|  |  |
| --- | --- |
| Defender Masterclass Lab  Automated Incident Report with Power Automate | Abstract  This lab document contains a step-by-step guide to create a Power Automate Flow that will generate an Incident Report each time a new Alert lands in Microsoft Defender for Endpoint.  James.Graham@microsoft.com  Written for Microsoft Defender Masterclass Series – a series of events for Microsoft Partners created by James Graham |

Contents

[Part 1 - Getting Started 2](#_Toc74065525)

[Part 2 – Create a new App Registration with access to APIs 3](#_Toc74065526)

[Summary 9](#_Toc74065527)

[Part 3 – The Incident Report Template 10](#_Toc74065528)

[Store the template in SharePoint. 12](#_Toc74065529)

[Summary 14](#_Toc74065530)

[Part 4 – Power Automate 15](#_Toc74065531)

[Create Flow and Trigger 15](#_Toc74065532)

[Create Variables 17](#_Toc74065533)

[Obtain Auth Tokens 20](#_Toc74065534)

[Get Alert and Incident Data 26](#_Toc74065535)

[Populate our String Variables 39](#_Toc74065536)

[Populate the Incident Report 51](#_Toc74065537)

[Part 5 - The Test! 59](#_Toc74065538)

[Lab Complete 61](#_Toc74065539)

# Part 1 - Getting Started

Please ensure you have completed the lab prerequisites:

Located at the GitHub Repository - [aka.ms/defendermasterclass-repo](https://aka.ms/defendermasterclass-repo)

<https://github.com/JamesGrahamMSFT/DefenderMasterclass1/blob/main/Defender%20Masterclass%20-%20Labs%20Getting%20started.pdf>

The getting started guide provides step by step instructions to create a demo tenant. You can then use this demo tenant to complete this lab.

I have also created a video that shows the steps necessary to complete the prerequisites:

<https://youtu.be/btXpclS23Po>

Subscribe to our channel for future recordings and updates. [Microsoft Defender Masterclass - YouTube](https://www.youtube.com/channel/UCbcgrDBXvC1oC5ivKIrzshQ)

To complete this lab, you essentially will require an M365 Demo Tenant with Microsoft Defender for Endpoint. You will also want an active Evaluation Lab, which provides the ability to generate an alert/incident to produce the Incident Report.

Graphical user interface, text, application

Description automatically generated

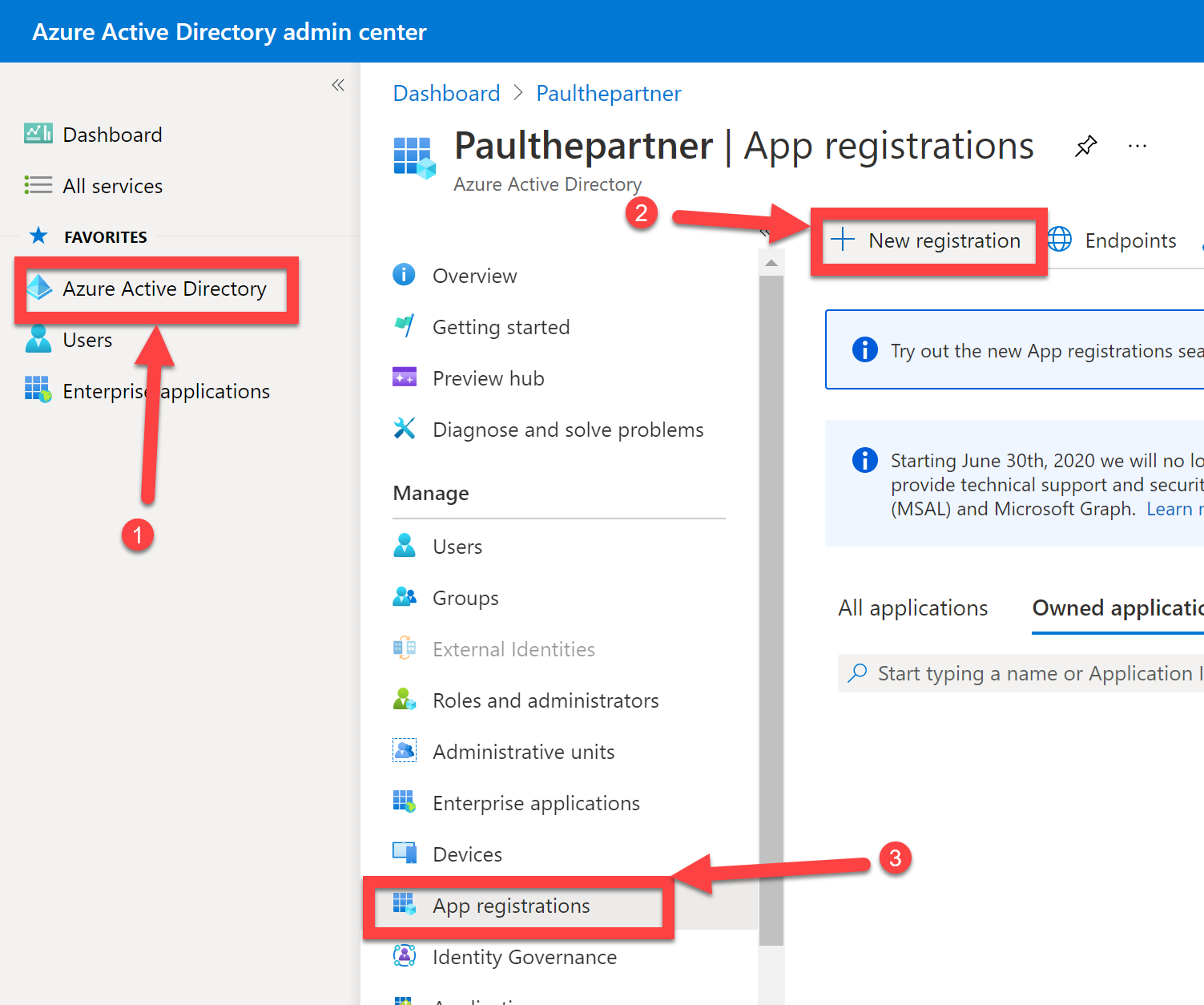
# Part 2 – Create a new App Registration with access to APIs

In this part we walk through the steps to create an app registration. The app registration is required to obtain an auth token and access the APIs required to populate the Incident Report.

1. Create Azure AD App Registration

***In this initial step we will create an Azure AD App Registration that contains the permissions required for you to interrogate the Defender for Endpoint and Microsoft Defender APIs for alert and incident information.***

1. Navigate to the Azure Active Directory admin center at <https://aad.portal.azure.com/> and login using your administrator credentials.
2. Select **Azure Active Directory**, select **App Registrations on the right-hand side**, click **New Registration.**



1. Enter a suitable **Name** for your App Registration, e.g. “MDE App”.
2. Under Supported account types – Select Accounts in this organizational directory only (*for the purpose of this lab we are only focused on a single tenant, although if you wanted you could modify this an create a multitenant app*)
3. Leave the Redirect URI blank – not required for this lab.

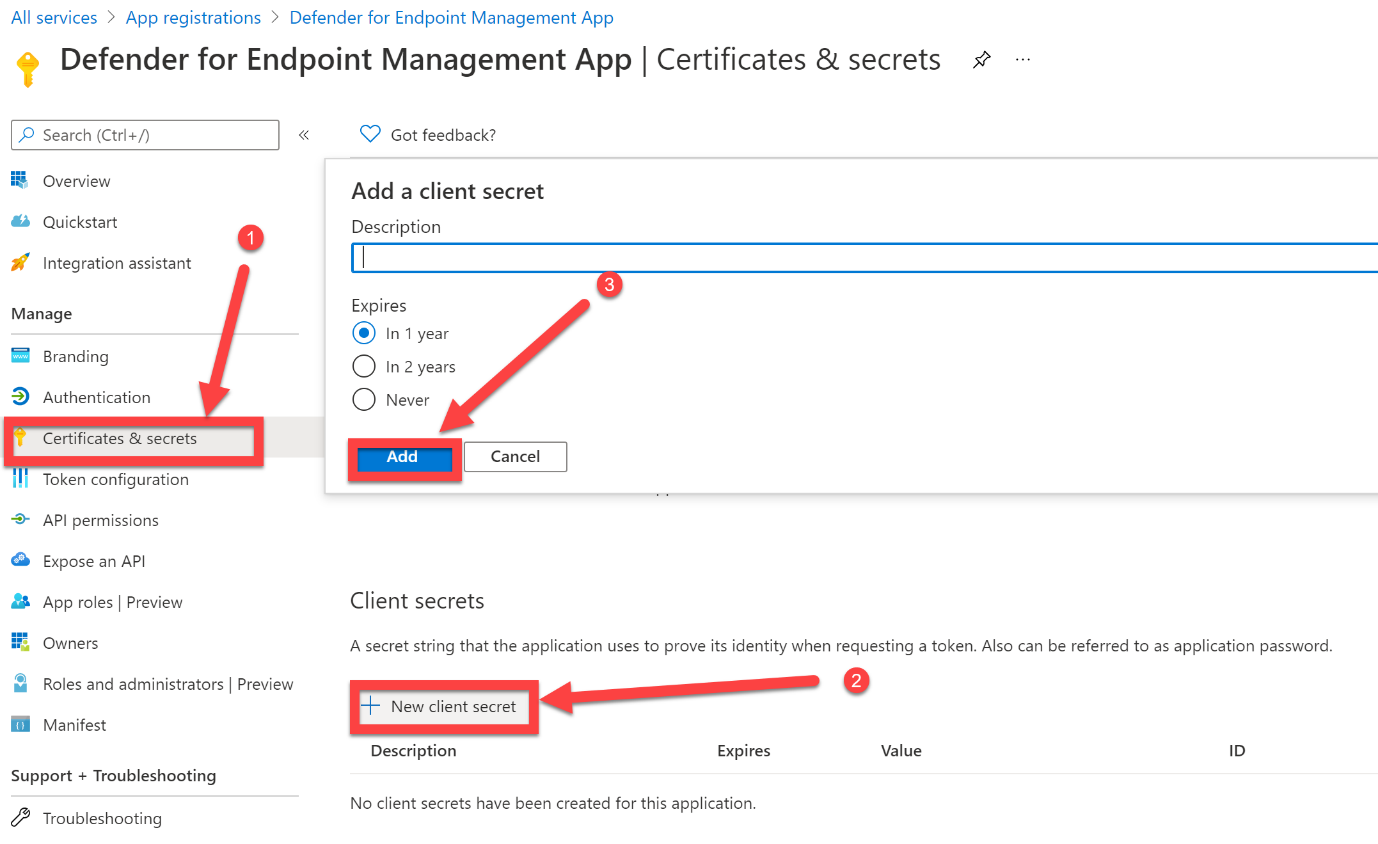
Graphical user interface, text, application, email

Description automatically generated

1. Click **Register.**
2. Capture the Application (client) ID, displayed in GUID format, and store it inside a notepad, you will need to use this later in the lab. It should look like:



1. Also take a copy of the Directory (tenant) ID and store in notepad.
2. We now need to create a secret for our application, to allow us to authenticate as the application, when accessing the APIs to retrieve alert/incident information.
3. Click **Certificates & secrets**, select **New client secret**, click **Add.**

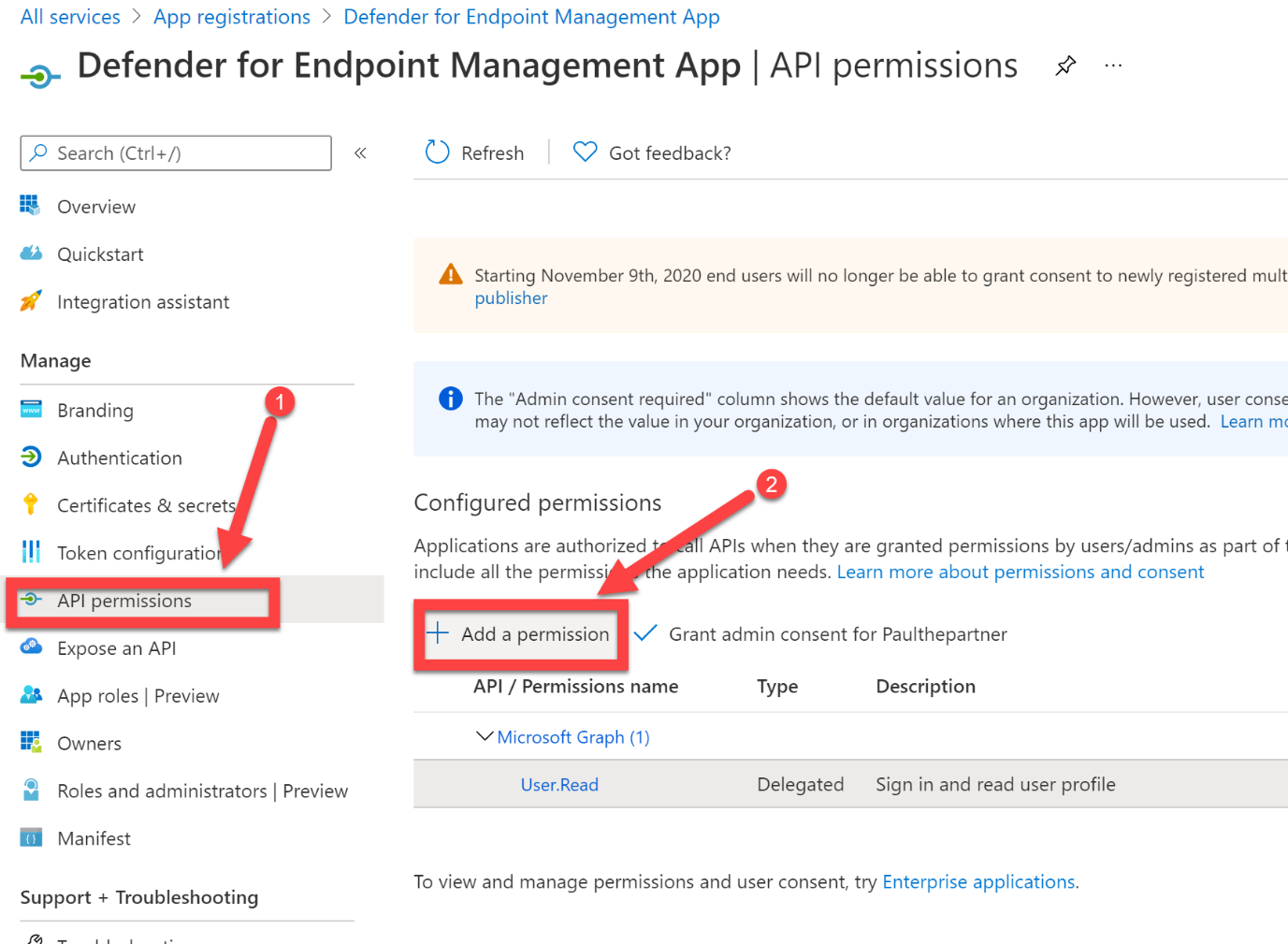


1. The Client secret will then be generated and will looks like a random string of letters and numbers. Capture the **Value** displayed and store it alongside the Application (client) Id you captured.

This will then be used to authenticate your application and receive an access token from Azure Active Directory

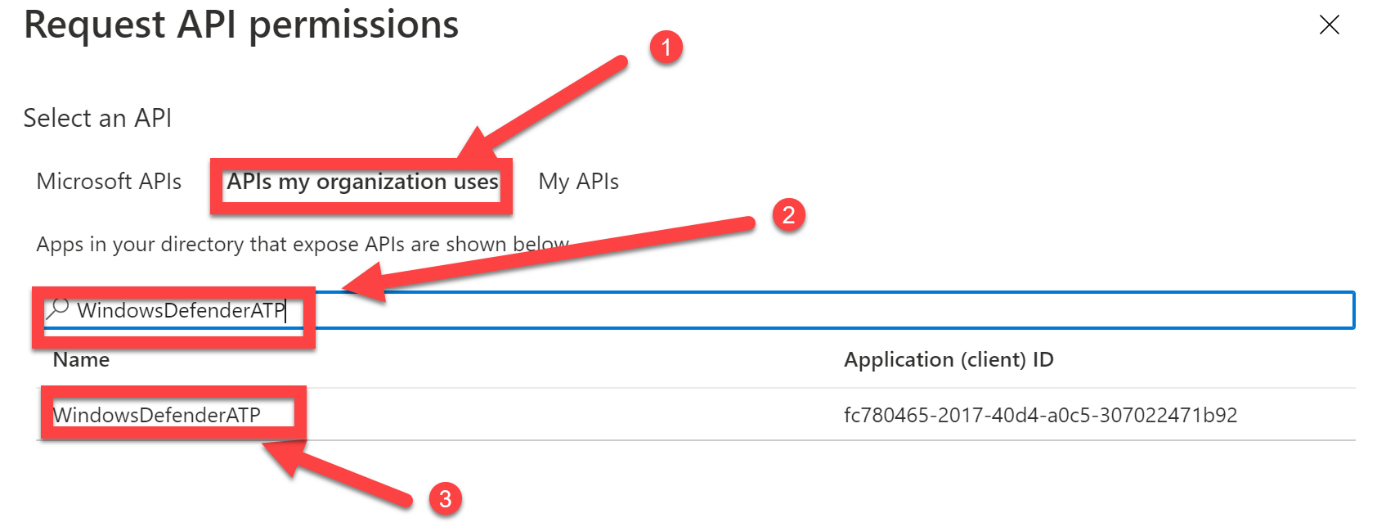


1. Select **API permissions**, click **Add a permission.**

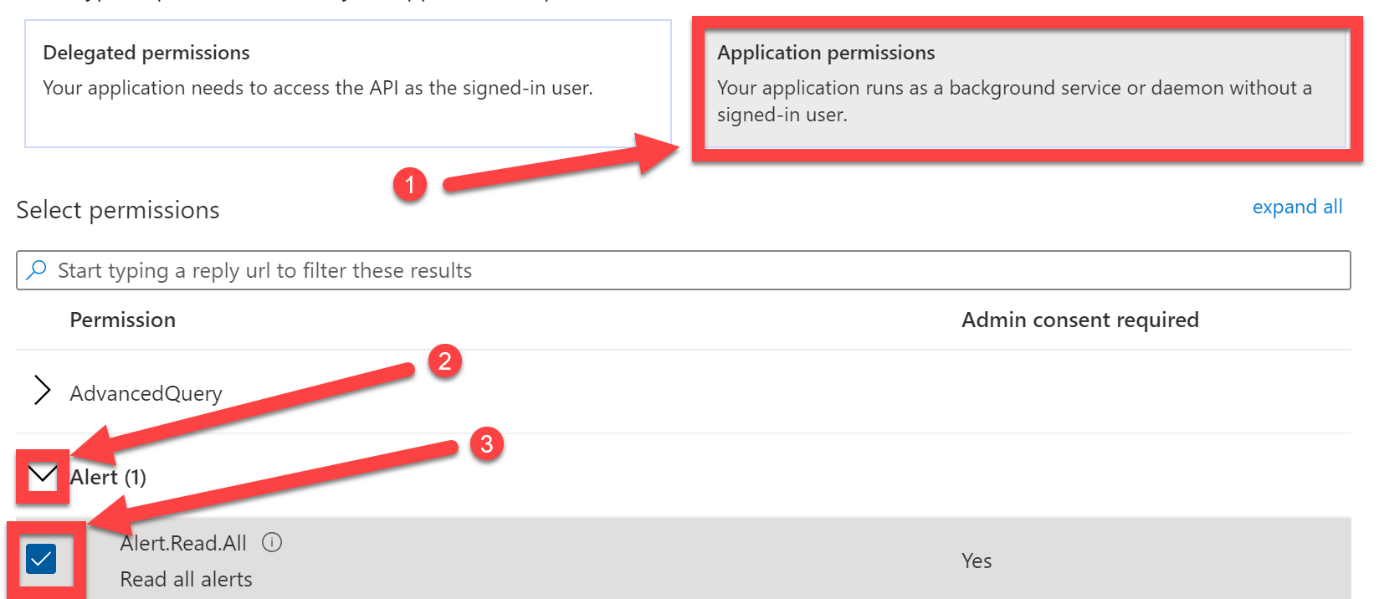


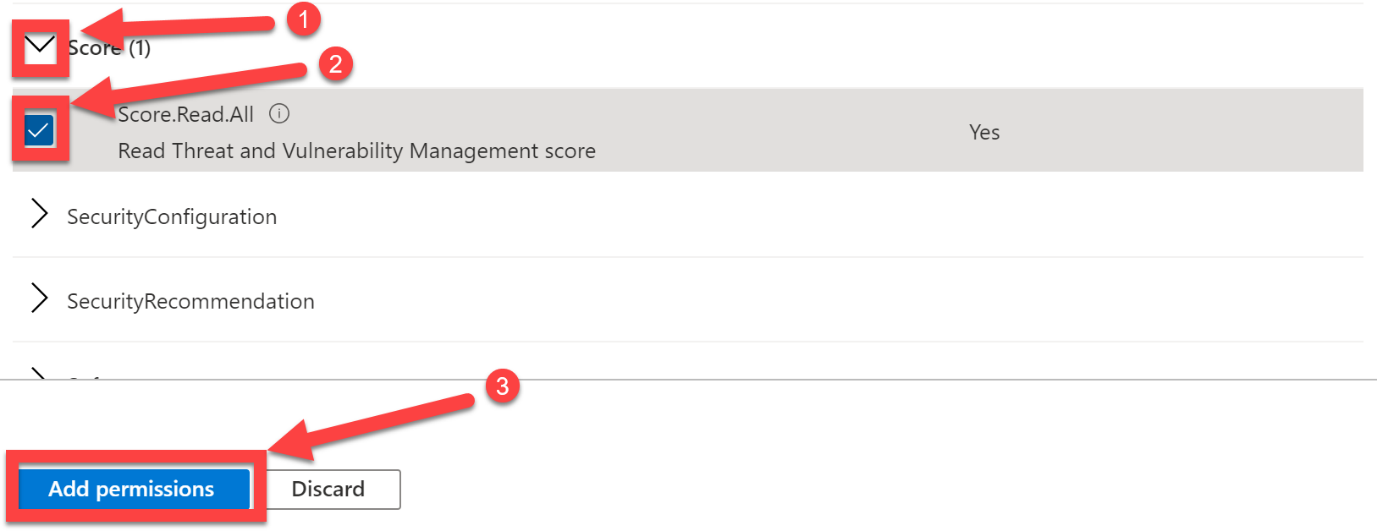
1. Select **APIs my organization uses** and in the search dialog box enter **WindowsDefenderATP**, select **WindowsDefenderATP**

*(note: if WindowsDefenderATP does not appear, then you are using a tenant that is not currently licensed for Defender for Endpoint. You will need to sign up for a Defender for Endpoint Trial, this can be achieved with an M365 E5 trial. Once you have activated a trial, go to securitycenter.microsoft.com to setup the tenant, and then return to this step once completed – if you followed the Labs Getting Started guide, you should already have Defender for Endpoint configured and available).*



1. Select **Application permissions**, expand **Alert** and select **Alert.Read.All**



1. Scroll down, expand **Score**, select **Score.Read.All**, click **Add permissions**
2. Repeat the process to add the following permissions for WindowsDefenderATP:
   1. Alert.ReadWrite.All
3. Also, we require the following permissions added for Microsoft Threat Proection:
   1. Incident.Read.All
   2. Incident.ReadWrite.All

[List incidents API in Microsoft 365 Defender | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender/api-list-incidents?view=o365-worldwide#permissions)

1. Once Added – click on “Grant Admin Consent for <tenant name>”.
2. If successful, all should look as follows:

Graphical user interface

Description automatically generated

1. You have now successfully configured your App Registration and it is now ready for you to ask customers to consent to the permissions you have just selected.
2. Finally, if not done already - we require the AAD Tenant ID – locate this in the Tenant Information box located on the main AAD Overview Blade or this is located just below your App (Client) ID:

Graphical user interface, text, application

Description automatically generated

1. Store this value in notepad alongside the App ID and Secret Value.

## Summary

At the end of this section, we should have the following:

1. An App Registration with access to the follow APIs:
   1. Windows Defender ATP
      1. Alert.ReadWrite.All
      2. Alert.Read.All
   2. Microsoft Threat Protection
      1. Incident.Read.All
      2. Incident.ReadWrite.All
   3. Microsoft Graph (default, ok to leave this):
      1. User.Read
2. The Application (client) ID stored on notepad.
3. The Client Secret Value stored on notepad.
4. The Tenant ID stored on notepad.

# Part 3 – The Incident Report Template

For the purposes of this lab, I have created a template that can be used.

The template is located on the GitHub repo:

<https://github.com/JamesGrahamMSFT/DefenderMasterclass1/blob/main/Defender%20Masterclass%203%20-%20Incident%20Report%20Template.docx>

Please download the file.

Graphical user interface, text, application

Description automatically generated

To create the template, I have enabled Developer mode in Word – this allows you to create variables that can be populated later.

Graphical user interface, text, application

Description automatically generated

1. Enable Developer mode in Word – Open Word, navigate to Options and enable the Developer tab:

A screenshot of a computer

Description automatically generated with medium confidence

1. Open the Defender Masterclass 3 – Incident Report Template document.
2. Navigate to the Developer tab – enable Design Mode.
3. Once enabled you will be able to see all the fields that I have created.

A screenshot of a menu

Description automatically generated with medium confidence

1. You don’t need to edit this document – this is simply to show you how I created the template.
2. Each field is a Plain Text Content Control – this is the control that works with Power Automate (more on this later).
3. If you want to create your own fields later:
   1. Navigate to the area of the document where you want the field.
   2. Click on Plain Text Content Control button

Graphical user interface, application

Description automatically generated

* 1. This will generate the field Text

     Description automatically generated
  2. Optional – insert description text within the field.
  3. Whilst the cursor is active inside the field – click on Properties in the Developer Tab.

Graphical user interface, text, application, email

Description automatically generated

* 1. From here we can give the field an appropriate name.
  2. Important - If the output will be spread over multiple lines, we must tick the box at the bottom – Allow carriage returns.

## Store the template in SharePoint.

1. For us to use this template later we must first store it in the tenant.
2. In this lab we will use SharePoint.
3. Navigate to the M365 Admin Center with your administrator credentials - [Microsoft 365 admin center - Home (office.com)](https://portal.office.com/adminportal/home?#/homepage)
4. Click on … Show all.

Graphical user interface, website

Description automatically generated

1. Click on SharePoint – under Admin Centers to navigate to the SharePoint Admin Center.

Graphical user interface, application

Description automatically generated

1. Within SharePoint Admin Center – navigate to Sites -> Active Sites.
2. We will use the SOC Team site to host our template.
3. Click on SOC Team in the list of Active Sites – on the left tab that opens, click on the URL.

Graphical user interface, text, application, email

Description automatically generated

1. Within the SharePoint site – Click on Documents.
2. Click on Upload – Files.

Graphical user interface, application

Description automatically generated

1. Select the Defender Masterclass 3 – Incident Report Template you downloaded earlier.
2. Once completed you should see the template file listed in the Documents:

Graphical user interface, text, application, email

Description automatically generated

*At this point some of you may be wondering (or not) why our template is saved as a .docx file and not a dotx file that is typically used for templates. This is because Power Automate only works with docx files and still treats them as a template when used.*

## Summary

In this part we looked at the Incident Report template. We downloaded and examined the document and explored how you could potentially edit this template or create your own.

At this stage it won’t be clear how this template will be used, but that will become clear later.

We should now have the following:

1. An Incident Report template in docx format.
2. A SharePoint site with the template uploaded to the Documents section.

# Part 4 – Power Automate

We now have all the prerequisites in place to build our Power Automate Flow.

* An M365 Demo Tenant with Microsoft Defender for Endpoint.
* The ability to generate an Alert/Incident within Microsoft Defender for Endpoint.
* An App Registration with access to required APIs.
* A report template stored within a SharePoint site.

## Create Flow and Trigger

1. Navigate to <https://flow.microsoft.com> and sign in with your Administrator credentials.

Graphical user interface, application

Description automatically generated

1. Click on My Flows.
2. Click on + New Flow -> Automated cloud flow.

Graphical user interface, application

Description automatically generated

1. Provide a name, e.g. Incident Report Flow.
2. Within the search bar for the flow’s trigger, start typing Defender.
3. Click on the Trigger when a new WDATP alert occurs. Graphical user interface

   Description automatically generated
4. Click on Create.
5. Click on the Sign In box within the Flow Trigger – sign in with Administrator credentials.

Graphical user interface, application

Description automatically generated

1. In the dialog box – consent to the permissions and accept.

Graphical user interface, text, application

Description automatically generated

1. We now have a trigger in place that will execute each time a new alert occurs in Defender for Endpoint.

## Create Variables

We must now create String variables that will be used to populate some of the fields in our Incident Report template.

1. Click on + New Step.
2. We want the Initialize Variable action – this can be found by typing “initialize” in the Search bar.
3. Enter name – AlertDetails.
4. Type – String
5. Value – leave blank.

Graphical user interface, application

Description automatically generated

1. Repeat steps 1-5 for DeviceDetails.

Graphical user interface, application

Description automatically generated

1. Repeat steps 1-5 for UserDetails.

Graphical user interface, text, application, email

Description automatically generated

1. Repeat steps 1-5 for ProcessDetails.

Graphical user interface, text, application

Description automatically generated

1. Repeat steps 1-5 for RegistryDetails.

Graphical user interface, application

Description automatically generated

1. Repeat steps 1-5 for FileDetails.

Graphical user interface, application

Description automatically generated

1. Repeat steps 1-5 for IPDetails.

Graphical user interface, application

Description automatically generated

1. Repeat steps 1-5 for MailMessageDetails.

Graphical user interface, text, application, email

Description automatically generated

1. Repeat steps 1-5 for MailboxDetails.

Graphical user interface, text, application

Description automatically generated

1. Repeat steps 1-5 for MailClusterDetails.

Graphical user interface, text, application

Description automatically generated

1. Repeat steps 1-5 for MITRETechniques.

Graphical user interface, text, application, email

Description automatically generated

We should now have 11 x Initialize Variable steps. Each of these strings will be populated later with Incident information that spans multiple elements and lines – hence we need to pack them into Strings before pushing them to the document.

There is a final variable we need to initialize. This is an integer and will be used to display the number of alerts associated with the incident.

1. + New Step.
2. Initialize Variable.
3. Name – AlertCount.
4. Type – Integer.
5. Initial Value – 0.

Graphical user interface, application

Description automatically generated

## Obtain Auth Tokens

In this section we must obtain an authentication token for the M365 Defender API and the Defender for Endpoint API.

[Create an app to access Microsoft 365 Defender without a user | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender/api-create-app-web?view=o365-worldwide#get-an-access-token)

[Create an app to access Microsoft Defender for Endpoint without a user | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/exposed-apis-create-app-webapp?view=o365-worldwide#get-an-access-token)

The reason must create two independent tokens is because the audience of each token request is different, and you cannot generate a single token for multiple audiences (aka scopes).

For this section we will be using the Tenant ID, App ID and Secret we noted earlier.

1. Click on +New Step.
2. In the search bar type - http.
3. Click on the HTTP (Premium) Action.

If this is a new tenant – you will now be prompted to start a free 90-day trial for Power Automate – click on Start Trial to continue.

Graphical user interface

Description automatically generated

1. Complete the HTTP Action as follows:
   1. Method – POST
   2. URI - https://login.microsoftonline.com/<insert tenant ID here>/oauth2/v2.0/token
   3. Headers (left box) - Content-Type
   4. Headers (right box) - application/x-www-form-urlencoded
   5. Body –

client\_id=<insert App ID here>

&scope=https://api.security.microsoft.com/.default

&client\_secret=<insert secret value here>

&grant\_type=client\_credentials

If done correctly, the HTTP box will look as follows (I’ve renamed mine so it’s clear):

Graphical user interface, text, application, email

Description automatically generated

The response back from this HTTP POST will be a JSON file that contains the auth token. We must now use a PARSE JSON action to provide us with the ability to call the auth token when needed.

To leverage the PARSE JSON action we need to create a schema so the action can read the JSON file. There are steps to achieve this but for the purpose of this lab, I have already completed those steps to save time and thus will provide you with the schema to populate the PARSE JSON action.

1. Click on + New Step.
2. Search for Parse JSON and select it from the available actions.
3. Select the Content box – the Dynamic Content dialog box will appear.
4. You want to select Body from your HTTP action (this is why I renamed my HTTP action, so it’s easier to locate).

Graphical user interface, text, application

Description automatically generated

1. Within the Schema box – paste the following:

{

"type": "object",

"properties": {

"token\_type": {

"type": "string"

},

"expires\_in": {

"type": "integer"

},

"ext\_expires\_in": {

"type": "integer"

},

"access\_token": {

"type": "string"

}

}

}

1. Remove the extra line that may get added when you paste the above into the Schema box.
2. It should look as follows: Graphical user interface

   Description automatically generated with medium confidence
3. Click on Save at the top right to ensure there are no errors in the Schema – when you click save, error checking takes place.
4. To be able to identify the correct auth token later it’s worth renaming this Parse JSON action to M365D Auth – click on … to achieve this. Graphical user interface, text, application

   Description automatically generated

We now need to repeat these steps to obtain an auth token for Defender for Endpoint – notice the change in scope for the auth token request.

1. Click on +New Step.
2. Search for https and select the HTTP action.
3. Complete the HTTP Action as follows – to save time I would recommend copying the contents from the previous HTTP request action box, but remember to use the different &scope value below:
   1. Method – POST
   2. URI - https://login.microsoftonline.com/<insert tenant ID here>/oauth2/v2.0/token
   3. Headers (left box) - Content-Type
   4. Headers (right box) - application/x-www-form-urlencoded
   5. Body –

client\_id=<insert App ID here>

&scope= https://securitycenter.onmicrosoft.com/windowsatpservice/.default

&client\_secret=<insert secret value here>

&grant\_type=client\_credentials

Should look as follows:

Graphical user interface, text, application, email

Description automatically generated

1. Repeat steps 5-13 to create the associated PARSE JSON action – the schema will be the same.
2. In the content box – ensure you select the Body from the new HTTP request action you just created.
3. Copy the same schema from step 9 – or copy/paste it directly from the flow.

Graphical user interface, text, application

Description automatically generated

1. Click Save when done.

We now have the necessary Auth Tokens to call APIs from M365 Defender and Defender for Endpoint.

## Get Alert and Incident Data

In this section we will use the Auth Tokens we’ve obtained to:

* Obtain more information about the Alert that has been generated – we are after the Incident ID. We achieve this by making a call to the Defender for Endpoint API and passing the Alert ID.
* Use the Incident ID we have obtained to get the detailed Incident information to populate our Incident Report. We achieve this by making a call to the M365 Defender API and passing the Incident ID.

[Get alert information by ID API | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/get-alert-info-by-id?view=o365-worldwide)

[Get incident API | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender/api-get-incident?view=o365-worldwide#http-request)

1. Click on +New Step
2. Search for http and select the HTTP action.
3. Complete the HTTP Action as follows:
   1. Method – GET
   2. URI - https://api.securitycenter.windows.com/api/alerts/<AlertID> - Obtain AlertID by scrolling to the bottom of the Dynamic Content and selecting it from the trigger.

Graphical user interface, text, application, email

Description automatically generated

* 1. Headers (left box) – authorization
  2. Headers (right box) - Bearer <access token> - Obtain this by selecting the access\_token from the MDEAuth Parse JSON action performed in step 19 in previous section.

Graphical user interface, text, application, email

Description automatically generated

1. The completed HTTP should look as follows (note – there should be a single space between Bearer and the access\_token). Again, I’ve renamed mine so I can easily reference it in the next step.

Graphical user interface, text, application, email

Description automatically generated

The response will come back in JSON format, so again we must use the PARSE JSON action and supply the appropriate schema.

1. Click on +New Step.
2. Search for Parse JSON and select it from the available actions.
3. Select the Content box – the Dynamic Content dialog box will appear.
4. You want to select Body from your Get Alert Data HTTP action.
5. Paste the following schema into the Schema box:

{

"type": "object",

"properties": {

"@@odata.context": {

"type": "string"

},

"id": {

"type": "string"

},

"incidentId": {

"type": "integer"

},

"investigationId": {},

"assignedTo": {},

"severity": {

"type": "string"

},

"status": {

"type": "string"

},

"classification": {},

"determination": {},

"investigationState": {},

"detectionSource": {},

"detectorId": {},

"category": {},

"threatFamilyName": {},

"title": {

"type": "string"

},

"description": {

"type": "string"

},

"alertCreationTime": {

"type": "string"

},

"firstEventTime": {

"type": "string"

},

"lastEventTime": {

"type": "string"

},

"lastUpdateTime": {

"type": "string"

},

"resolvedTime": {},

"machineId": {

"type": "string"

},

"computerDnsName": {

"type": "string"

},

"rbacGroupName": {},

"aadTenantId": {

"type": "string"

},

"threatName": {},

"mitreTechniques": {

"type": "array"

},

"relatedUser": {},

"comments": {

"type": "array"

},

"evidence": {

"type": "array",

"items": {

"type": "object",

"properties": {

"entityType": {

"type": "string"

},

"evidenceCreationTime": {

"type": "string"

},

"sha1": {},

"sha256": {},

"fileName": {},

"filePath": {},

"processId": {},

"processCommandLine": {},

"processCreationTime": {},

"parentProcessId": {},

"parentProcessCreationTime": {},

"parentProcessFileName": {},

"parentProcessFilePath": {},

"ipAddress": {},

"url": {},

"registryKey": {},

"registryHive": {},

"registryValueType": {},

"registryValue": {},

"accountName": {},

"domainName": {},

"userSid": {},

"aadUserId": {},

"userPrincipalName": {},

"detectionStatus": {}

}

}

}

}

}

1. Remove the extra line at the bottom that may get added when you paste the above into the Schema box.
2. It should look as follows (I renamed mine):

Graphical user interface, text

Description automatically generated

1. Click on Save at the top right to ensure there are no errors in the Schema.

Within the JSON that gets returned will be the Incident ID – we now want to use this to obtain the Incident data.

Therefore, we must do one final HTTP and PARSE JSON action!

1. Click on +New Step.
2. Search for http and select the HTTP action.
3. Complete the HTTP Action as follows:
   1. Method – GET
   2. URI - https://api.security.microsoft.com/api/incidents/<Incident ID> - Obtain Incident ID from your previous Parse JSON action (step 6).

Graphical user interface, text, application, email

Description automatically generated

* 1. Headers (left box) – authorization
  2. Headers (right box) - Bearer <access token> - Obtain this by selecting the access\_token from the M365D Auth Parse JSON action performed in step 13 in previous section.

Graphical user interface, text, application

Description automatically generated

1. The completed HTTP should look as follows (note – there should be a single space between Bearer and the access\_token). Again, I’ve renamed mine so I can easily reference it in the next step.

Graphical user interface, text, application, email

Description automatically generated

The response will come back in JSON format, so again we must use the PARSE JSON action and supply the appropriate schema – I’ve done the hard work for you to create this schema.

1. Click on +New Step.
2. Search for Parse JSON and select it from the available actions.
3. Select the Content box – the Dynamic Content dialog box will appear.
4. You want to select Body from your Get Incident Data HTTP action.
5. Paste the following schema into the Schema box (this one is big and contains multiple arrays with some nested!):

{

"type": "object",

"properties": {

"@@odata.context": {

"type": "string"

},

"incidentId": {},

"redirectIncidentId": {},

"incidentName": {},

"createdTime": {},

"lastUpdateTime": {},

"assignedTo": {},

"classification": {},

"determination": {},

"status": {},

"severity": {},

"tags": {

"type": "array"

},

"comments": {

"type": "array",

"items": {

"type": "object",

"properties": {

"comment": {},

"createdBy": {},

"createdTime": {}

}

}

},

"alerts": {

"type": "array",

"items": {

"type": "object",

"properties": {

"alertId": {},

"incidentId": {},

"serviceSource": {},

"creationTime": {},

"lastUpdatedTime": {},

"resolvedTime": {},

"firstActivity": {},

"lastActivity": {},

"title": {},

"description": {},

"category": {},

"status": {},

"severity": {},

"investigationId": {},

"investigationState": {},

"classification": {},

"determination": {},

"detectionSource": {},

"detectorId": {},

"assignedTo": {},

"actorName": {},

"threatFamilyName": {},

"mitreTechniques": {

"type": "array",

"items": {}

},

"devices": {

"type": "array",

"items": {

"type": "object",

"properties": {

"mdatpDeviceId": {},

"aadDeviceId": {},

"deviceDnsName": {},

"osPlatform": {},

"version": {},

"osProcessor": {},

"osBuild": {},

"healthStatus": {},

"riskScore": {},

"rbacGroupName": {},

"firstSeen": {},

"tags": {

"type": "array",

"items": {}

},

"defenderAvStatus": {},

"onboardingStatus": {},

"vmMetadata": {}

}

}

},

"entities": {

"type": "array",

"items": {

"type": "object",

"properties": {

"entityType": {},

"evidenceCreationTime": {},

"verdict": {},

"remediationStatus": {},

"sha1": {},

"sha256": {},

"fileName": {},

"filePath": {},

"processId": {},

"processCommandLine": {},

"processCreationTime": {},

"parentProcessId": {},

"parentProcessCreationTime": {},

"accountName": {},

"domainName": {},

"userSid": {},

"detectionStatus": {},

"deviceId": {},

"registryHive": {},

"registryKey": {},

"registryValueType": {},

"ipAddress": {},

"url": {},

"aadUserId": {},

"userPrincipalName": {},

"mailboxDisplayName": {},

"mailboxAddress": {},

"clusterBy": {},

"sender": {},

"recipient": {},

"subject": {},

"deliveryAction": {},

"securityGroupId": {},

"securityGroupName": {},

"registryValue": {}

}

}

}

}

}

}

}

}

1. Remove the extra line at the bottom that may get added when you paste the above into the Schema box.
2. It should look as follows (I renamed mine):

Graphical user interface, text, application

Description automatically generated

1. Click on Save at the top right to ensure there are no errors in the Schema.

This is the last PARSE JSON we need to do. We now have all the information we need about the Incident and Alert to populate the Incident Report template.

The next challenge will be trawling through the incident JSON to pull out the information we require – that’s up next.

## Populate our String Variables

Looking at the Incident API documentation and/or the Schema above we can see the format of the Incident information that will be returned in the JSON.

[List incidents API in Microsoft 365 Defender | Microsoft Docs](https://docs.microsoft.com/en-us/microsoft-365/security/defender/api-list-incidents?view=o365-worldwide)

We receive some standard information about the Incident which can be easily pushed to our Incident Report Template fields, however there is also some arrays that contain elements such as tags, comments, and alerts. The alerts array then contains multiple arrays such as devices and multiple types of entities.

In the next steps we must loop through the Alerts associated with the Incident (and the sub-arrays contained within) and populate the strings we initialised at the start of the Flow.

1. +New Step.
2. Search for and select the “Apply to each” action.
3. The output we want to loop through is the Alerts array from the Parse Incident JSON step. From the dynamic content – select alerts from the long list of items under the Parse Incident JSON heading. Graphical user interface, application

   Description automatically generated
4. We now populate our first variable with Alert Information.
5. Click on Add and action within the Apply to each box.
6. Search for and select “Append to string variable”.
7. In the Name drop down select – AlertDetails.
8. Within the value box – complete as per the screenshot. Ensure you add a new line after the dashes. You can search for each element in the Search dynamic content – make sure you select the element from the Parse Incident JSON and not the Alert JSON.Graphical user interface, application

   Description automatically generated
9. Next step is increment the AlertCount variable – this will give us a count of the associated Alerts for this Incident.
10. Click on Add an Action under the previous action.
11. Search for and select “Increment variable”.
12. In the name dropdown select – AlertCount.
13. In the Value box type – 1. Graphical user interface, text, application, email

    Description automatically generated
14. Now we will loop through the Devices associated with the Alert – so therefore, we require a new loop inside the current loop.
15. Click on Add an action.
16. Search for and select “Apply to each” – you can rename this one if it helps, e.g. Devices.
17. Select “devices” from the list of dynamic content (I used search). Graphical user interface, application

    Description automatically generated
18. Now we must populate the DeviceDetails string.
19. Click on Add an action within the Devices loop.
20. Search for and select “Append to string variable”.
21. From the Name dropdown box select – DeviceDetails.
22. As with the Alert Details – complete the value box as per the screenshot. Graphical user interface, application

    Description automatically generated
23. To save time you can paste the following and add the variables against each item.

Device Name:

OS Platform:

Version:

Processor:

Build:

Health Status:

Risk Score:

Defender AV Status:

Onboarding Status:

---------------------------------------------

1. That completes the Devices loop – I would suggest you minimise the loop to prevent mistakes in the next step. Click on the top bar of the Devices apply to each box to minimise it:

Chart

Description automatically generated with medium confidence

1. Click on Add an action to create a new loop for the MITRE techniques.
2. Search for and select “Apply to each”. Optional – rename it to MITRE.
3. Within the Select an output.. box – select mitreTechnique. Make sure to select the correct one! Graphical user interface, application

   Description automatically generated
4. Within the MITRE loop – click on Add an action.
5. Search for select “append to string variable”.
6. From the Name dropdown select – MITRETechniques.
7. Within the Value box – select Current item under the MITRE loop. Graphical user interface, application

   Description automatically generated
8. Add a new line under Current Item – this ensures each output will be on a new line in the report.
9. Minimise the MITRE Loop by clicking on the top bar. A picture containing graphical user interface

   Description automatically generated
10. Click on Add an action.
11. One more loop – Search for and select “Apply to each”.
12. Within the output box – select entities from the list of dynamic content. Graphical user interface, text, application

    Description automatically generated
13. Each alert will contain entities of different types, so we need to treat each entity type differently. Click on Add an action within this new loop.
14. Search for and select the “Switch” control.
15. Within the On box select – entityType from the list of dynamic content un the Parse Incident JSON heading.
16. There are 8 different entity types, so therefore we must create 8 Case Values. We’ll complete one at a time.
17. Case 1 Equals (type the following into the box) – User
18. Within Case 1 – Click on Add an Action.
19. Search for and select “Append to String variable”.
20. From the drop down select – UserDetails.
21. Here are the row headings:

Username:

Domain:

User SID:

AAD SID:

UPN:

--------------------------------

1. Complete as per screenshot – remember to select the dynamic content from the Incident JSON!:

Graphical user interface, application

Description automatically generated

1. Create a new Case – click on the + to the right of the first Case box. Graphical user interface, application

   Description automatically generated
2. Case 2 – Process
3. Repeat step 43-44 for the ProcessDetails string.
4. Here are the row headings for you:

Process ID:

sha1:

sha256:

Filename:

File Path:

Process Commandline:

Process Creation Time:

Parent Process ID:

Parent Process Creation Time:

Account Name:

Domain Name:

Detection Status:

-----------------------------------------------------------

1. Complete as per screenshot: Graphical user interface, application

   Description automatically generated
2. New Case.
3. Case 3 – Registry
4. Here are the headings for the RegistryDetails string:

Registry Hive:

Registry Key:

Registry Value Type:

Registry Value:

-----------------------

1. Complete as per screenshot: Graphical user interface, application

   Description automatically generated
2. New Case.
3. Case 4 – Mailbox
4. Here are the headings for the MailboxDetails string:

UPN:

Display Name:

Address:

------------------------------

1. Complete as per screenshot: Graphical user interface, application, Teams

   Description automatically generated
2. New Case.
3. Case 5 – MailCluster
4. Complete as per screenshot: Graphical user interface, application

   Description automatically generated
5. New Case.
6. Case 6 – Ip
7. Complete as per screenshot: Graphical user interface, application

   Description automatically generated
8. New Case.
9. Case 7 – MailMessage
10. Compete as per screenshot: Graphical user interface, application

    Description automatically generated
11. New Case. Final one!!
12. Case 8 – File
13. Complete as per screenshot: Graphical user interface, application

    Description automatically generated
14. Minimise the Switch and Entities loop, as per screenshot: A picture containing graphical user interface

    Description automatically generated
15. That it, no more loops!

We now have everything we need to populate our Incident Report Template document.

## Populate the Incident Report

Before we begin this next section, let’s make sure everything looks correct. With every action minimised, our Flow should look as follows: A picture containing timeline

Description automatically generated

A picture containing graphical user interface

Description automatically generated

A picture containing graphical user interface

Description automatically generated

1. Click on +New Step at the bottom.
2. We now populate the Microsoft Word Template.
3. Search for and Select the Word Online action “Populate a Microsoft Word Template” – near the bottom of the list. Graphical user interface, text, application, email

   Description automatically generated
4. Activate the Location dropdown and locate the SharePoint site where hosted our template document. If you recall, we used – SharePoint Site – SOC Team.
5. The Document Library will be – Documents.
6. Click on the folder button to the right of the File box and locate the Defender Masterclass 3 – Incident Report Template document.
7. After you select the document the action box should expand to show all of our Content Controls that are contained in the document. Graphical user interface, application

   Description automatically generated
8. Feel free to manually update the Author and Company details.
9. Now we must populate each box with the appropriate value. Values are obtained from the Dynamic content.
   1. Alert ID = AlertID (under trigger heading).
   2. Filedetails = FileDetails (string variable).
   3. Alerts = AlertCount (Integer variable).
   4. Registrydetails = RegistryDetails (string variable).
   5. Status = status (from Parse Incident JSON – there are two, pick the top one).
   6. Alertdetails = AlertDetails (string variable).
   7. IncidentName = incidentName (from Parse Incident JSON).
   8. Severity = severity (from Parse Incident JSON – there are two, pick the top one).
   9. Title = incidentName (from Parse Incident JSON).
   10. Classification = classification (from Parse Incident JSON – pick first one).
   11. Comments = Leave blank <not implemented for this lab – would require a loop>.
   12. Ipdetails = IPDetails (string variable).
   13. Company = <Add company name>.
   14. Title = incidentName.
   15. CreatedTime = createdTime (from Parse Incident JSON – pick first one).
   16. IncidentName = incidentName.
   17. IncidentID = incidentID (pick top one).
   18. createdTime = createdTime (same as above).
   19. lastUpdateTime = lastUpdateTime (pick top one).
   20. assignedTo = assignedTo (pick top one).
   21. Determination = determination (pick top one).
   22. IncidentLink = https://security.microsoft.com/incidents/<incidentID>/overview
   23. deviceDetails = deviceDetails (string variable).
   24. Accountdetails = UserDetails (string variable).
   25. Processdetails = ProcessDetails (string variable).
   26. MailMessageDetails = MailMessageDetails (string variable).
   27. Mailboxdetails = MailboxDetails (string variable).
   28. Mailclusterdetails = MailClusterDetails (string variable).
   29. Mitre = MITRETechniques (string variable).
10. Well done – should now look as follows: Graphical user interface, application

    Description automatically generated with medium confidence Graphical user interface, application

    Description automatically generated
11. We now need to use this object to create a new Word Document and store it in our SharePoint library.
12. Click on +New Step.
13. Search for and select “Create file” (SharePoint).
14. The site address can be populated with the dropdown – select the SOC Team SharePoint site address from the list.
15. Click on the folder icon in the Folder Path box and click on Shared Documents in the list (don’t click on the right arrow).
16. Filename we will use the AlertID varialbe – select AlertID from dynamic content (under trigger heading) and end it with .docx (see screenshot below for example).
17. The File Content box is populated with our template object we completed in the previous action – select Microsoft Word document from the list of dynamic content.
18. Should now look as follows: Graphical user interface, text, application, email

    Description automatically generated
19. That’s it! All done.
20. Don’t forget to Save!

Optionally – we could go further and create an action to email the file to a set of recipients, but I’ll leave that up to you. In its current form the Flow will create a new Incident Report document each time a new alert lands in Defender for Endpoint.

Additional Option – you could create a filter or logic that ensure the Report is only produced for High Severity Alerts.

# Part 5 - The Test!

To test the Flow, we need to trigger an Alert in Defender for Endpoint.

If you completed the prerequisites, you would have an evaluation lab in Microsoft Defender - [Evaluation - Microsoft 365 security](https://security.microsoft.com/evaluation/overview)

Create a simulation to generate some Alerts and Incidents. If you’re not sure how, please refer to the Labs Getting Started document referenced at the start of this guide.

Of course, if you’re an Advanced User, please feel free to generate an Alert how you see fit.

Graphical user interface, text, application

Description automatically generated

After you create and run a Simulation, the Flow will trigger, and Incident Reports will begin landing in the SharePoint site.

Flow runs will appear in the Flow Summary Dashboard: Graphical user interface

Description automatically generated

Once a run has completed – one per Alert (a simulation will likely generate multiple Alerts, and thus multiple runs of the Flow) they will appear under the 28-day run history. Graphical user interface, text, application

Description automatically generated

Also, navigate to the SOC Team SharePoint site to view the Incident Reports that have been generated. Graphical user interface, text, application, email

Description automatically generated

We can now open these to view the results of our hard work. I prefer to open them in the local Word client, as the Word Online version tends to mess with the formatting and doesn’t look as good i.e. just download a report and open it locally.

Here’s an example of one of my reports:



Feel free to take these learnings and starting customising your own reports – or even expand this solution to be multitenant!

# Lab Complete

***Congratulations! You have now successfully completed this lab. We hope that you found this lab, and the associated lab materials useful. We look forward to seeing what you build as a result of attending this lab!***

**Lab Complete.**