Intune Reports Module Tutorial

OEA Curated

Date Published: November, 2021

This document's content is governed by Microsoft's Legal Notices for open source contributions.

Table of Contents

1. Initial Module Setup

i. Update/Install the latest OEA framework

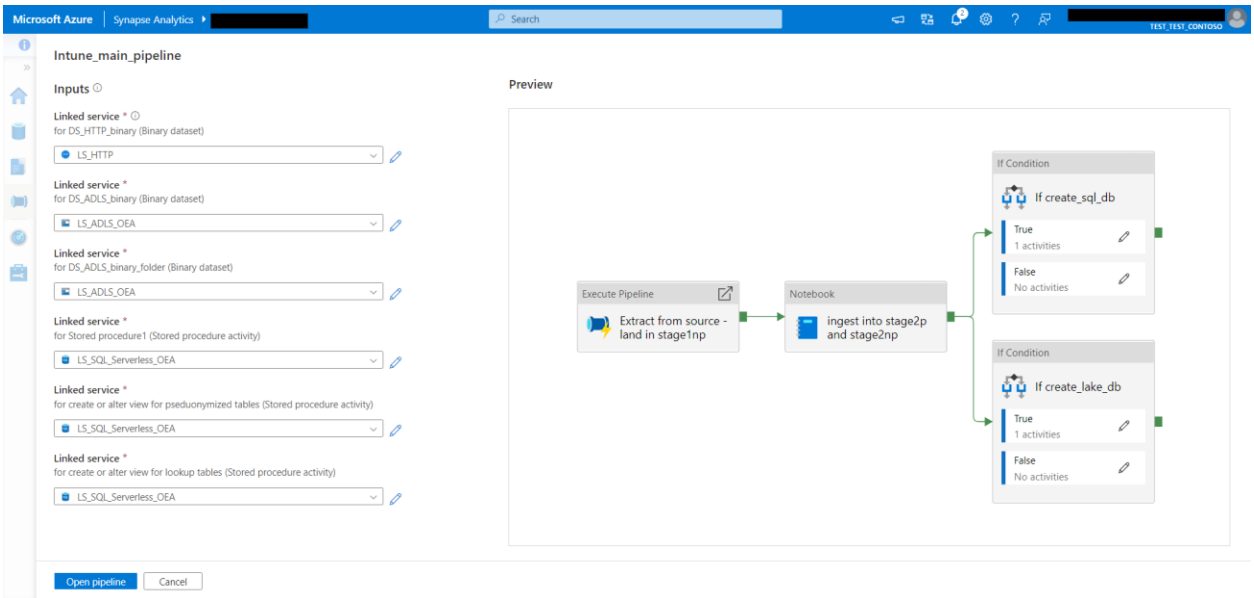
- Make sure that the Azure Synapse environment you're using is up to date. At the date of this tutorial creation, you should be using OEA framework v0.5. This can be done by following the steps outlined on the main OEA GitHub repository landing-page readme.
- You should expect to see many sample OEA pipelines and notebooks in your synapse workspace (ensure that you've followed the steps also outlined/assets provided in the OpenEduAnalytics GitHub under framework).

ii. Cloning OEA GitHub Repository via GitHub Desktop

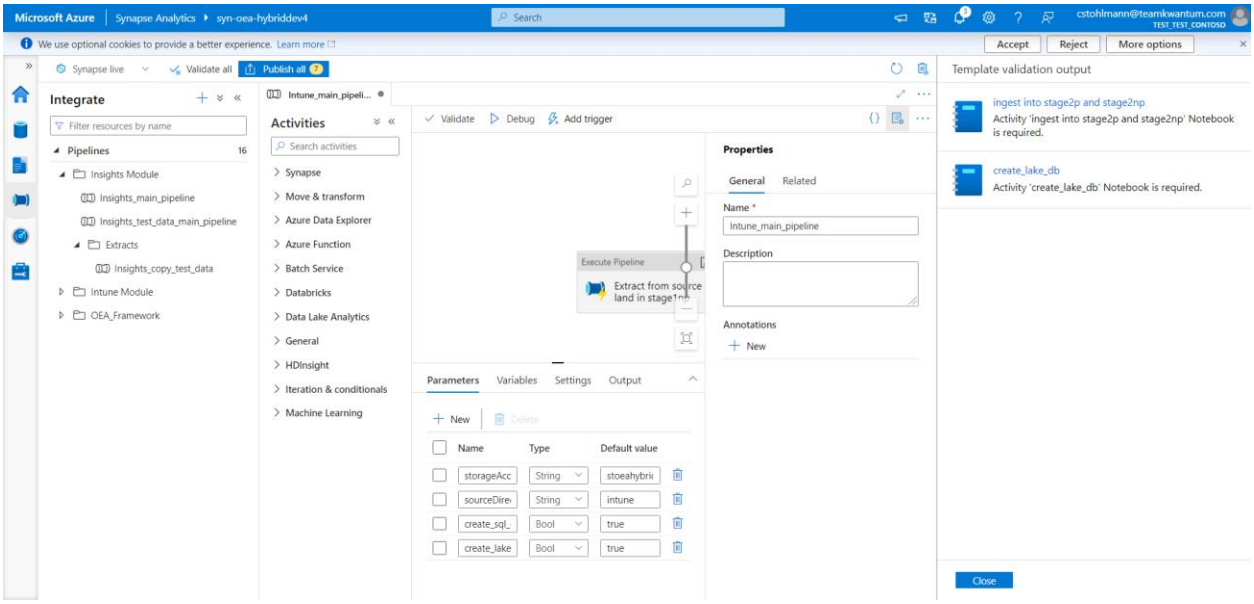
- To first familiarize yourself with the Intune module and what the data that Microsoft Intune Reports has to offer: start by cloning the entire OEA GitHub repository through your own GitHub Desktop.
    - If you've already done this from previous modules, make sure you have the latest updates to the repository by clicking "Pull origin" or "Fetch origin".

iii. Uploading and Using Intune Module assets into Azure Synapse

- Start by importing the two notebooks for this module: "Intune_py" (the class notebook defining the functions for the Intune module), and "Intune_module_ingestion" (the OEA connector notebook for the module).
    - And then connect your Intune_module_ingestion notebook to a spark pool you've already created. You may need to also import "OEA_py" and "OEA_connector" if not automatically/already imported. Connect the OEA_connector notebook to your spark pool as well.
- Next, import the "Intune_main_pipeline" template, under the Integrate section. The "Intune_copy_test_data" pipeline should automatically be imported.



- From here, you will need to properly connect your linked services to the pipeline (note: make sure you've run the shell setup script under framework to use the following linked services); the first one should be "LS_HTTP", then the next two should be "LS_ADLS_OEA", and the last three should be "LS_SQL_Serverless_OEA".
    - You will then be prompted to connect a notebook for the activities "ingest into stage2p and stage2np" and "create_lake_lb". For "ingest into stage2p and stage2np", connect the Intune_module_ingestion.ipynb notebook. For "create_lake_db", open up the pipeline that was imported from the template (it will probably be renamed something like "create_lake_db1" if you already have one a pipeline here) and connect the "OEA_connector" notebook from the OEA framework.



- Make sure the parameter "storageAccount" in the Intune_main_pipeline is changed to match your Synapse workspace's storage account name, and the parameter "timezone" is changed to match your current timezone (for the sake of folder naming)
- Now you're ready to trigger the "Intune_main_pipeline". This should take several minutes, and will automatically pull in the Intune test data, run the Intune module notebook, ingest and transform the data, and then create two databases with the Intune devices tables: s2_intune (the lake database), and sqls2_intune (the SQL database).

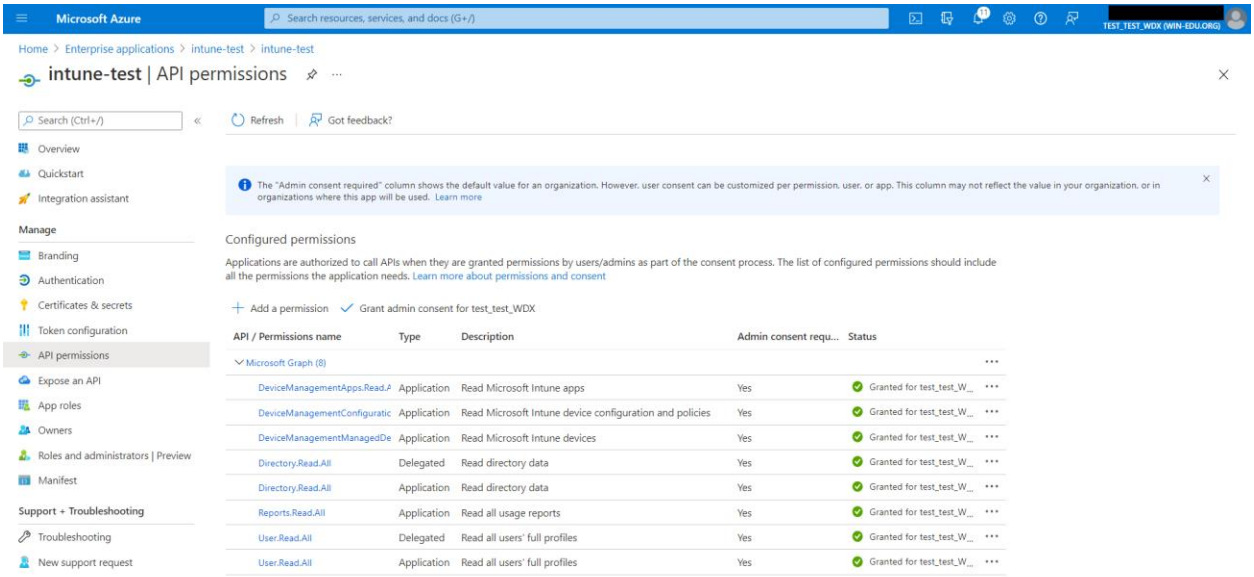
iv. Using the Intune PowerBI Dashboard Template

- After the pipeline has completed running and created the Spark databases, you can open your locally downloaded Intune PowerBI Dashboard Template.
    - The dashboard initially is using Import. You'll want to use a DirectQuery of the SQL database sqls2_intune; click here for information on how to do this.
    - For the sake of replicating the dashboard with the test data, use the "devices_pseudo_refined" table and corresponding data with the dashboard.
- You can interact with this dashboard to gain understanding of what this template can provide within the scope of this module.

2. Intune Production-Level Data Pipeline Setup

i. Create a new app registration for Microsoft Graph API and Intune Reports

- In your Azure environment, first go to AAD -> New registration and create a new app registration specifically for accessing the Graph API for Intune from Synapse.
- Then click “API Permission” -> “Add Permission” and select Microsoft Graph -> select Application Permissions -> choose the “User.Read.All” permission, and click on the “Add permissions” button.
- Follow the same steps above, except now add the “Directory.Read.All”, “Reports.Read.All”, “DeviceManagementApps.Read.All”, “DeviceManagementConfiguration.Read.All”, and “DeviceManagementManagedDevices.Read.All” permissions for Graph, as application permissions.
- After adding these permissions, you should see something similar to the following screenshot in your own Azure Portal:



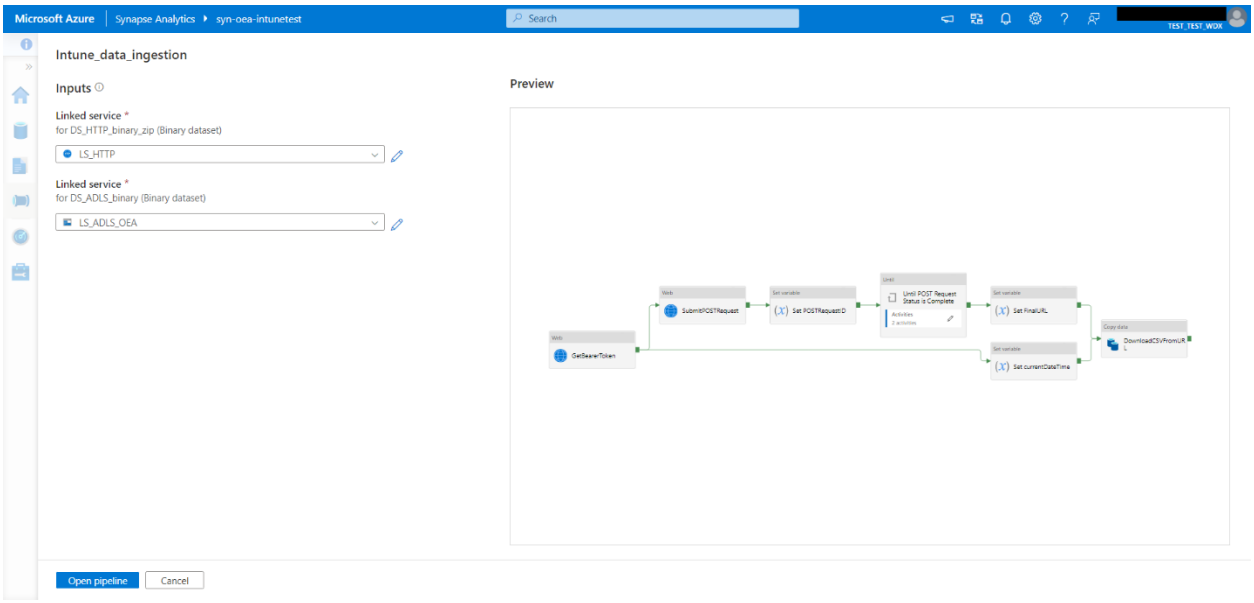
  - Click on the “Grant admin consent” link once these permissions have been successfully added, so that you can verify all permissions show a status of “Granted” under the status column.
- Now go to “certificates & secrets” and add a new client secret -> copy that value.

ii. Navigating to your Microsoft Intune/Graph app registration

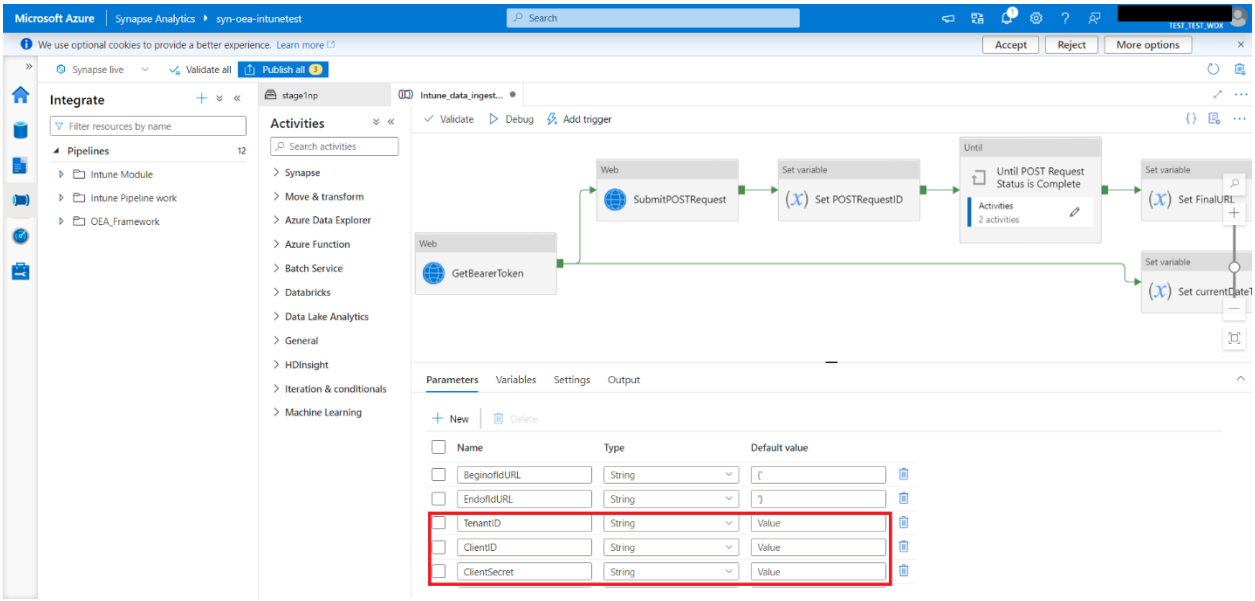
- If you lose the place of your new app registration, a quick way to find it is by searching for “Enterprise Applications” -> under “All applications” find your newly created app and click on it -> under “Single sign-on” click on the link for name of your new app in blue. This is the landing page for this application.

iii. Using the pipeline template in Synapse for actual Intune devices data ingestion

- First, start by deleting all test data previously used. The quickest way to do this is deleting the folder “intune” from stage1np, stage2p, and stage2np.
- Next, you will be importing the other pipeline template under “Extracts” on GitHub, titled “Intune_data_ingestion”.



- From here, you will be prompted to connect two linked services: one for “DS_HTTP_binary_zip” (a binary integration dataset for the HTTP call to download and unpack the zip for the CSV after the POST request is completed), and one for “DS_ADLS_binary” (a binary integration dataset that lands the unpacked-CSV in stage1np of the data lake). You should connect “LS_HTTP” to DS_HTTP_binary_zip and “LS_ADLS_OEA” to DS_ADLS_binary, as shown below.



- After successfully importing the Intune_data_ingestion pipeline template, navigate to the pipeline parameters (this can be done clicking on any of the whitespace in the pipeline GUI activities). There are a total of 6 parameters (as shown below). Leave “BeginofIdURL” and “EndofIdURL” as is, but you will need to edit “timezone” to match your own timezone (for the sake of folder naming) and add “TenantID”, “ClientID”, and “ClientSecret” manually.
    - For the “ClientSecret”, paste the value from when you created the client/application secret under this “ClientSecret”.
    - Navigate to application landing page in Azure, and copy the “Application (client) ID” -> navigate back to the Intune pipeline in your Synapse environment and paste this value under “ClientID”.
    - Lastly, go back to that app landing page in Azure, and copy the “Directory (tenant) ID” -> navigate back to your Synapse again, and paste this in the pipeline template under “TenantID”.
- Make sure to publish/commit these changes.

iv. Running the Intune_main_pipeline

- Now that the Intune_data_ingestion pipeline is ready for use, go back to the Intune_main_pipeline, and change the connection of the pipeline trigger-activity “Extract from source – land in stage1np” to “Intune_data_ingestion” in “Settings”.
    - Note: you may need to copy the default parameter values shown from the Intune_data_ingestion pipeline, into the values for the pipeline trigger-activity. Just copy and paste these values into their corresponding fields.
- After doing so, you are good to trigger the main pipeline, but now using your own data for ingestion. The run may take a few minutes. If you run into any errors, make sure to go back through the steps and make sure you filled out the information properly. Also make sure all necessary notebooks are connected to spark pools (i.e. Intune_module_ingestion and OEA_connector).
    - If you’re running into an error with the lake database creation, this module references the OEA_connector notebook, which calls the ContosoSIS_py notebook (but doesn’t use any of its functions). Make sure the ContosoSIS_py is imported to your Synapse workspace.
- You can then connect the PowerBI dashboard to the SQL database using the Serverless SQL database ID (you will probably want to use the devices_pseudo_refined table since this contains the additional columns and data analysis).