","value":"779e4a6e-13a0-49cb-a15d-e21c11f4ac93"},{"key":"Set-Cookie","value":"JSESSIONID=DE7F82C12BDFEA8DDBD9D; Path=/automation-api; Secure; HttpOnly"},{"key":"X-XSS-Protection","value":"1; mode=block"},{"key":"Content-Length","value":"151"},{"key":"Expect-CT","value":"max-age=120, enforce"},{"key":"Content-Type","value":"application/json"}],"body":"\u001F�\b\u0000\u0000\u0000\u0000\u0000\u0000\u0000��K\u000E� \u0014\u0000�=\u0005a��(M�\u001Btk�\u0001@ 6)��ce��U/�n63��\u0018���r�0��h��\u0014Lb�p���Y\u0018\u0007V\t�\u001C�y4��q\*ij�v]�ރ(�\u0005�:���,����\u000E�=E\*i\u000F\u0007\u0004ԍRд�\b:ox��\u0017¿���\r\u0000\u0000��\u0003\u0000Ӥ]Ͱ\u0000\u0000\u0000"}  
  
**SESSION:** {"cookies":[{"name":"JSESSIONID","value":"DE7F82C12B1B90DBD9D","path":"/automation-api","domain":"[<IP or FQDN>.controlm.com](http://se-preprod-aapi.us1.controlm.com/)","version":1}]}

**For instructions on how to configure a HIA policy to use the workflow above, click here: Innovation Studio workflows**

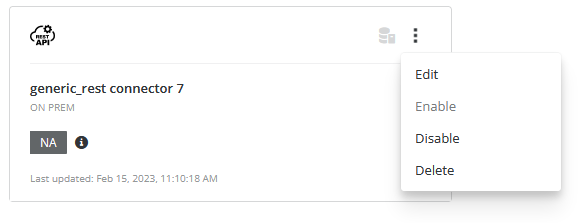
### Generic Rest Connector: How to locate the configured Endpoint URL

The endpoint URL is always appended to the start of whatever API Path is used by actions.

To check what is currently configured, open HIA and go to the Connectors->Configured Connectors tab.



And then click the corresponding 3 dots for the desired connector and choose “edit”



The page that loads will show what the Endpoint URL was configured as:

