Rapid Pilot Program 2024  
Day 3 – Enhanced Integration

Exercises for SAP Integration Suite

Contents

[Goals 2](#_Toc177034194)

[Prerequisites 2](#_Toc177034195)

[Prepare your Integration Suite tenant 3](#_Toc177034196)

[**Download and import the template IFlow package** 3](#_Toc177034197)

[**Importing the official SAP Advanced Event Mesh Adapter** 3](#_Toc177034198)

[Scenario 1: “From Scratch” 5](#_Toc177034199)

[**Queue Setup** 5](#_Toc177034200)

[**IFlow Creation** 10](#_Toc177034201)

[Scenario 2: “Dead Message Queue” 14](#_Toc177034202)

[Automated AEM Broker Setup via APIs and CI/CD (optional) 19](#_Toc177034203)

[Scenario 3: “Sales Order E-Mail Notification” 21](#_Toc177034204)

[**Setup/Configure Dependency Services** 21](#_Toc177034205)

[**Queue Setup** 21](#_Toc177034206)

[**IFlow Setup** 22](#_Toc177034207)

[Scenario 4: “Business Partner Address Check” (optional) 24](#_Toc177034208)

[**Setup/Configure Dependency Services** 24](#_Toc177034209)

[**Queue Setup** 25](#_Toc177034210)

[**IFlow Setup** 26](#_Toc177034211)

[Scenario 5: “SFTP as Legacy Integration” (optional) 29](#_Toc177034212)

[**Setup/Configure Dependency Services** 29](#_Toc177034213)

[**Queue Setup** 29](#_Toc177034214)

[**IFlow Setup** 30](#_Toc177034215)

[Security and fine-grained access control to topics (optional) 33](#_Toc177034216)

[**Experimenting publishing and subscribing to protected topics** 33](#_Toc177034217)

[**Broker Topic ACLs** 34](#_Toc177034218)

# Goals

* Configuring an AEM brokers' queues and topic subscriptions
* Event enabling integration flows and connecting them to AEM brokers to create event-driven integration flows
* How to use the broker's config APIs to automate configuration and enable CI/CD pipelines
* Fine-grained security access in AEM

# Prerequisites

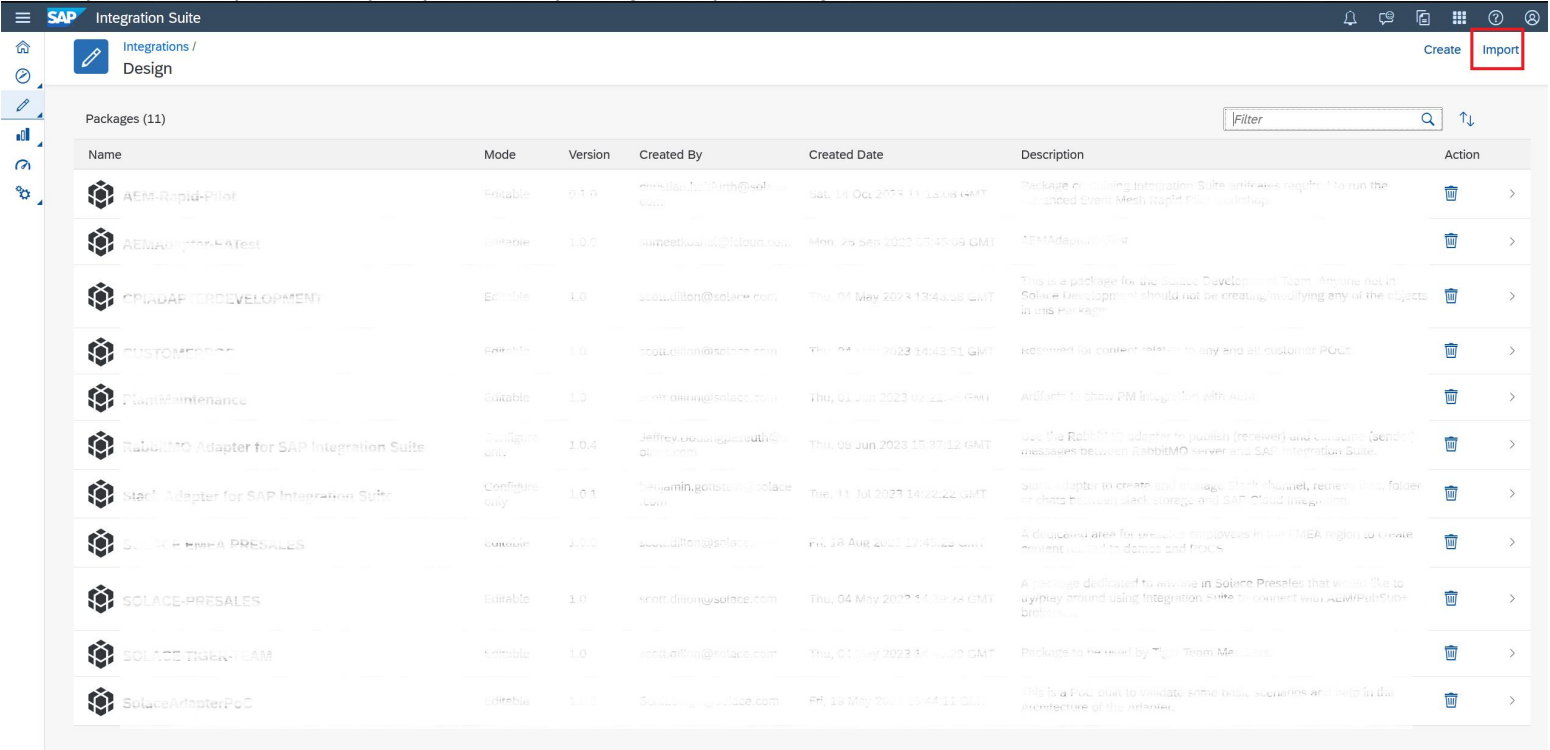
* Complete all activities in day 1 & 2  
  You access and use the same broker you setup previously as well as the simulator to push events for testing
* Have access to an active Integration Suite/Cloud Integration tenant
* Have an SFTP server and account credentials if you want to test successful integration of events to a file based interface of a legacy system (optional)

# Prepare your Integration Suite tenant

### **Download and import the template IFlow package**

Download Day3\_Content.zip

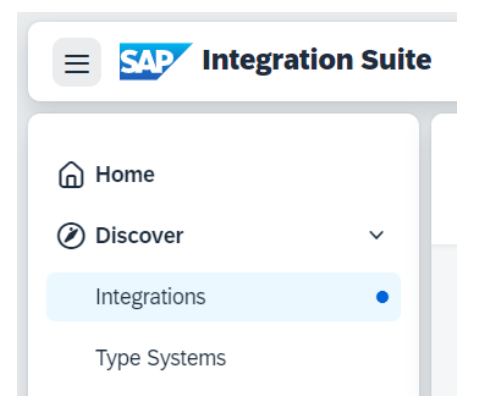
Import the downloaded file as a new package into you Integration Suite tenant:



### **Importing the official SAP Advanced Event Mesh Adapter**

The Advanced Event Mesh specific adapter was made available in January 2024. If you haven't used this adapter in your Integration Suite tenant before, you may need to import it once. Follow these steps to get the official adapter from SAP

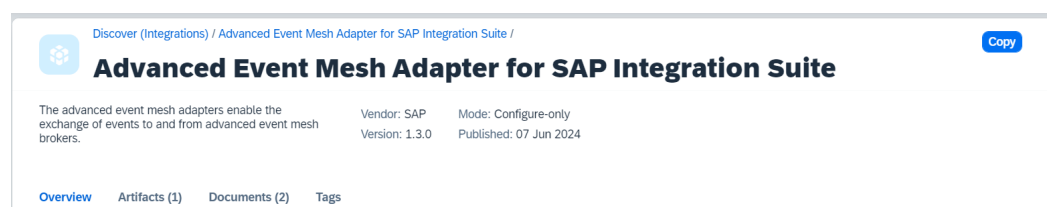
* Navigate to “Discover” -> “Integrations” in the left hand menu:



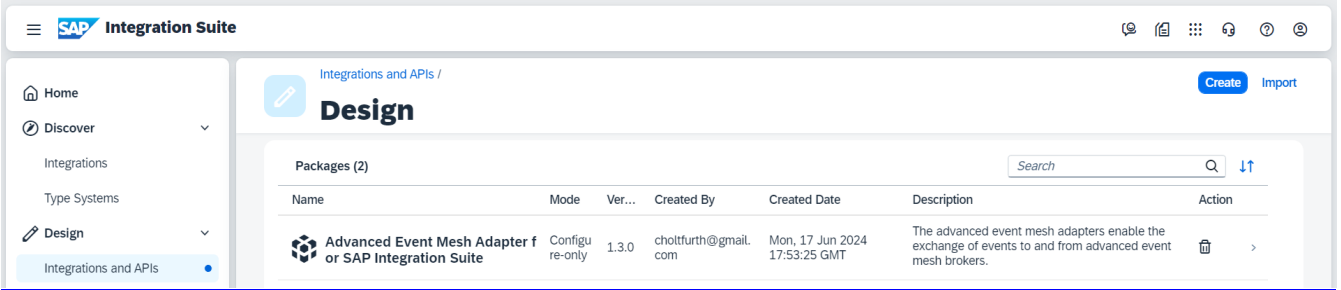
* Search for “advanced” to find the “Advanced Event Mesh Adapter for SAP Integration Suite:



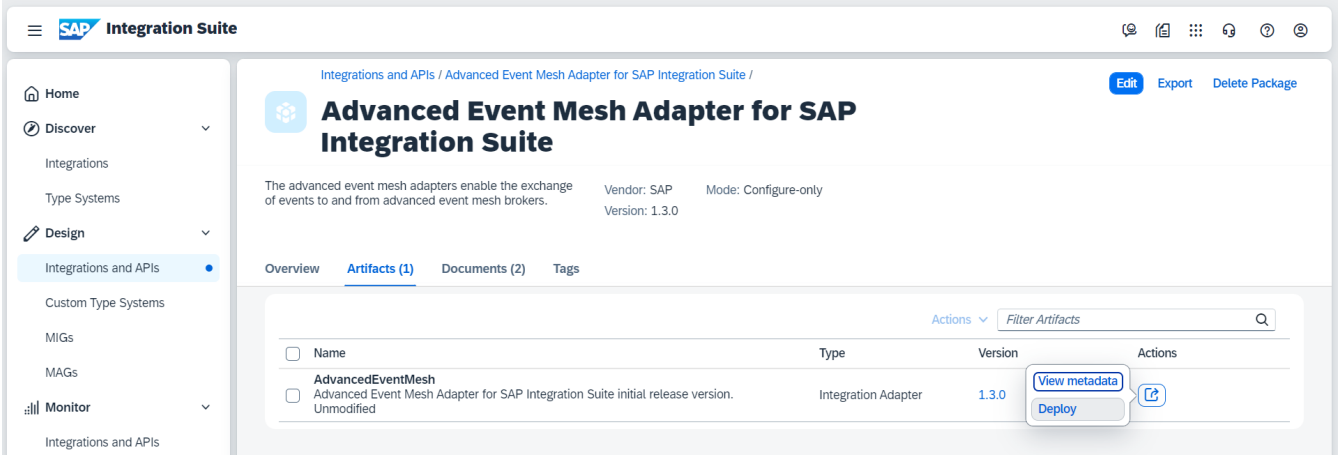
* Select the adapter package by clicking on it, then click on “Copy” on the top right.



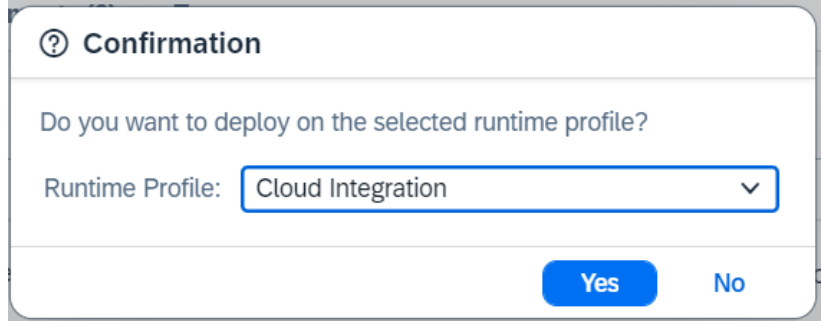
* Go to “Design” -> “Integrations and APIs” in the left hand menu and click on the newly created “Advanced Event Mesh Adapter for SAP Integration Suite” package



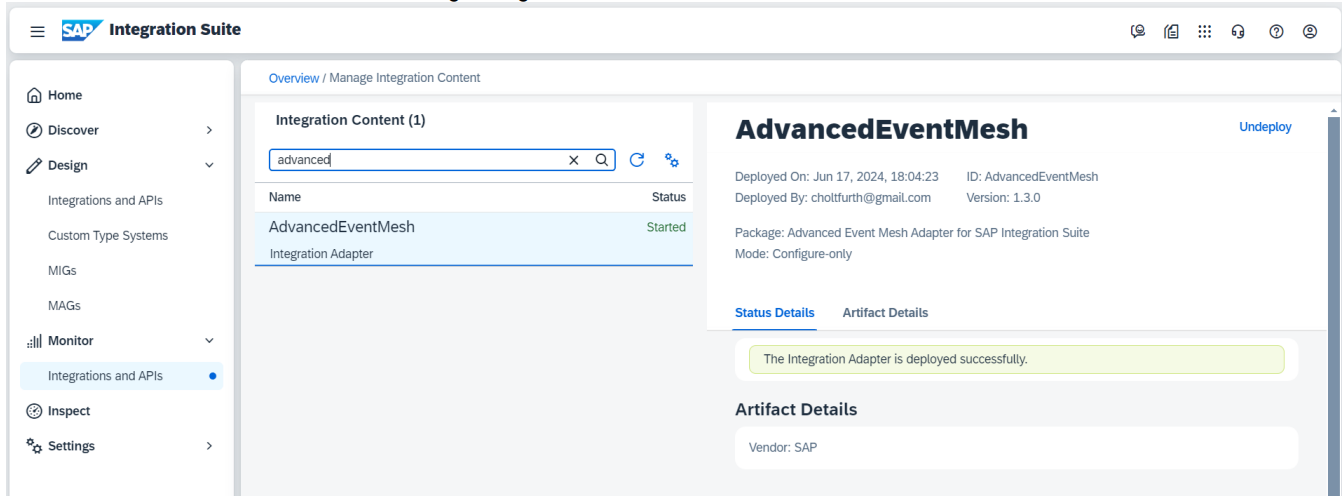
* Navigate to “Artifacts” to see the Integration Adapter and click on and select “Deploy” from the “Actions” menu



* Select the “Cloud Integration” Runtime Profile



* You should now be able to see the AdvancedEventMesh Integration Adapter if you navigate to “Monitor” -> “Integrations and APIs” and click on the tile “All” under “Manage Integration Content”



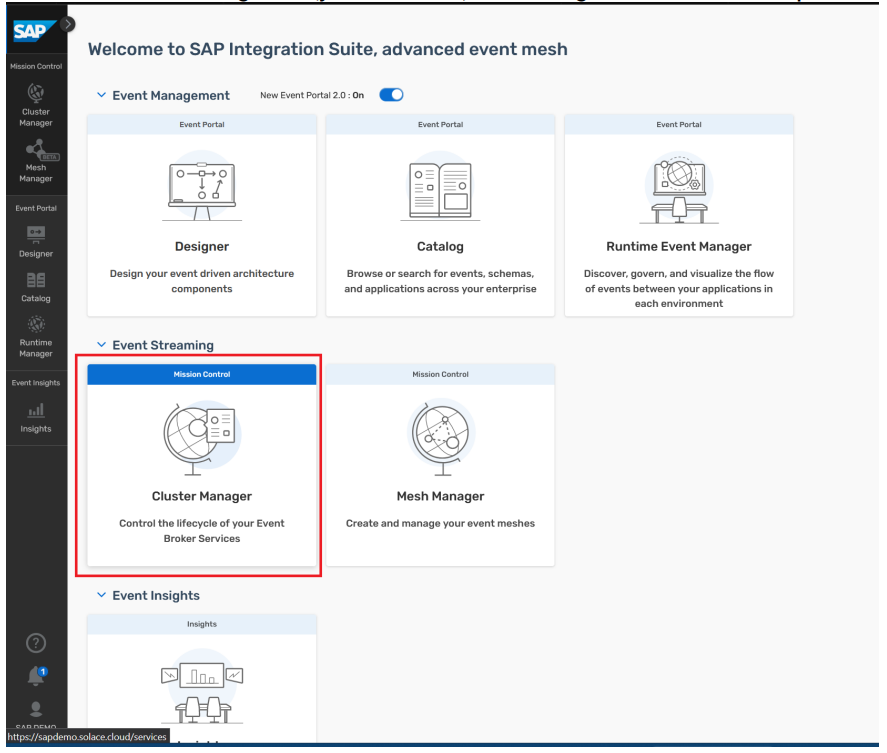
# Scenario 1: “From Scratch”

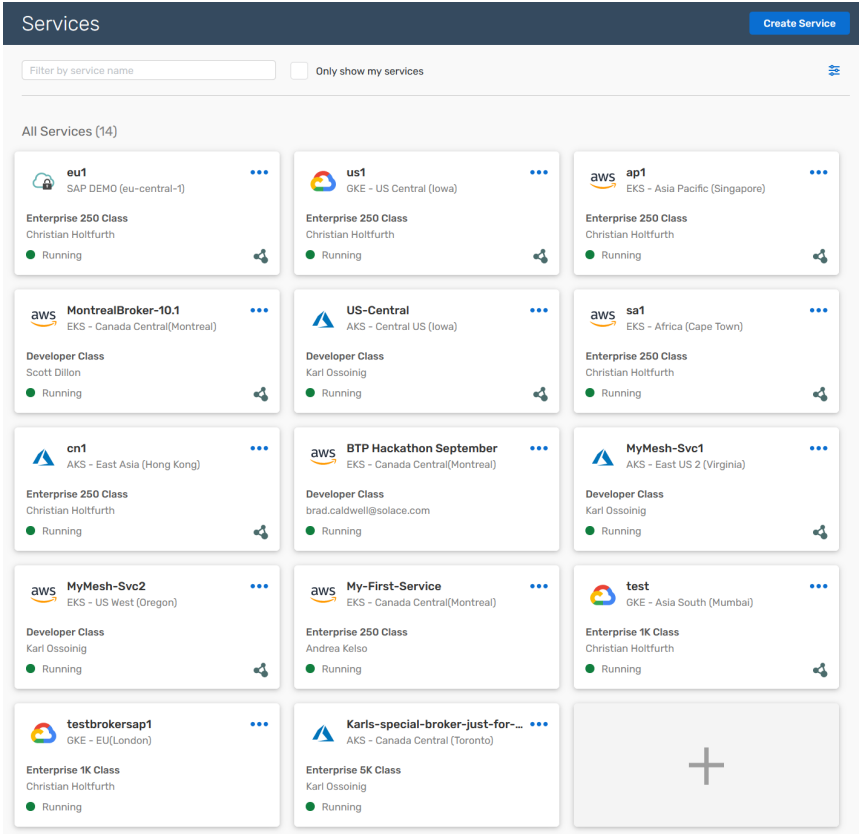
In this section we will perform the necessary steps to create an event driven IFlow from scratch

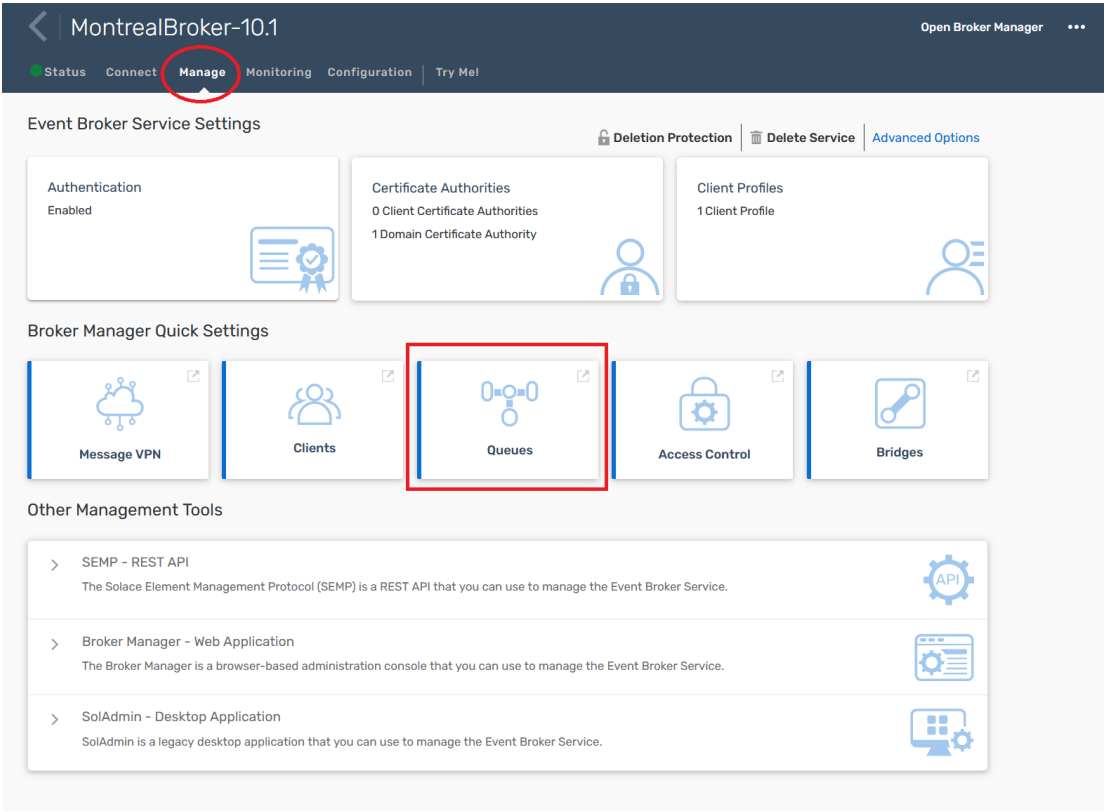
### **Queue Setup**

First, we will create the required input queue for the integration flow.

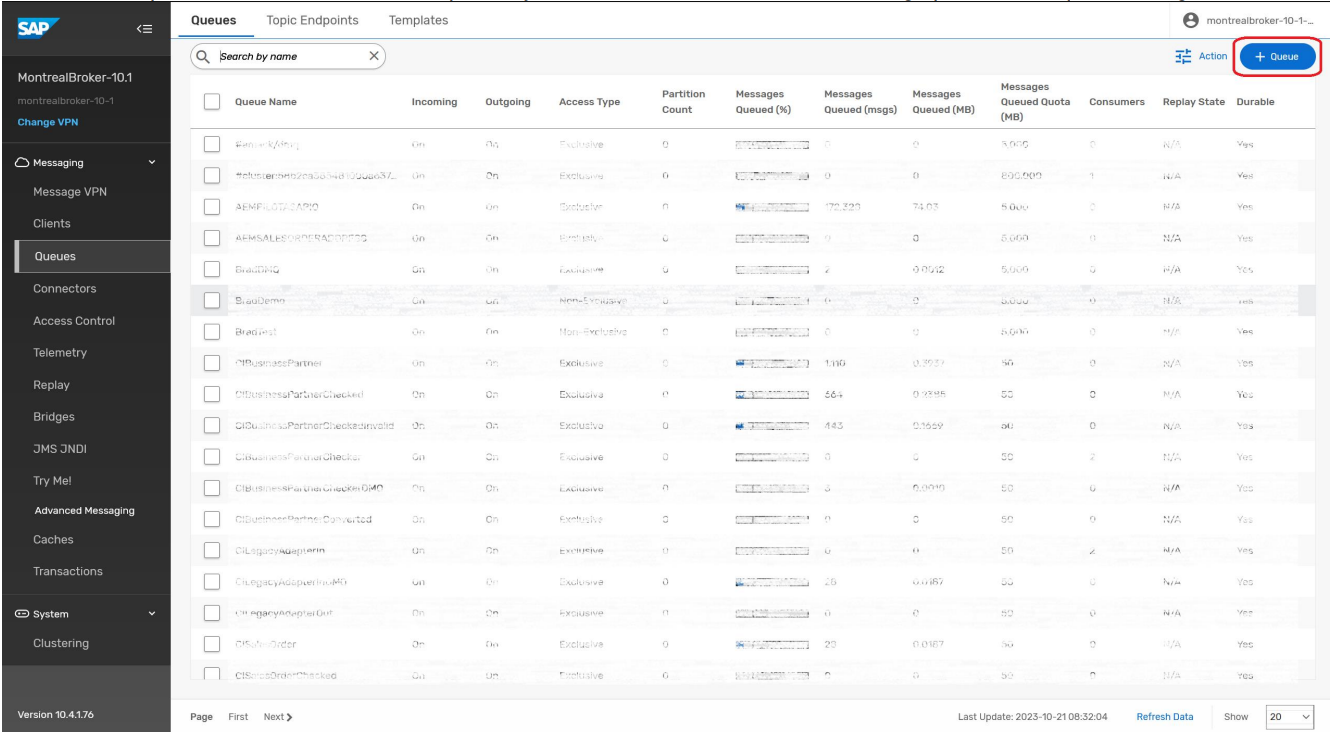
* Go to Cluster Manager -> {your service} -> Manage -> Queues - to open the Broker UI



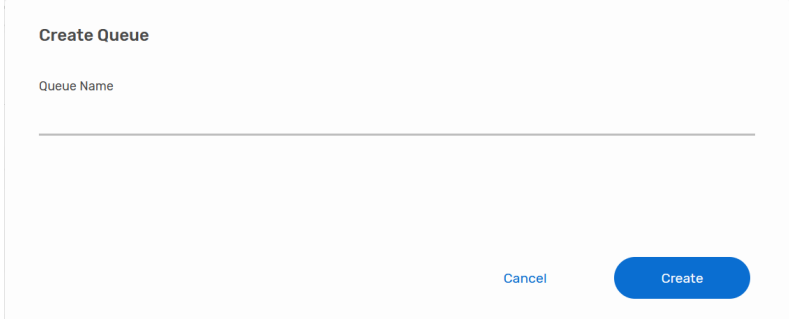




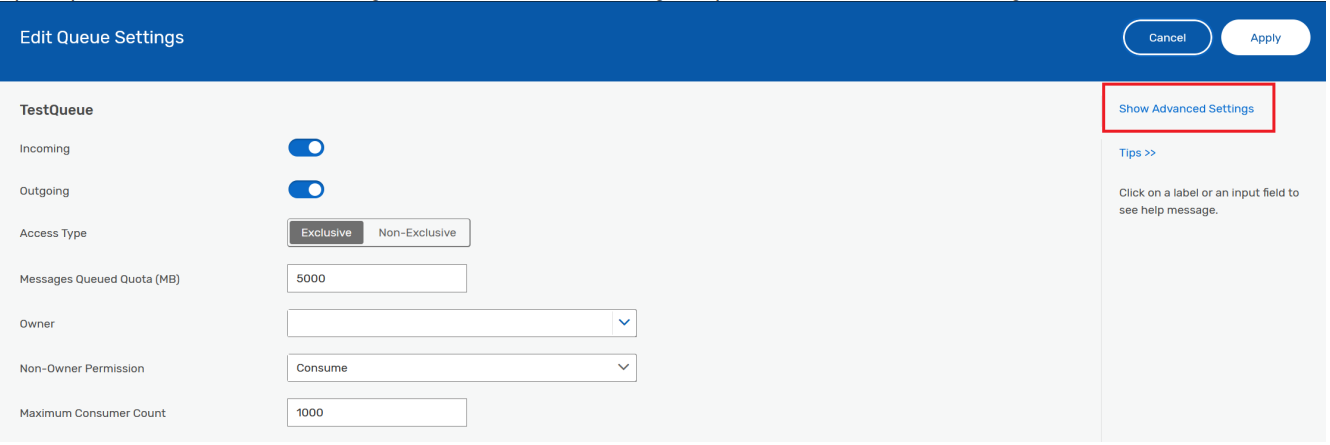
* To create a queue click the “+ Queue” button to bring up the create queue dialog

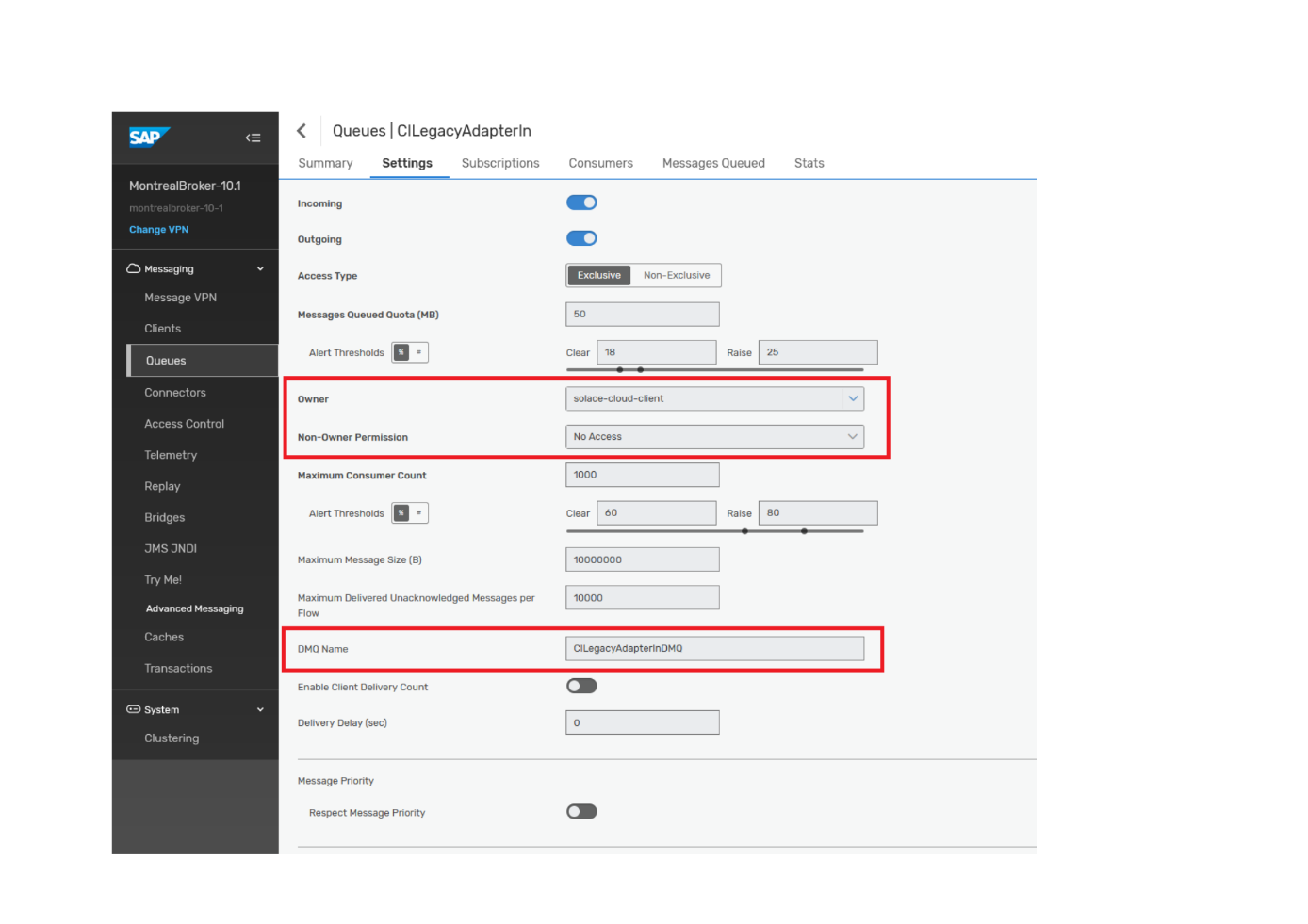


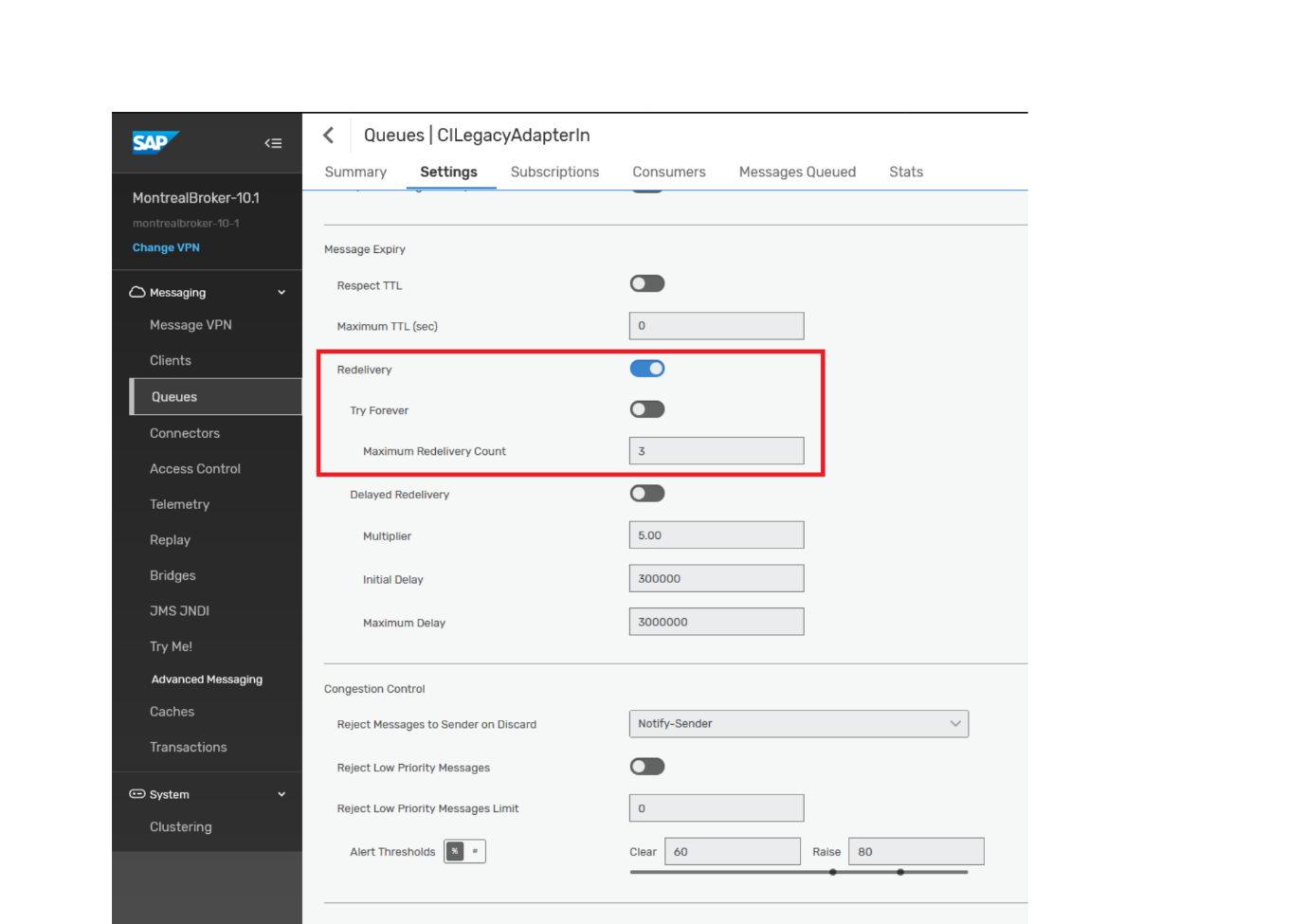
* Provide a name for this queue or decide on your own  
  ! – For the sake of compatibility with the Sales Order Dashboard from Day 1 the name should be chosen as *CILegacyAdapterIn*



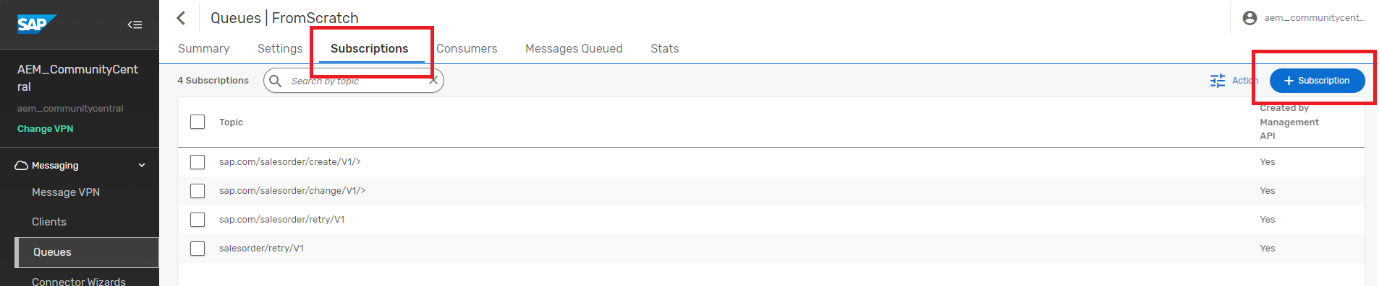
* Open up the "Advanced Queue Settings" section, then follow along and provide the details as showing in the screenshots below





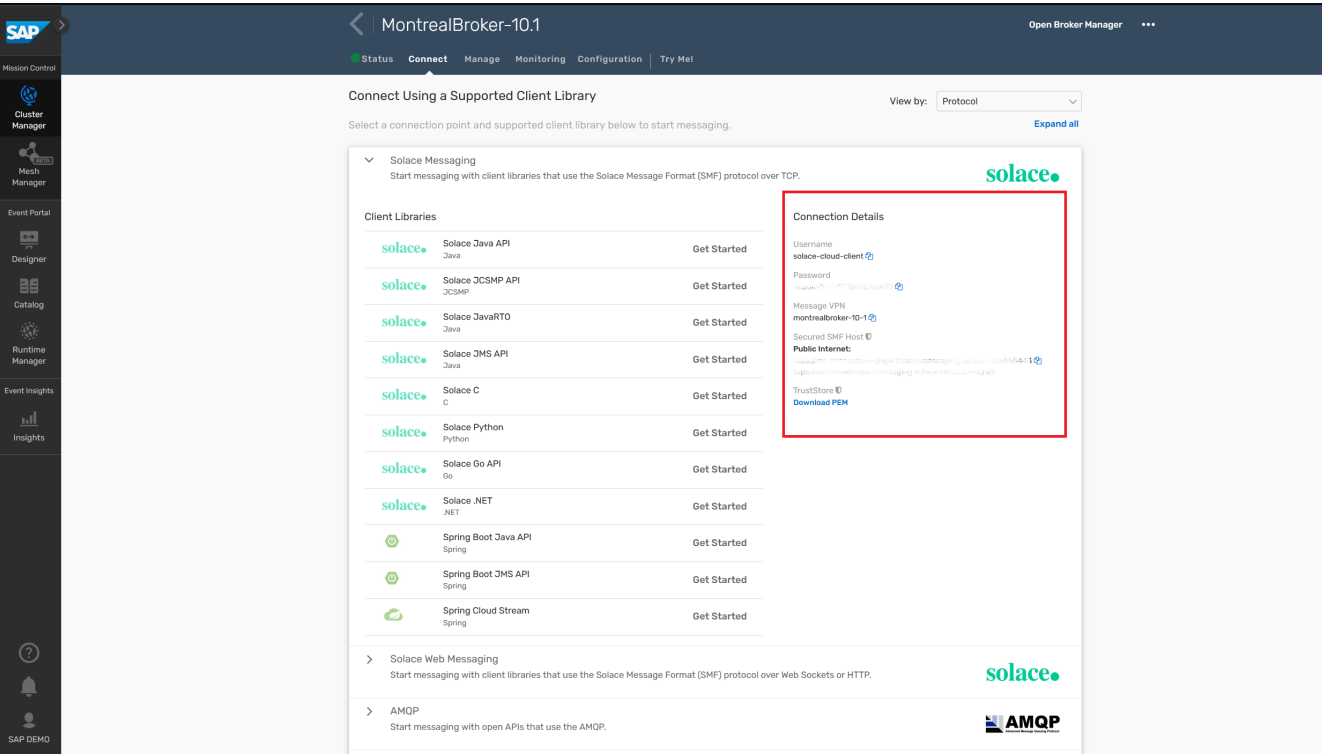


* Once the queue is created, click on the queue name in the list, navigate to the Subscriptions tab and open the subscriptions dialog.



* Add the following subscriptions to the queue
  1. sap.com/salesorder/create/V1/>
  2. sap.com/salesorder/change/V1/>
  3. sap.com/salesorder/retry/V1
  4. salesorder/retry/V1

Now, before we jump into Integration Suite: Let's head to our Advanced Event Mesh Console and go to Cluster Manager, select the service that you want to connect your Integration Suite flows to and go to the "Connect" tab. Take a note of the connectivity details underneath "Solace Messaging" (click on the section to open it up):

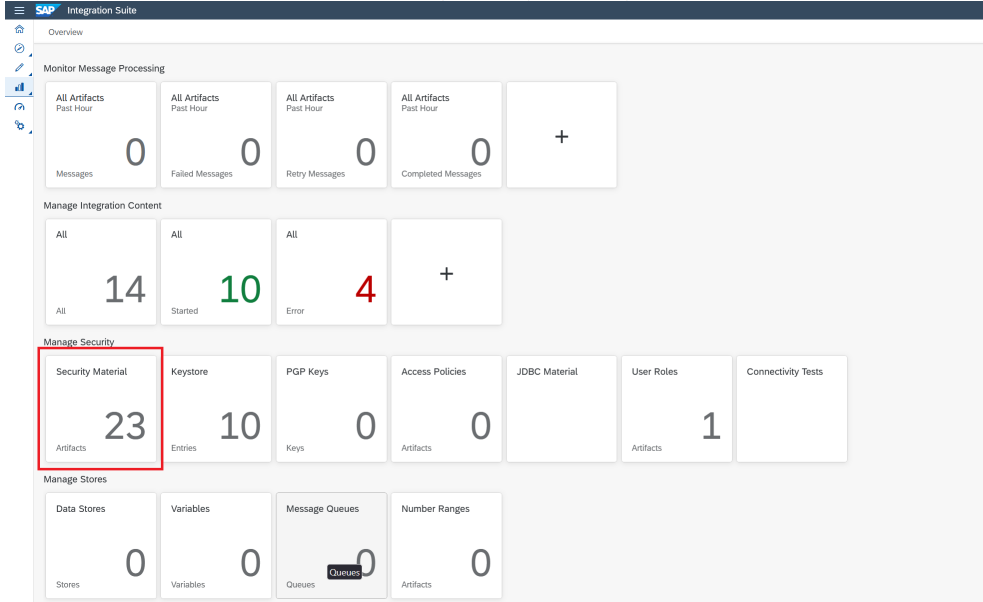


The connect tab lists all the various connectivity details for the various supported protocols. The AEM adapter uses the Solace Messaging protocol, which is AEMs very own protocol with a broad feature support. Each AEM service also comes with a default client user called *solace-cloud-client* that is configured for convenience reasons and is allowed to publish and subscribe to all topics. We will be using this user for all our IFlows. In a real production environment where security is important, you or your administrator will likely have this user disabled and will be creating separate users for each of the applications that connect to the AEM broker. Or this may even be deferred to an external authentication service over LDAP or OAuth with no client users stored on the broker itself and managed by your IAM service instead.

### **IFlow Creation**

The Integration Suite stored security details such as username/password pairs in so called Security Material. First, we configure security details for Basic Authentication against the Advanced Event Mesh.

* Go to Integration Suite Monitor Artifacts -> Manage Security -> Security Material



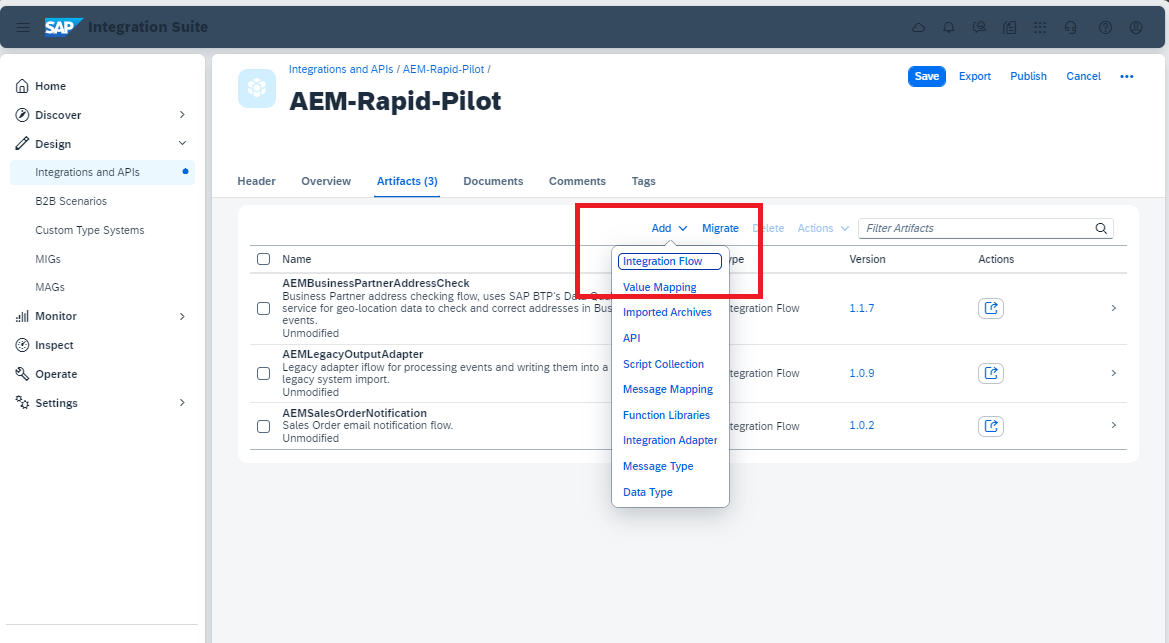
* Create Secure Parameter *CABrokerUserPass* and store the password for your solace-cloud-client application user credentials. We use this name since it will be the default used by the other provided IFlow content, but feel free to adhere to other naming conventions for this scenario.

Next, we will create a new IFlow and configure it with the minimum configuration to start receiving events.

* Navigate to the imported package in the Design section of the Integration Suite



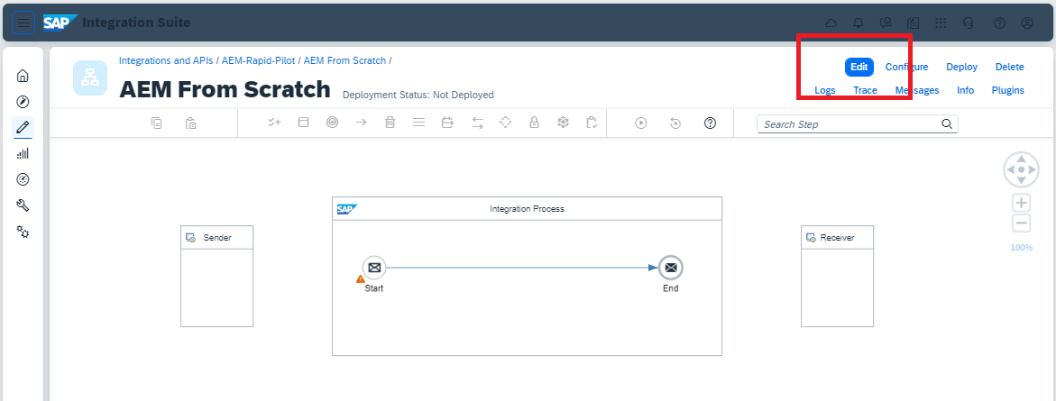
* To create a new IFlow first edit the Integration Package



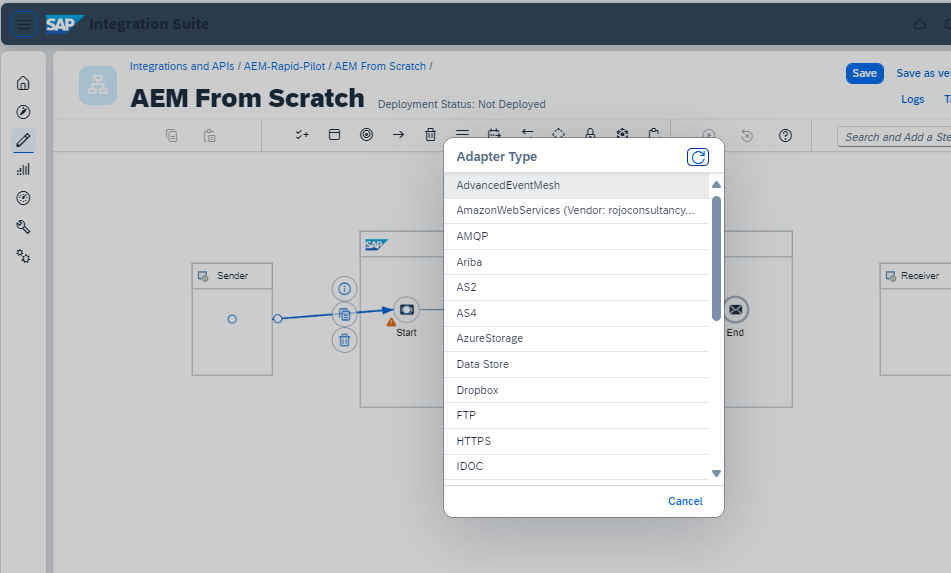
* The name and further details of this IFlow can be chosen however you like. In case you already have established Integration Suite guidelines within your landscape feel free to adhere to them



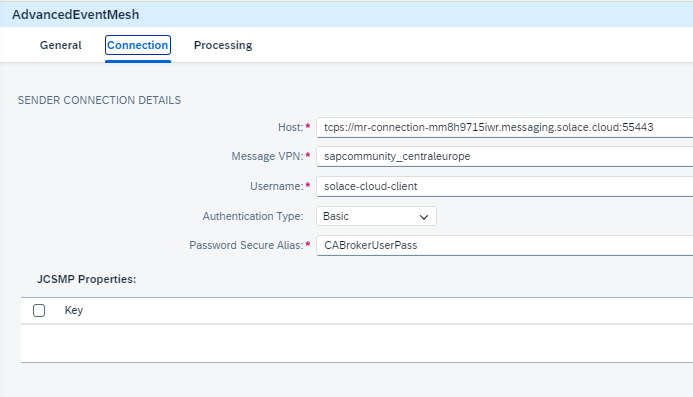
* Click “Add and Open in Editor”, you will be shown a completely empty IFlow template which we want to edit



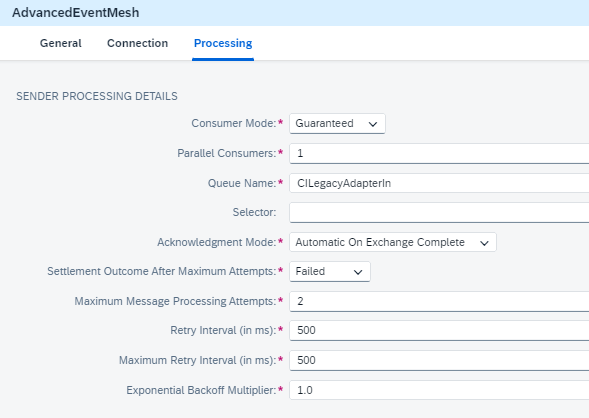
* Start by clicking the “Sender” and drag a connection to the “Start” of you Integration Process. Choose the AdvancedEventMesh Adapter



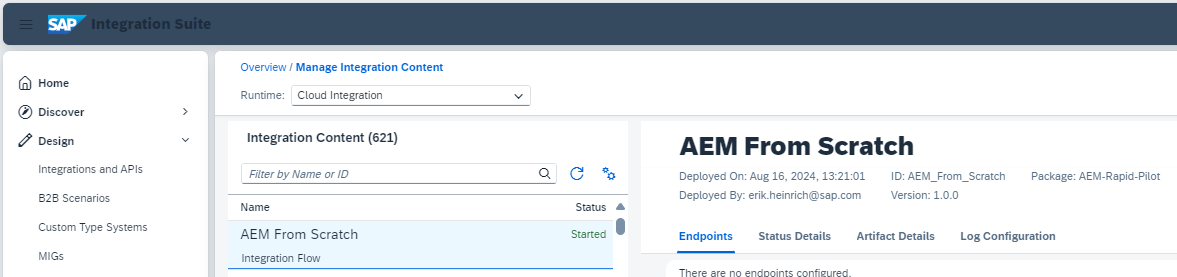
* The bare minimum that needs to be configured to enable the IFlow to be triggered by events is the following (replace the Host and Message VPN with the information you found in your broker at the beginning of these exercises and the name of the queue with yours).



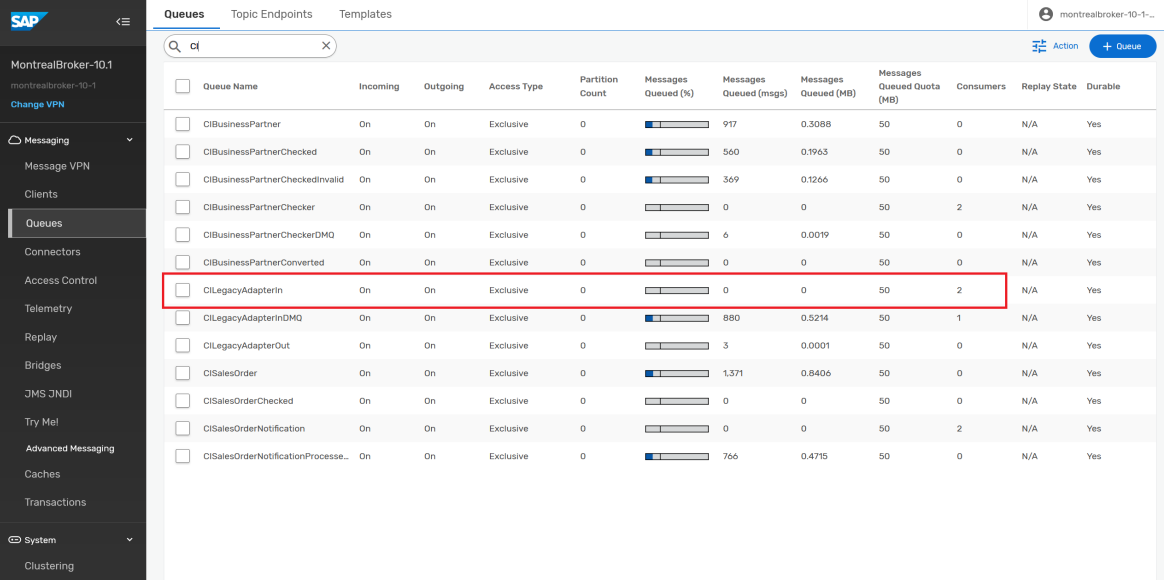
* Take care when configuring the Processing details. The next exercise will go into more detail on these settings.



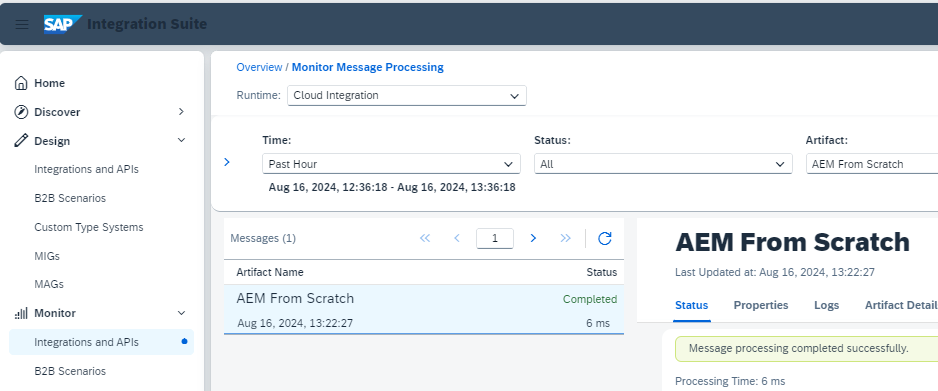
* Save your configured IFlow and hit deploy at the top right
* Go to Monitor Integrations and APIs -> Manage Integration Content -> All  
  If you did everything correctly you should find your IFlow with Status “Started”



* Navigate back to the queues in your AEM Broker, check that the queue you used for this exercise now has at least one consumer connected to it



Congratulations, if you can see the consumer connected to your queue, then your IFlow has successfully connected to the AEM and is up and running waiting for event messages to arrive. If you have the simulator tool set up and your queue configured properly you should be able to trigger the IFlow. To confirm go to Monitor Integrations and APIs -> Monitor Message Processing -> All

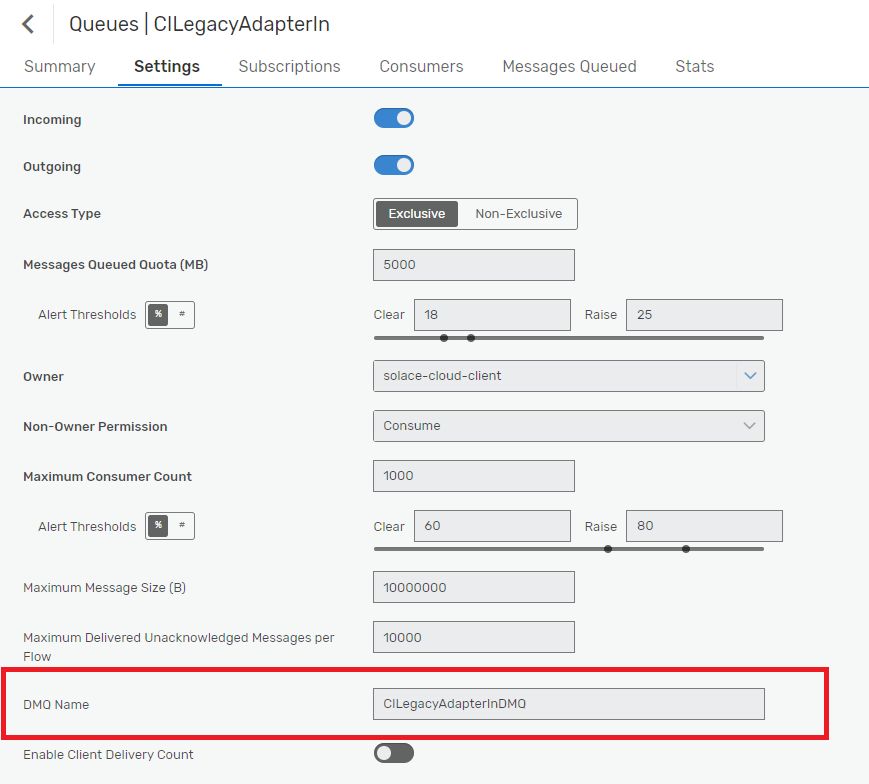


Of course, since the created IFlow is just an empty message consumer you will see nothing happening except for the entry in the Monitoring of the Integration Suite. The following exercises will introduce a more elaborate event handling.

# Scenario 2: “Dead Message Queue”

In this scenario we will modify the IFlow we just created to intentionally result in an error which in turn will cause AEM to move the message to a Dead Message Queue (DMQ).

* In the previous scenario we already configured the DMQ in our queue in AEM:



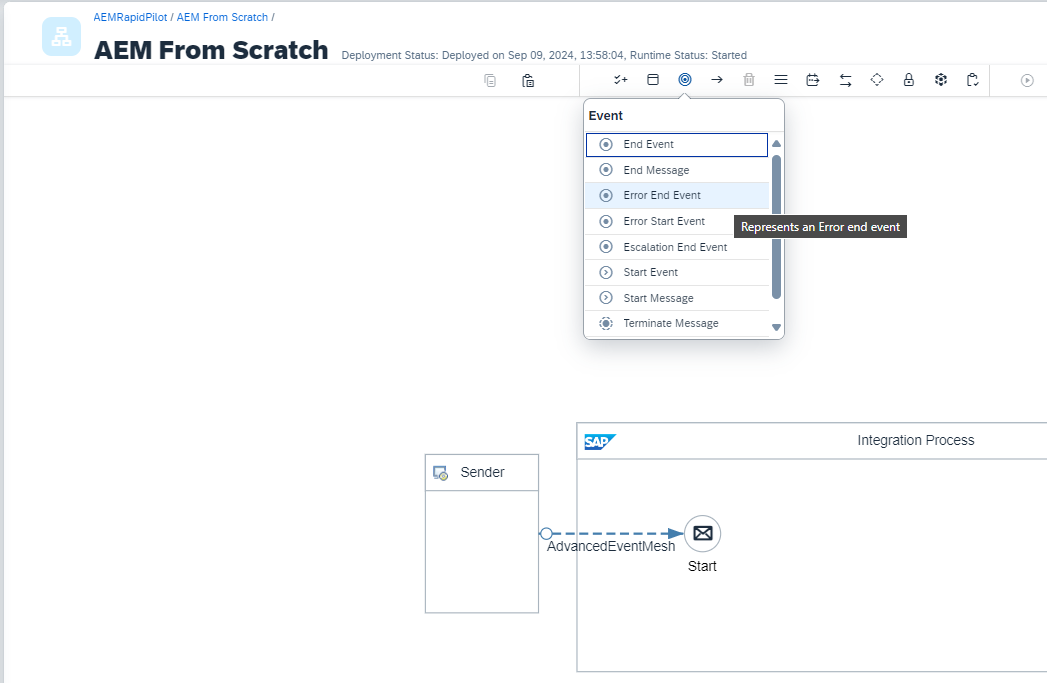
* Create a new queue with the name you have configured as DMQ Name in your initial queue. This queue will not need any subscriptions since the messages will be automatically moved here after Retry handling.



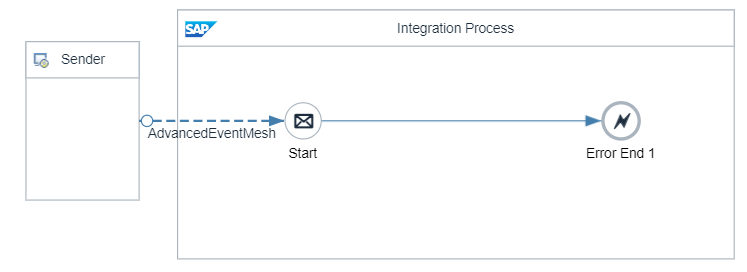
* Now we will make a tiny change to our IFlow. Go back into the Integration Suite and edit your IFlow.



* Delete the Message “End” and replace it with an Error End Event



* Click on the Message “Start” and draw the connector to the new Error End



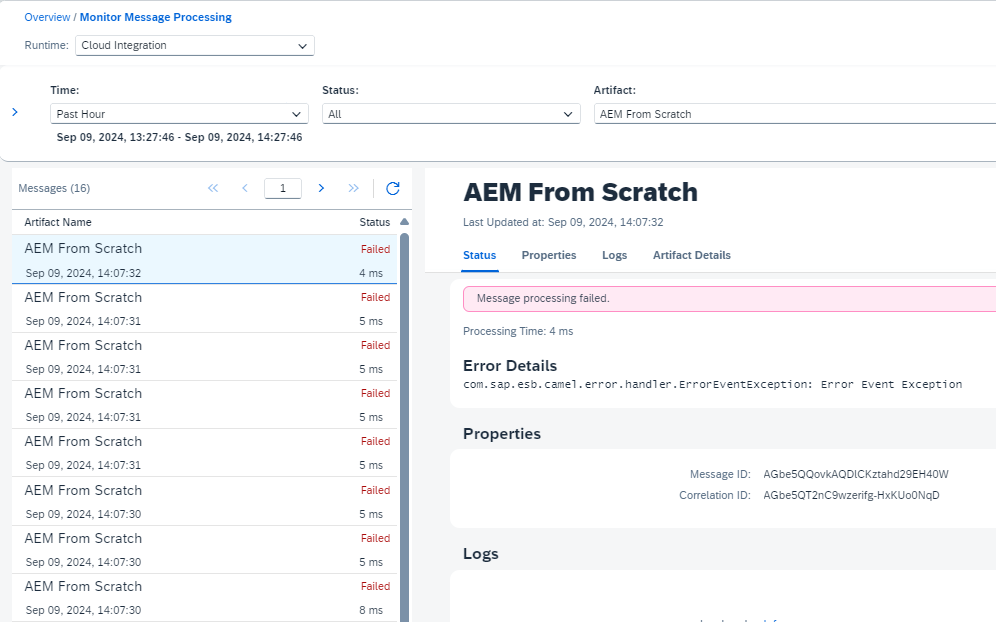
* Save your changes and deploy the IFlow again

Let's now look at these Processing settings in the AEM adapter one by one:

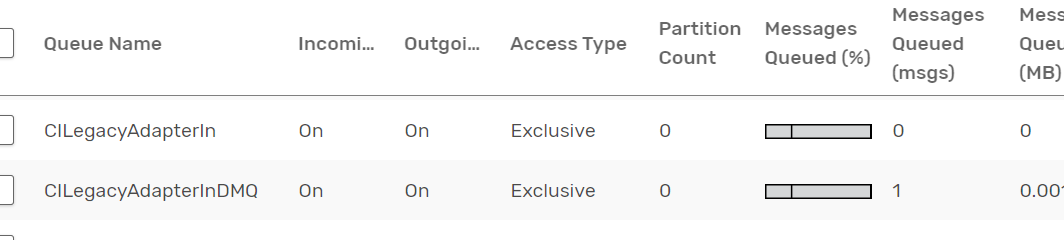
* + **Acknowledgement Mode: "Automatic on Exchange Complete"**The most important setting when it comes to not accidentally acknowledging and therefore removing a message from the broker's queue. This setting tells the flow/AEM adapter to only acknowledge (ack) the message after the flow has successfully completed processing the message. If any in the processing occurs, the AEM adapter will instead send a negative acknowledgment back (nack) to tell the broker to keep the message and retry it, because it couldn't be successfully processed by the flow. The alternative is to immediately ack the message when it's received, which will always result in the message being removed from the queue even if the flow fails to successfully process the message. (!!)
  + **Settlement Outcome After Maximum Attempts: "Failed"**This setting controls the nack type and behaviour, we have two options here:  
     a) Failed, which will nack the message back to the broker and let's the broker check the retry count of the message to trigger retries based on the queue settings and only sending messages to DMQ when the retry count on the message has exceeded the max retry settings on the queue.  
    b) Rejected, which will nack the message telling the broker to immediately move the message to DMQ when the AEM adapter settings (Maximum Message Processing Attempts) are exceeded irrespective of queue settings.
  + **Max. Message Processing Attempts: 2**  
    Controls how often we want to retry a message inside the IFlow before we "give up" and pass it back to the broker.
  + **Retry interval, Max Retry Interval and Exponential Backoff Multiplier**  
    These are all settings that control how quickly we want to retry and whether we want to incremently increase our retry delay with each failure. A good retry delay value prevents the IFlow from repeatedly retrying a message within a few milli-seconds and gives some time for transient error situations to clear before we retry.

Note that the error handling and retry settings go hand-in-hand with the DMQ and retry settings on the input queue for this flow (queue retry settings multiply with the internal retry settings in the IFlow, e.g. if the IFlow tries 2 times internally every time we pass it a message and the broker is configured to retry the same message 3 times to the broker, then we might get 8 executions before the message is actually stopped being processed and moved to the DMQ [(1 initial attempt + 3 times retry) \* 2 times retry inside the IFlow = 8 processing attempts]). We will observe this behaviour in the next exercise/scenario.

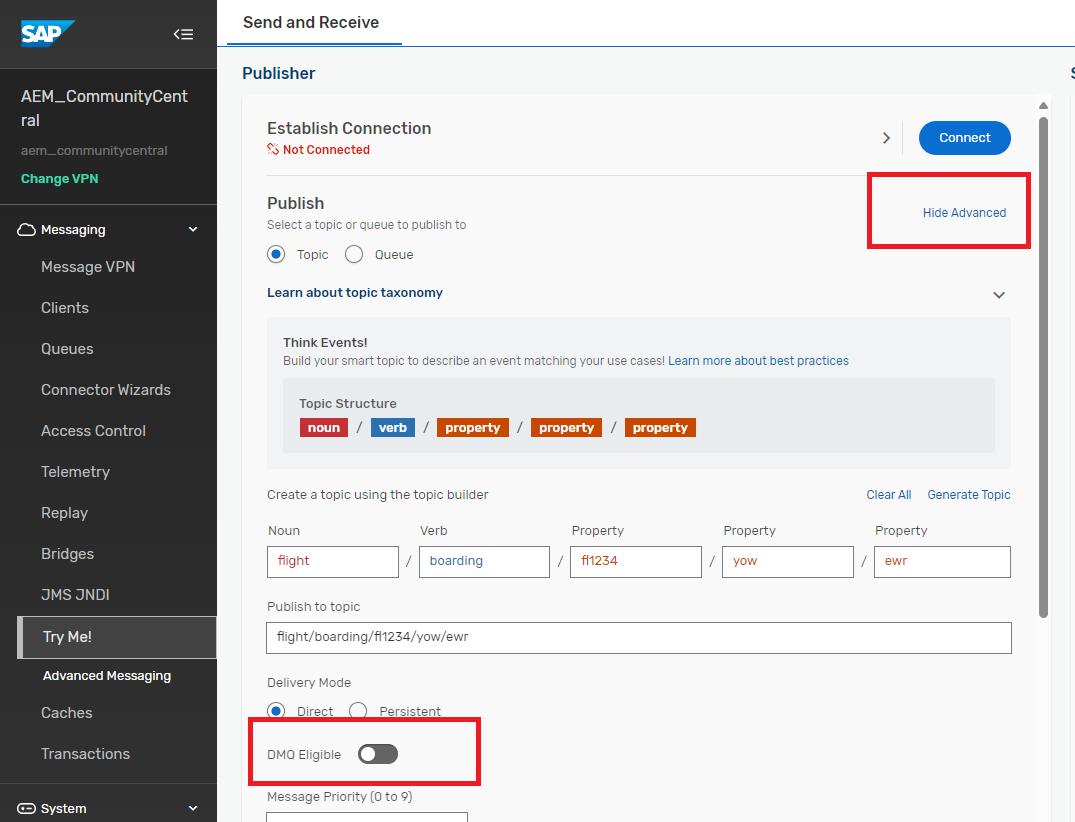
Now if you use any tool you prefer to publish a Sales Order Event to our queue (the ERP simulator or the Try-Me console in the AEM broker itself) you will find the IFlow has been triggered and resulted in failure exactly 8 times:



And your DMQ in your message Broker will now have a new message in it:



Be aware that the Publisher decides if an event is eligible to be stored in a DMQ, so when testing with the Try-Me feature be sure to check “DMQ eligible” under “Show Advanced”



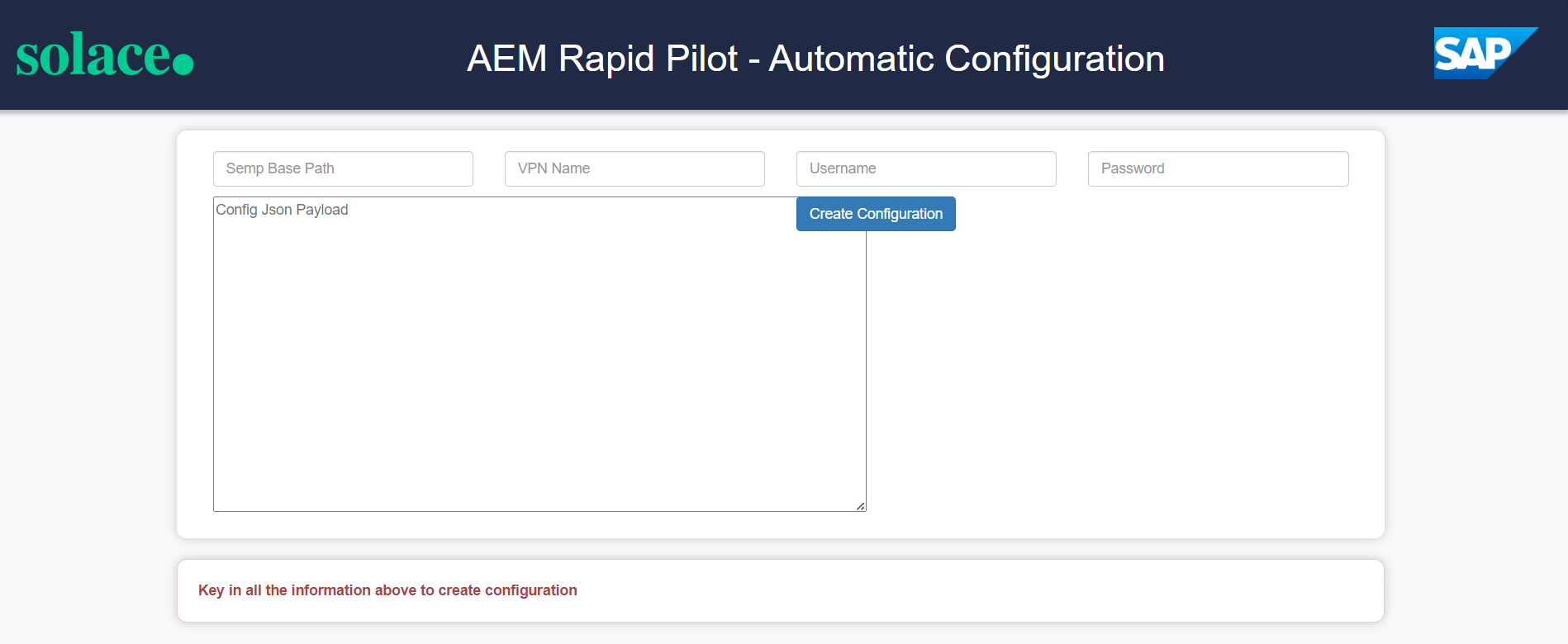
# Automated AEM Broker Setup via APIs and CI/CD (optional)

All configuration within AEM can be automated using a combination of two APIs:

* AEM Cloud API (for any configuration on the AEM Cloud Console)
* The broker's SEMP API (for any configuration on the broker service directly)

The documentation of these APIs can be found in the AEM docs [here](https://help.pubsub.em.services.cloud.sap/Cloud/gqs_building_apps.htm)

Both these APIs are RESTful and can be used in numerous ways to pull/push configuration like queues and client configuration through CI/CD pipelines or configure/promote Event Portal content in step with code promotion from environment to environment. There are many options/tools that can be used to automate these tasks, like Jenkins, Ansible, scripts, Terraform etc. For the purpose of this exercise, we are going to use a sample application written in CAP that has the nice advantage of running in our browser that we can point at our broker's API and feed some configuration files.

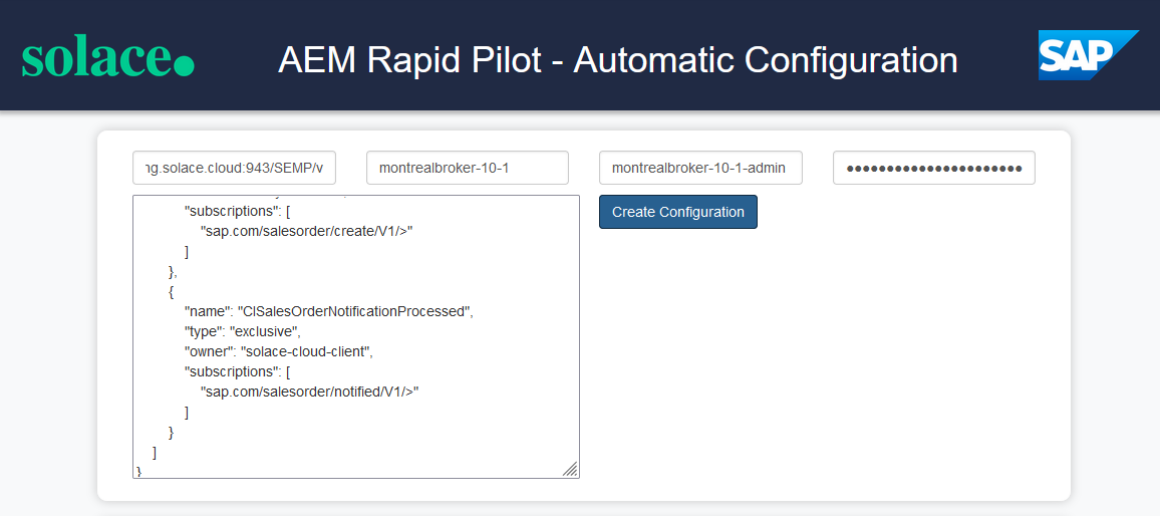


The AEM Rapid Pilot - Automatic Configuration can be accessed [here](https://rapid-pilot-createconfig-quiet-elephant-yt.cfapps.ca10.hana.ondemand.com/)

We will need some details from your AEM service again to connect the configuration tool with your AEM service. Let's head to our Advanced Event Mesh Console and go to Cluster Manager, select the service that you want to connect your Integration Suite flows to and go to the "Manage" tab. Take a note of the connectivity details underneath "SEMP - REST API" (click on the section to open it up)



Copy & paste the URL, vpn name, admin username and password into the config tool:



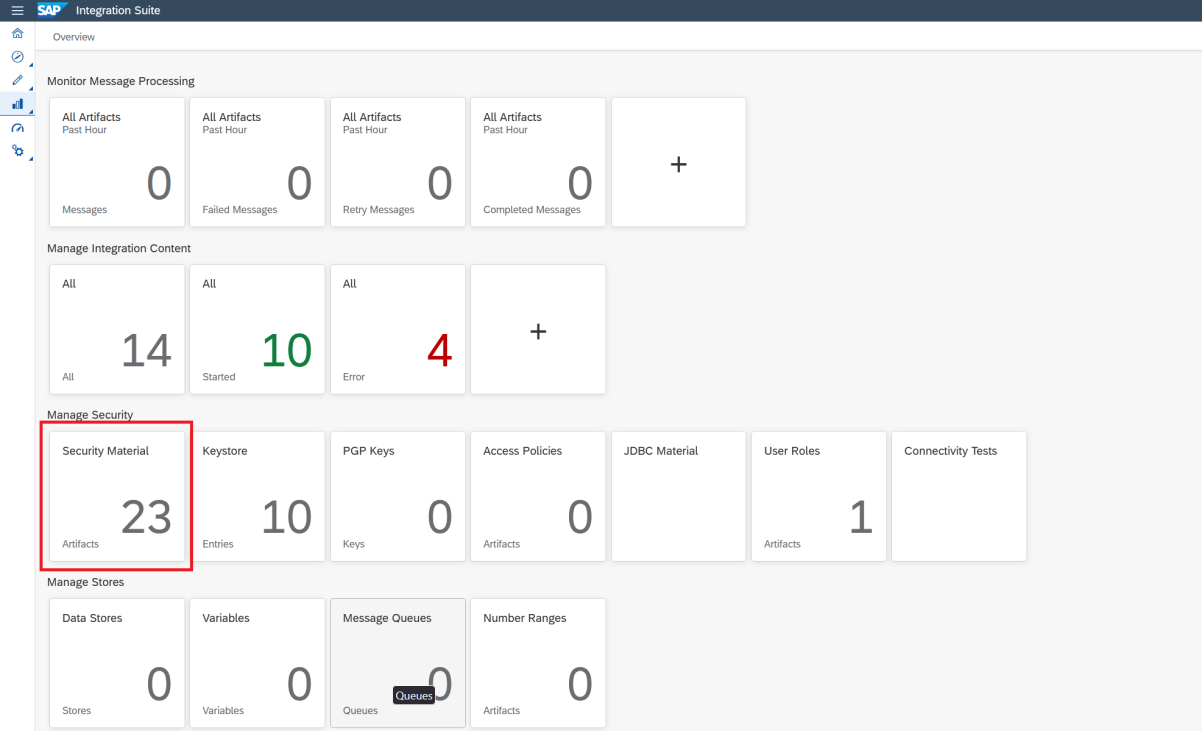
Please download the configuration file IS\_SEMP.json provided to you together with the content packages and copy & paste the content into the "Config JSON Payload" input field. Hit "Create Configuration" to apply this config to your broker.

# Scenario 3: “Sales Order E-Mail Notification”

### **Setup/Configure Dependency Services**

We will give you connectivity details to one of our brokers where we have an IFlow deployed that is configured to send emails via an external email service to enable us to automatically send confirmation emails. You are welcome to use a SMTP mail server of your own if you have access to the necessary credentials. You will need to edit the provided IFlow slightly but if you are interested, we are happy to support.

* If you want to use the broker we provide you need to go to the Integration Suite – Monitor Artefacts – Manage Security – Security Material



* You should have created a Secure Parameter (CABrokerUserPass) in the first exercise already
* Create another Secure Parameter *email-profile-pwd* and store the password we have handed out in the workshop.

### **Queue Setup**

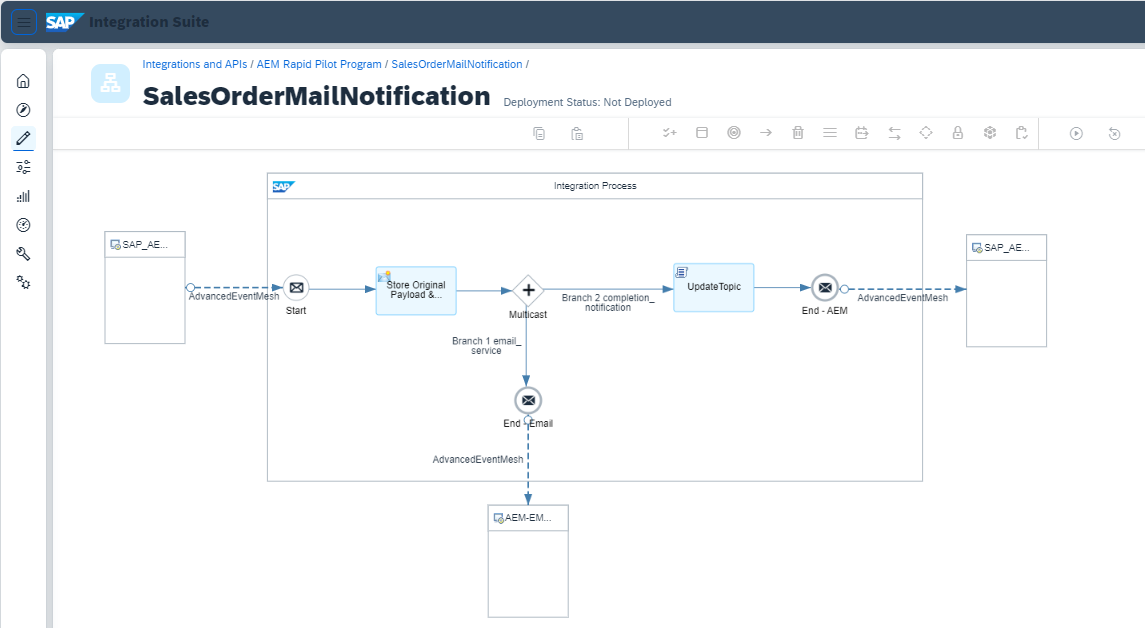
If you have used the CI/CD tooling in the previous section to automate the configuration of the AEM queues you can skip this section. If not, have a look at the IS\_SEMP.json provided to you. For this exercise you will need the following queues:

* **CISalesOrderNotification**
* **CISalesOrderNotificationProcessed**

Create the queues like you did in the previous sections using the details in the SEMP file for topic subscriptions.

### **IFlow Setup**

Let us first look at the SalesOrderMailNotification IFlow:



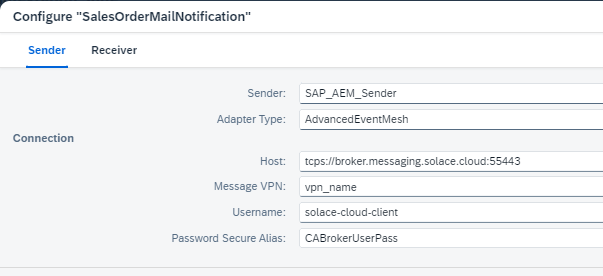
This flow gets triggered by Sales Order events and does two things:

a) It creates an email request and by forwarding this event to an email service iflow on another broker (hosted by us) on topic  
*sap.com/emailnotification/created/V1*.

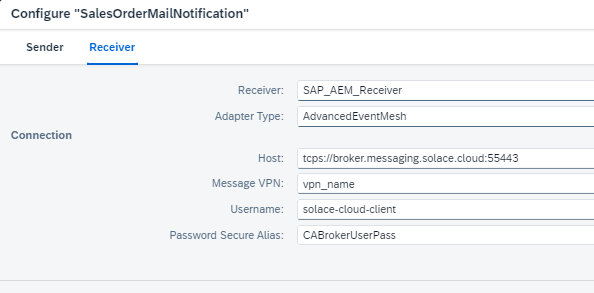
b) It sends a new event to *sap.com/salesorder/notified/V1/{salesOrg}/{distributionChannel}/{division}/{customerId}*

to indicate that the email request was successfully forwarded.

* Hit configure at the top right and fill in the details to connect to your AEM broker service like you did in the previous exercises



* There is no need to configure the Receiver separately, the configuration will fill for both.

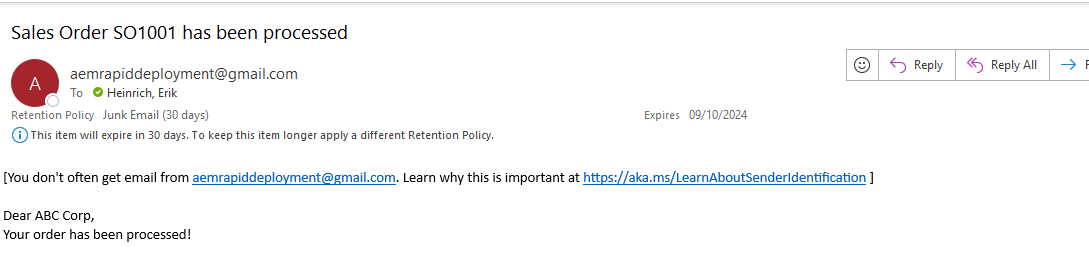


* No need to configure the adapter connecting to our email service, we've prepopulated this one and we have already deployed the necessary security configuration in the step above
* Hit deploy at the top right.

If you get confused about which parts of your IFlow to connect to your broker and which ones to connect to our broker, remember this simple rule: The sender and receiver (left and right of your IFlow) connect to your broker. The connector down at the bottom connects to our, from your point of view external, broker.

Like in the first exercises feel free to test if everything is setup correctly. The IFlow on our end that will send out the email notification expects an event with the same structure as the ExampleSOEvent.json provided by us. Simply modify the content of said SalesOrder events with your mail address before pushing it into your broker.

If all went well you will receive a mail (check your Spam/Junk folders):



# Scenario 4: “Business Partner Address Check” (optional)

### **Setup/Configure Dependency Services**

The next IFlow that we are going to deploy is invoking the SAP Data Quality Management service (DQM) to check and cleanse address data in the BusinessPartner events. For the flow to work properly, you will need a working DQM service subscription so you can configure your IFlow with this. For completing this section, you have two options:

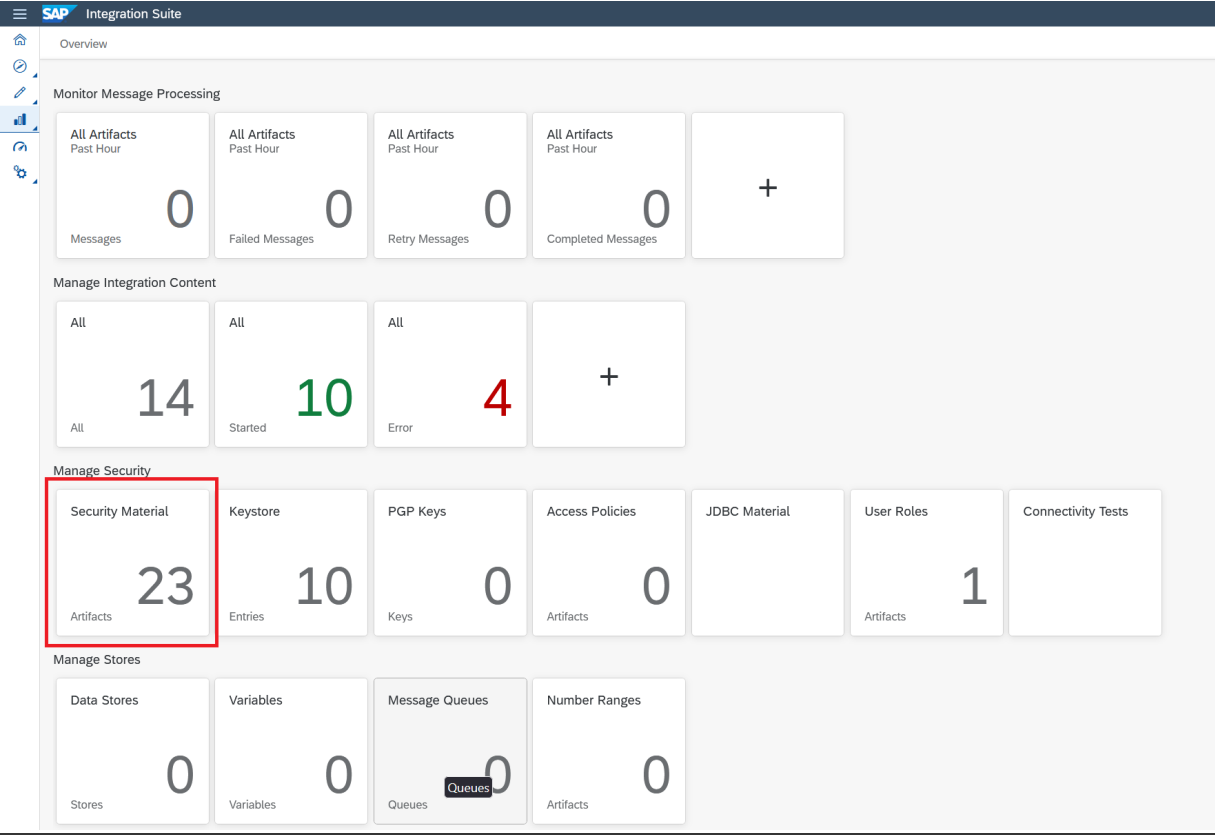
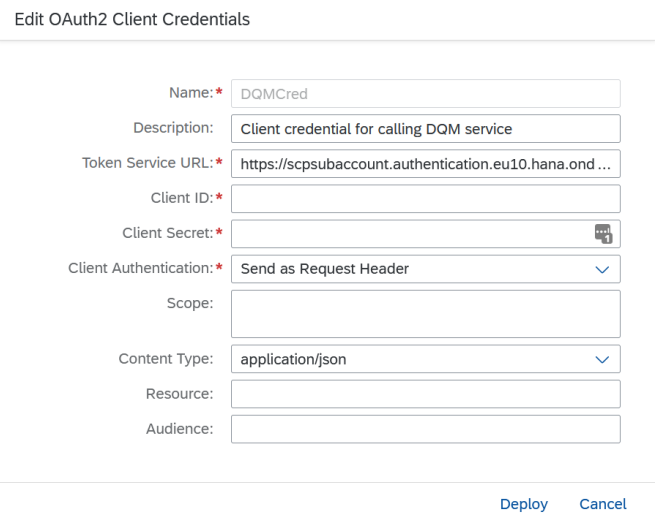
**A) Use DQM service credentials provided by us during the workshop**We will hand out the token and connectivity details to our DQM service, which can use

**B) Alternative: Activate your own SAP Data Quality Management service in BTP**Please note that if you want to proceed down this route, it may take some time to complete, so you may want to complete this in your own time after the workshop.

The good news, if you don't have a DQM subscription already or are not using our instance, then you can use a free tier subscription for this purpose. Please follow along the steps in this [blog post](https://community.sap.com/t5/technology-blogs-by-sap/getting-started-with-sap-data-quality-management-microservices-for-location/ba-p/13527838) by Hozumi Nakano to active the service.

Additionally, you will have to create a service instance and a service key to be configured with your integration flow later. Follow [these steps](https://developers.sap.com/tutorials/btp-sdm-gwi-create-serviceinstance.html) to create a service instance and key. Take a note of the URL and user credentials once you've activated the service.

Independent of whether you are using our provided DQM service or your own you will want to create an OAuth2 Client Credential.

* Go to Integration Suite Monitor Artifacts – Manage Security – Security Materials  
  
* Create OAuth2 Client Credentials with the name *DMQCred*, this is important because it will be used by the provided IFlow.  
  
* For Token Service URL use the uaa/url from the DQM service key **appended with /oauth/token** or the one we provide
* For ClientID/Secret use uaa/clientid and uaa/clientsecret from the DQM service key or the one we provide

### **Queue Setup**

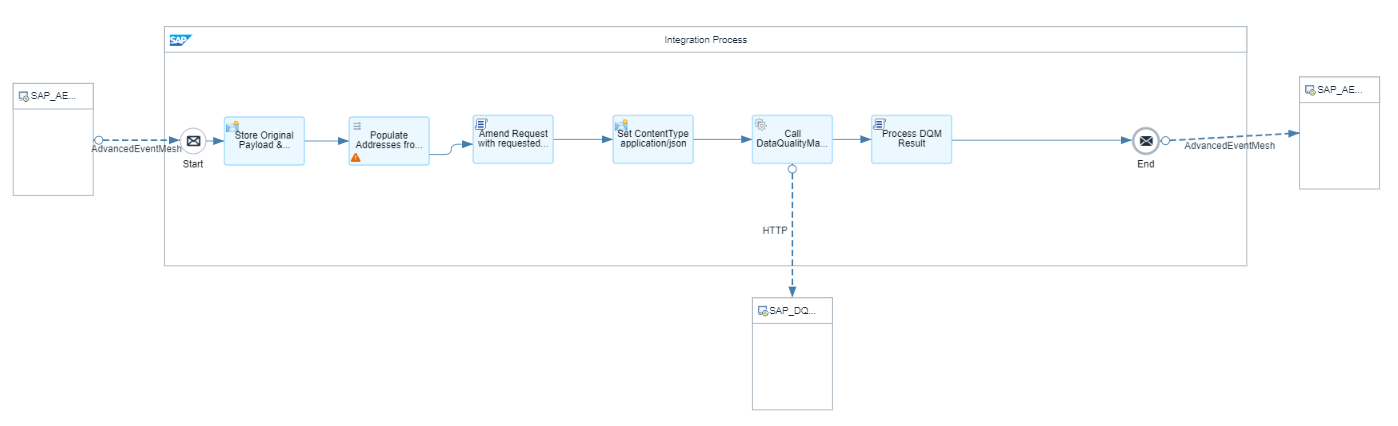
If you have used the CI/CD tooling in the previous section to automate the configuration of the AEM queues you can skip this section. If not, have a look at the IS\_SEMP.json provided to you. For this exercise you will need the following queues:

* **CIBusinessPartnerChecker**
* **CIBusinessPartnerCheckerDMQ**
* **CIBusinessPartnerChecked**
* **CIBusinessPartnerCheckedInvalid**

Create the queues like you did in the previous sections using the details in the SEMP file for topic subscriptions.

### **IFlow Setup**

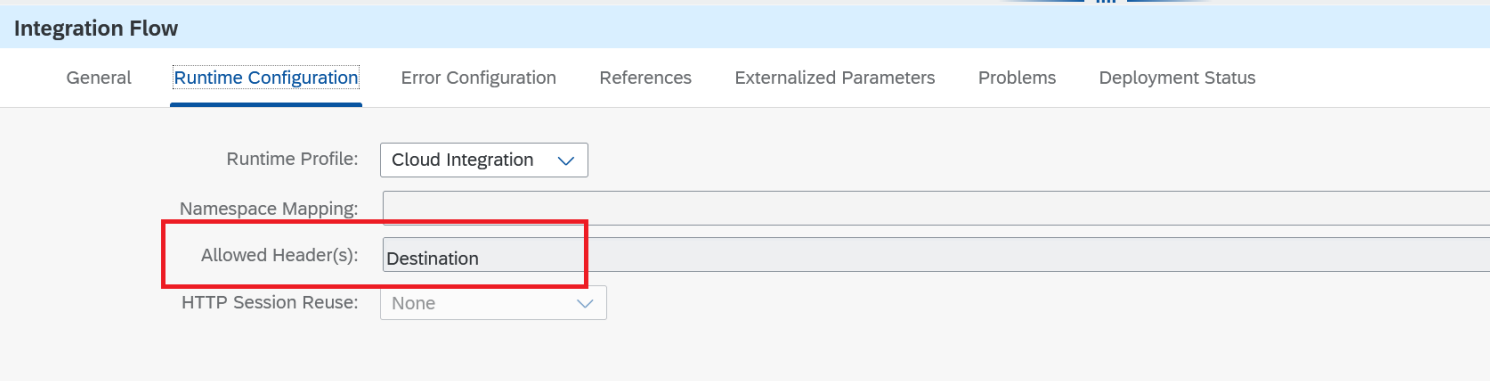
Let's look at the BusinessPartnerAddressCheck IFlow:



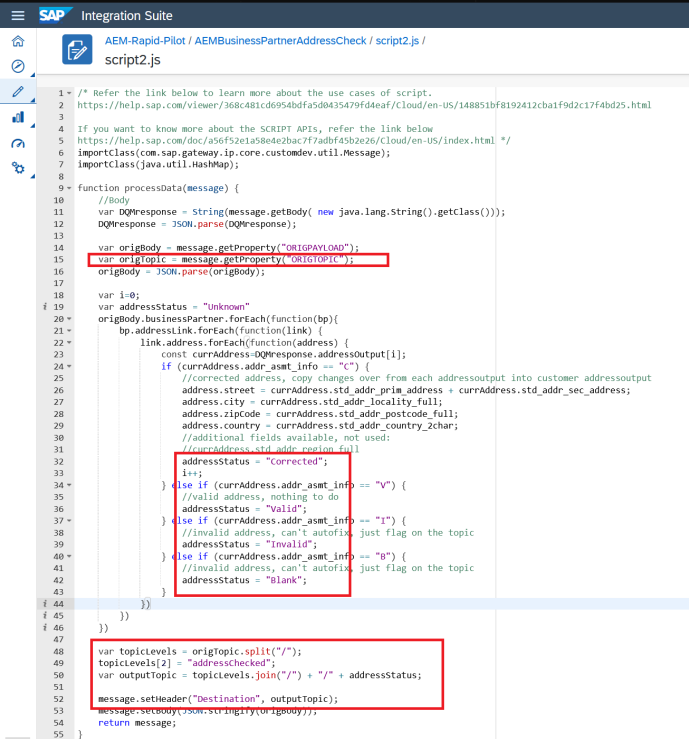
This flow receives Business Partner Create and Change events and invokes the Data Quality Management Service in BTP to check and correct the addresses inside the Business Partner event payload. It does this by

* + 1. Storing the original event payload in an environment variable.
    2. Populating the DQM request payload with the addresses in the input event.
    3. Invoking the DQM service over REST and
    4. Parsing the response, checking whether the DQM service evaluated the input addresses to be Valid, Invalid, Blank or has Corrected them.
    5. Merging any corrected addresses back into the original payload.
    6. And finally publishing the result back as a new event to the AEM broker with an updated topic in the format: *sap.com/businesspartner/addressChecked/V1/{businessPartnerType}/{partnerId}/{addressCheckStatus}*

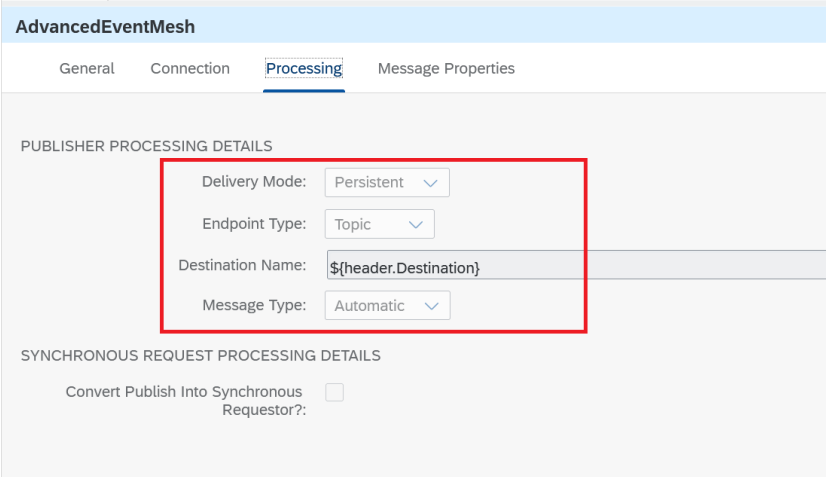
Let's also look at what happens in order to publish a new event back to the Advanced Event Mesh broker. First, on the integration flow overall configuration settings, we are preserving the destination header field to have access to the original topic that this event was published on. This matters, because the event may contain valuable meta-data that helps us and downstream consumers filter for events relevant to them and it saves us from reparsing the payload, which can be CPU and I/O intensive.



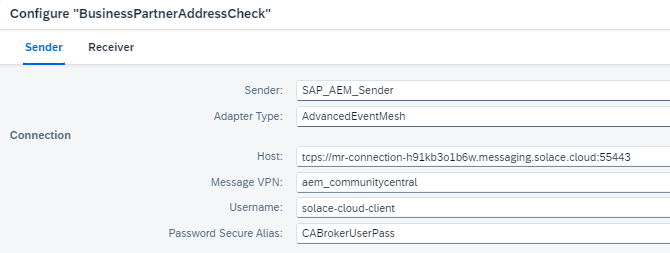
Secondly we are using a couple of lines in the script that is evaluating the DQM service result and merging the corrected addresses back into the original payload to retrieve and parse the original topic, replace one level (the verb) to create a new event and amend another extra meta-data level that contains the result of the address check (either Valid, Corrected, Invalid or Blank), which can be used by downstream systems to filter for specific outcomes. We are storing the newly created topic in the Destination field of the message header.



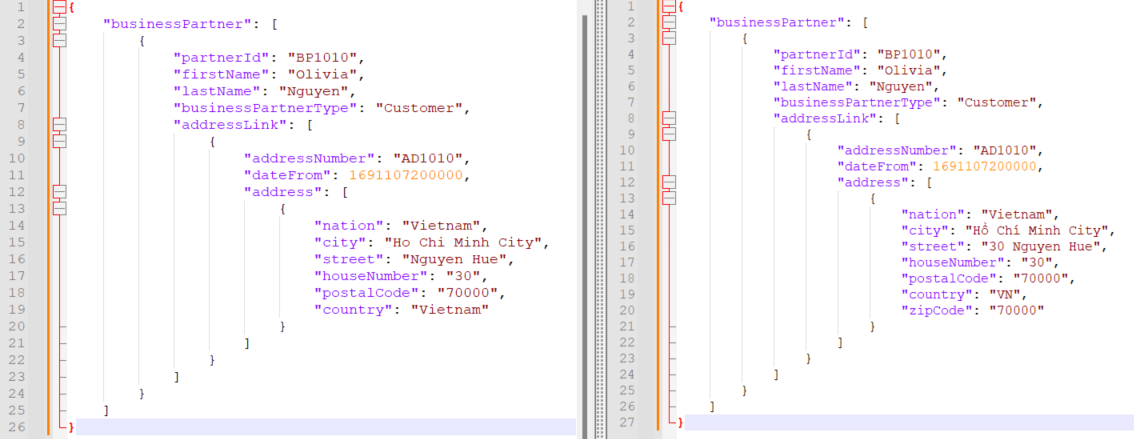
Lastly, the AEM Receiver adapter is configured to persistently (to avoid message loss) publish to a topic, taking the value from the header field that we set in the previous step/script.



Configuring and deploying the BusinessPartnerAddressCheck IFlow:

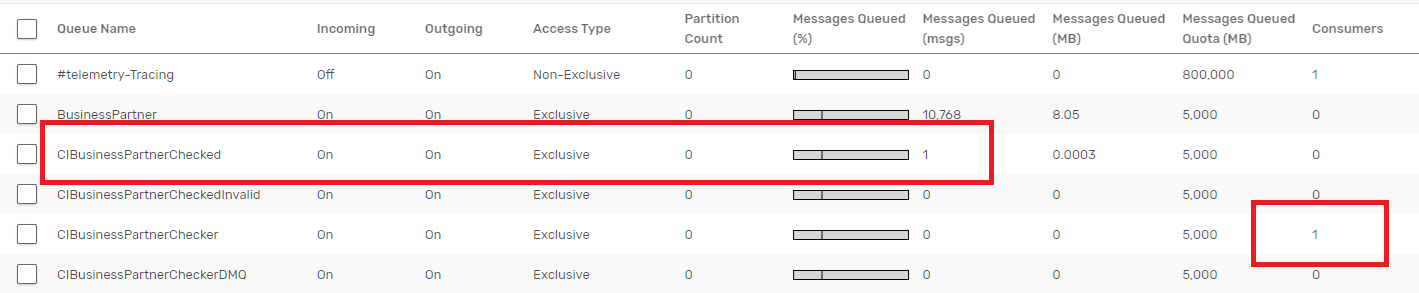
* Hit configure at the top right and fill in the details to connect to your AEM broker service like you did in the previous exercises.  
  
* No need to configure the adapter connecting to DQM, we've prepopulated this one and we have already deployed the necessary security configuration in the step above
* Hit deploy at the top right.

As before test if your scenario works using either the ERP simulator tool or the Try-Me feature. In the following example we happened to trigger a BusinessPartner event from the ERP simulator tool with an address in Vietnam that was missing necessary punctuation. You can see the actual JSON content of the event before the IFlow on the left and after on the right.



The event was then published on the topic:  
*sap.com/businesspartner/addressChecked/V1/Customer/BP1010/Corrected*

Which is included in the wildcard subscription of the “Checked” queue:

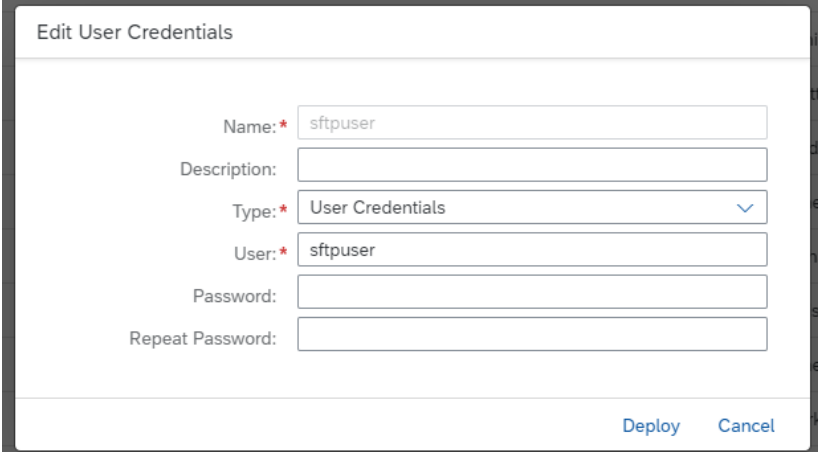


# Scenario 5: “SFTP as Legacy Integration” (optional)

### **Setup/Configure Dependency Services**

With the introduction of the Advanced Event Mesh, not all customers will be ready to adopt an Event Driven approach for all their applications. They will be able to innovate in some areas but also must sustain existing applications that use for example Flat Files. In this scenario, we will use a couple of the common Cloud Integration artifacts to facilitate this scenario. The common integration path for Flat Files is SFTP so in this scenario we will want to leverage a SFTP server. Similar to previous exercises you have two options:

**A) Use a SFTP server of you own business that you might have access to**If you want to proceed with this or if you want to complete this exercise after the workshop when you have access to an SFTP server you will have to create/modify 2 Security artifacts in the Integration Suite:

* A User Credential to Authenticate against your SFTP server. This can be as simple as Username/Password which you input and then deploy  
  
* You will have to add your SFTP server to the SSH Known Hosts list of your Integration Suite. Follow the instructions in [this blog](https://community.sap.com/t5/technology-blogs-by-members/generation-of-sftp-known-host-file-cloud-platform-integration/ba-p/13318162) for more information.

**B) Alternative: Use the included Groovy script in the IFlow to generate the file as an attachment to the IFlow run**This option will allow you to complete this exercise without an SFTP server. We still encourage you to try to setup the SFTP integration on your time.

### **Queue Setup**

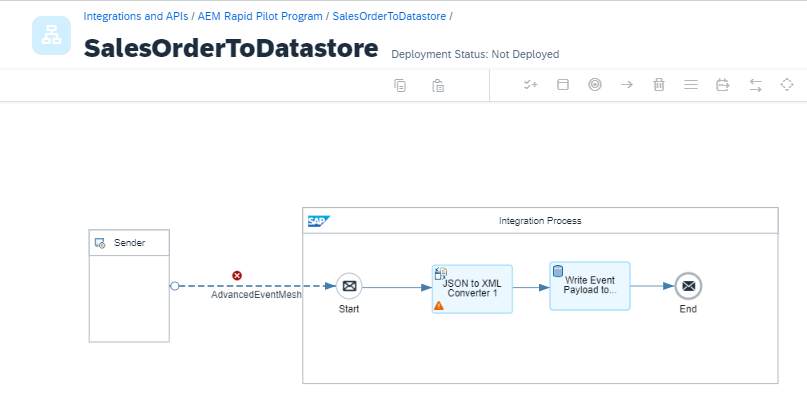
If you have used the CI/CD tooling in the previous section to automate the configuration of the AEM queues you can skip this section. If not, have a look at the IS\_SEMP.json provided to you. For this exercise you will need the following queues:

* **FFSalesOrders**

Create the queues like you did in the previous sections using the details in the SEMP file for topic subscriptions.

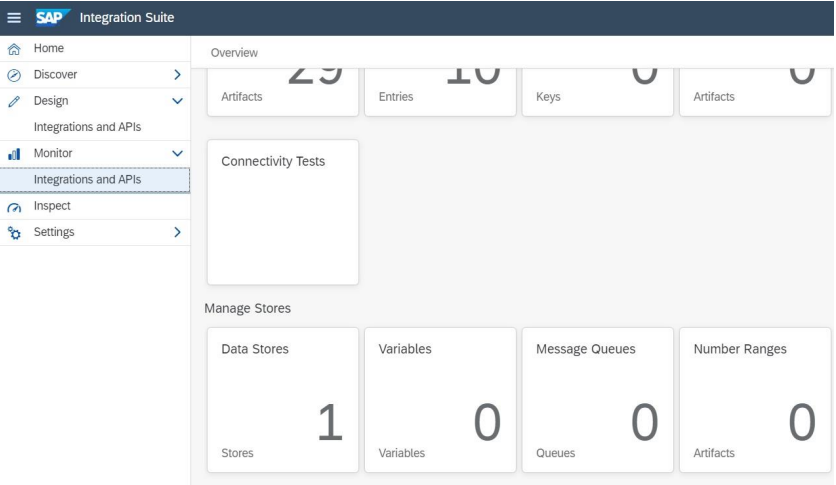
### **IFlow Setup**

The IFlow “SalesOrderToDatastore” will subscribe to events on the Advanced Event Mesh, convert them to XML (E.g. The DataStore object only works with XML) and then write them into the DataStore.

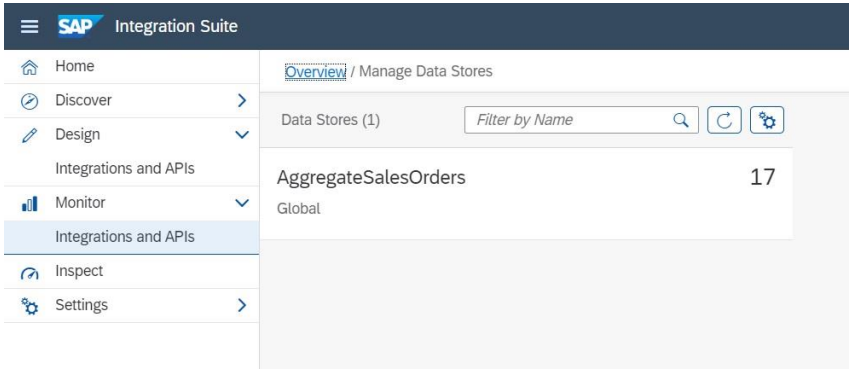


* To configure this IFlow all you need to do is enter your AEM broker information like in the previous exercises.
* Deploy the IFlow

By using any testing tool provided to you (ERP Simulator, Try-Me feature,..) you can to a quick testing. From the monitor, you should now see a Data Store as shown in the screenshot. (\*\*\* Of course, if you are already using Data Stores for your integrations, you should see the number incremented \*\*\*)



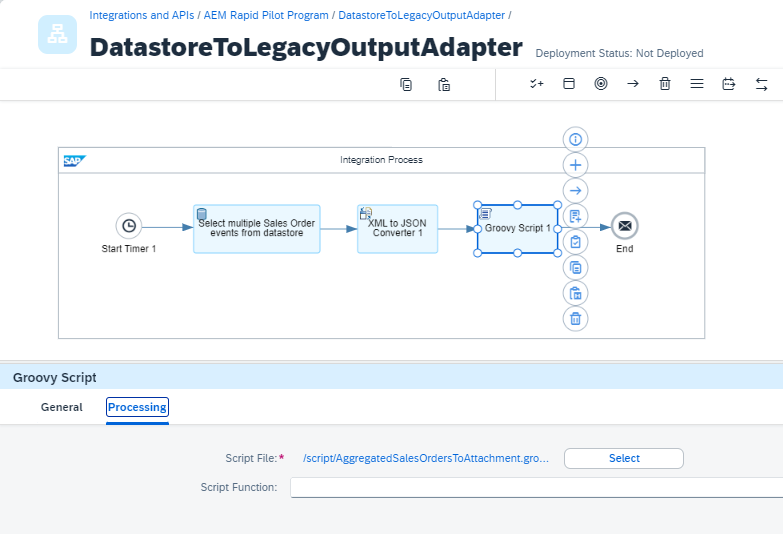
Clicking on the Data Stores Tile, you should see the name of the Data Store specified in the IFlow.



We will then use a second iFlow to pull the events from the DataStore, convert them back to JSON and then use the SFTP adapter to create a flat file with a batch of up to 10 of these events. This is based on the settings in the Data Store operation.

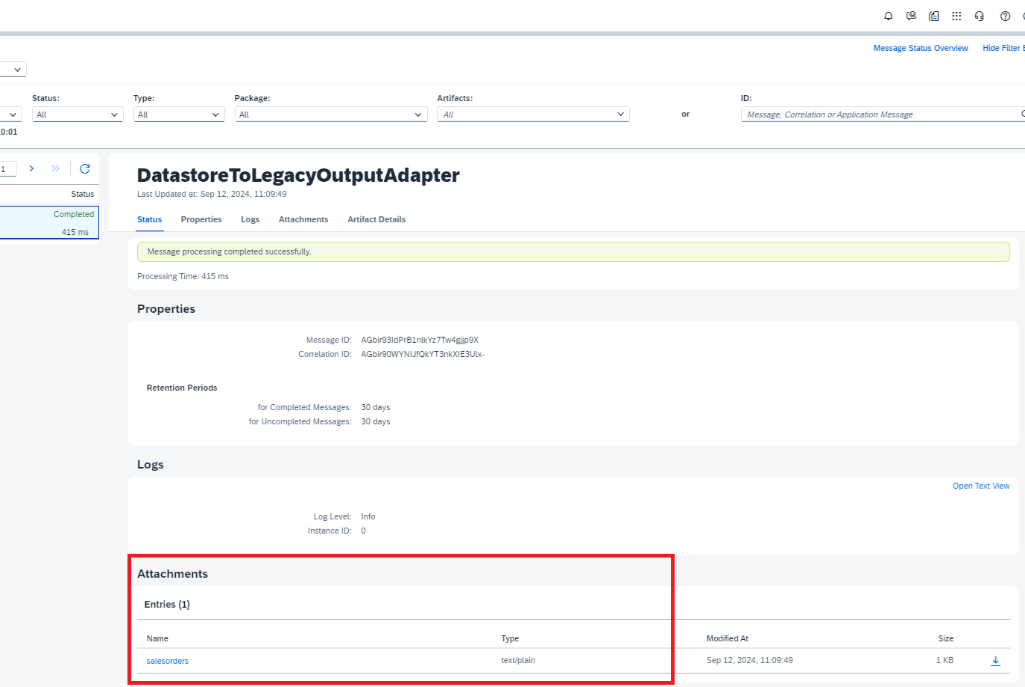


* For this IFlow you will need to configure your SFTP details and credentials. If you are not able to provide your own SFTP server, feel free to edit the IFlow, remove the SFTP adapter and instead add the Groovy script AggregatedSalesOrdersToAttachment.groovy in a Groovy script step to the end of your IFlow.

* The Timer of the IFlow is by default set to “execute once”. Therefore, it will run only one time after it is deployed. Make sure you have sent some events into the datastore before deploying the IFlow.

You should either see a flat file appear in the SFTP directory you specified or if you decided not to use SFTP you can check the Monitoring of the Integration Suite and see the flat file attached to the IFlow run there.



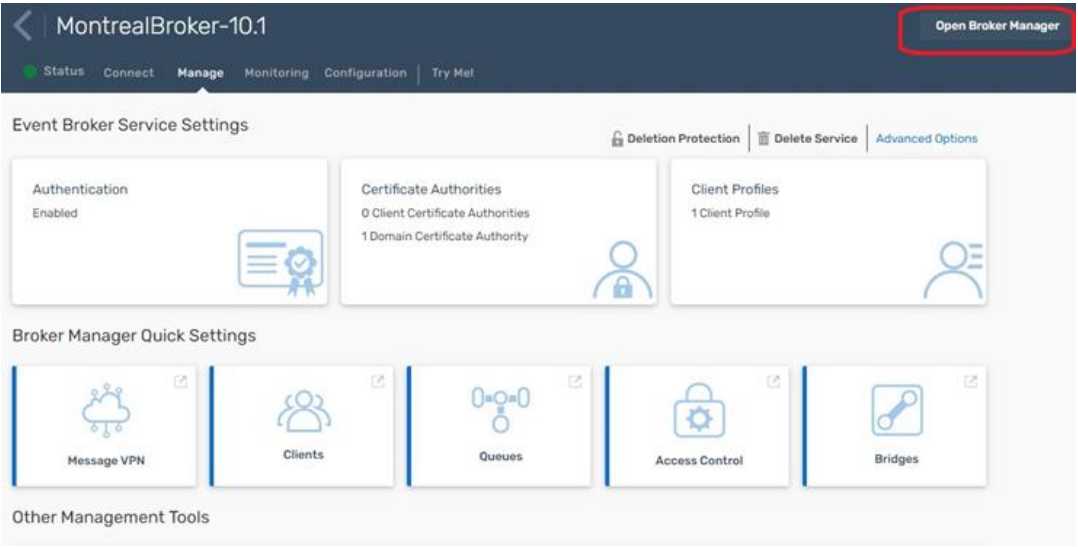
# Security and fine-grained access control to topics (optional)

For this section and in general, we would like to take this opportunity to answer a common question. Is it possible to prevent clients from publishing to certain Topics and/or subscribing to certain topics? **The answer is absolutely!**

We would like everyone to experience an Access Control List and how it can be used to control what is published or subscribed.

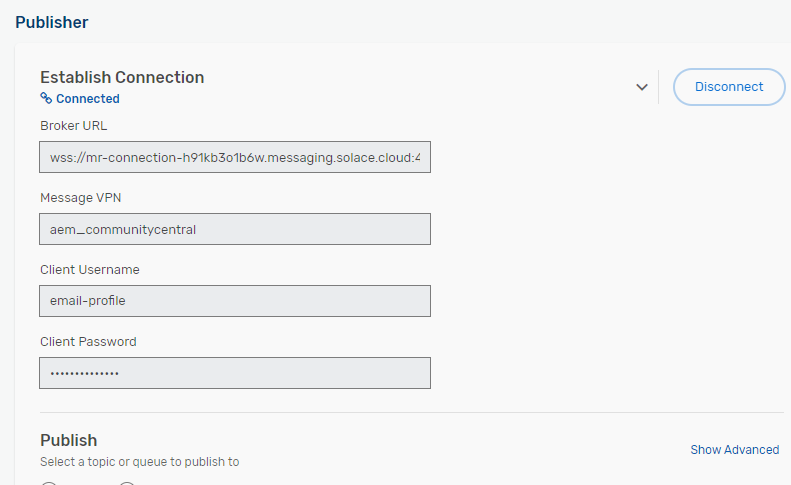
### **Experimenting publishing and subscribing to protected topics**

From the console, we need to navigate to the broker manager. You can get there by either clicking on the "Open Broker Manager" button or clicking any of the Tiles labelled "Clients", "Queues", "Access Control".

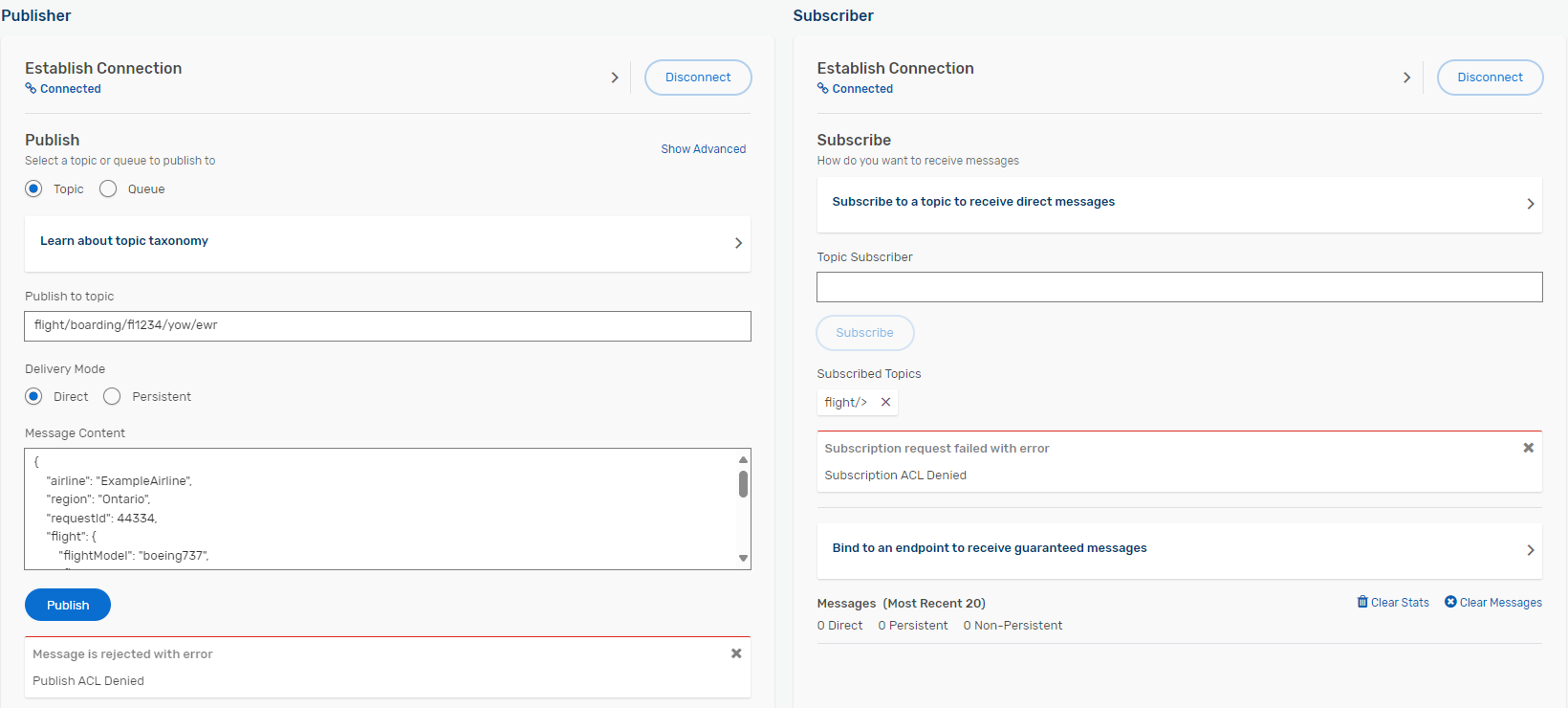


You will then see the Broker Manager Screen and on the left you will find the “Try-Me” test client. Click on it, to reveal the information you must provide to connect: For this screen, you will be trying to connect to our Broker (Note: We are doing this because your iFlow on Day 3 will send an Event to our broker to send an email) where we have created an ACL to limit what you can do and on what topics you can publish. The information you will use is as follows:

*Broker URL: tcps://mr-connection-h91kb3o1b6w.messaging.solace.cloud:55443  
Message VPN: aem\_communitycentral  
Client Username: email-profile  
Client Password: \*\*\*\*\* - provided during the course*



Once you have entered in the connectivity information, you should see the "Connected" message in blue. Once connected, change nothing and hit "Publish", you should immediately see the "Publish ACL Denied" on this action because the ACL will not permit you to complete this action. You can try the exact same thing with a subscription as well.



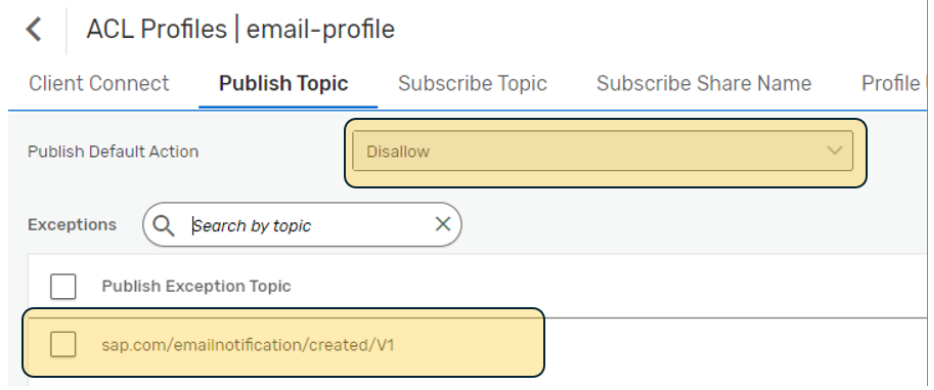
Now, lets head back to the "Publisher" and change the "Topic" to *sap.com/emailnotification/created/V1*

You can again use the content of the ExampleSOEvent.json we provided for exercise 3. Make sure to replace the email address with an actual address prior to hitting the publish button.

If you have entered the topic and message body correctly, you should see that 1 message has been published. So how did we do that? The magic happens in the ACL Profile as shown next.

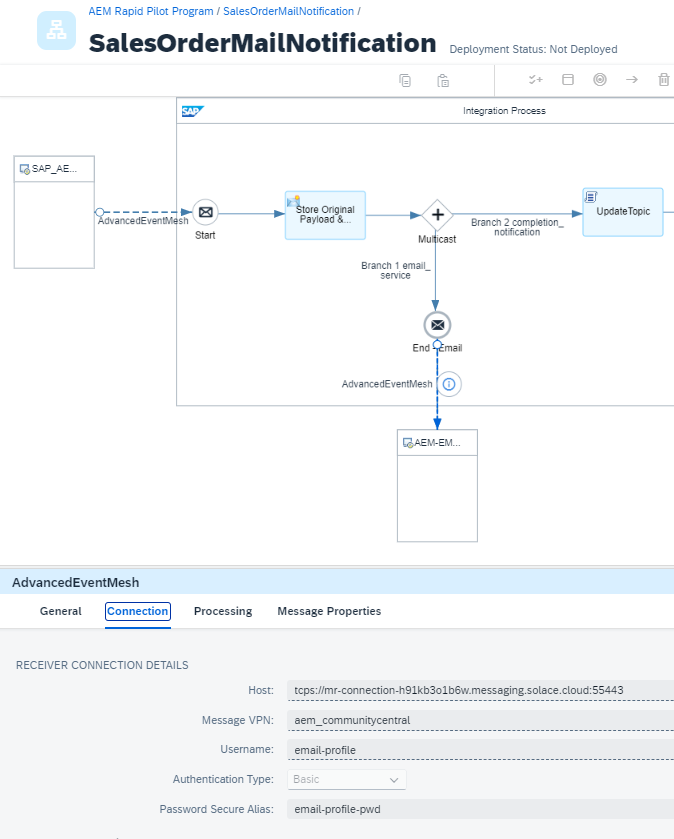
### **Broker Topic ACLs**

We have changed the Default Publish Action to be Disallow. In other words, unless we specify an exception, the user profile associated with this ACL cannot publish anything by default. In this case, as you can see, we have listed one exception.



For the subscription settings, it's very simple: We specify the Default Action is "Disallow" and do not provide any exceptions. AKA, this ACL does not permit any subscriptions.

Now that you have this understanding, you will see when you configure/deploy this iFlow why we have a Username "email-profile". It has been assigned the ACL email-profile so the iFlow can publish to our broker but only on that topic. Everything else is prohibited by the email-profile user ACL, so it can only be used for this single purpose. See below screenshot for where you would have used these credentials in your IFlow.



As you can see, broker ACLs are a quite powerful tool to tightly control access to the broker and its topics. You can separately control publish topics and subscribe topics and even IP address ranges that clients are allowed to connect from. In addition to topic ACLs, remember that queue access is controlled by the queue ownership model and the "other permission”.