

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
Part 2 – Create a new App Registration with access to APIs	3
Summary	9
Part 3 – The Incident Report Template.....	10
Store the template in SharePoint.	12
Summary	14
Part 4 – Power Automate.....	15
Create Flow and Trigger	15
Create Variables	17
Obtain Auth Tokens	20
Get Alert and Incident Data	26
Populate our String Variables	39
Populate the Incident Report.....	51
Part 5 - The Test!.....	59
Lab Complete	61

Part 1 - Getting Started

Please ensure you have completed the lab prerequisites:

Located at the GitHub Repository - aka.ms/defendermasterclass-repo

<https://github.com/JamesGrahamMSFT/DefenderMasterclass1/blob/main/Defender%20Masterclasses%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](#)

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.

The screenshot displays the 'Your evaluation lab' dashboard. It features a navigation bar with 'Overview', 'Devices', 'User Actions', 'Simulations', and 'Report'. The main content area is divided into three columns: 'Device allocation' showing 0 active devices, 'Simulations overview' showing 3 simulations executed with a progress bar (1 Failed, 1 Running, 1 Completed), and 'Report overview' showing 11 Alerts in, 4 Incidents, 0 Actions taken in, 2 Investigations, and 5 Key findings. A right-hand sidebar titled 'Evaluation Highlights' shows 'Set up status: Completed', 'Configuration' (16 devices for 12 hours), and 'Simulation integration' (2/2 simulators enabled). Below this are 'Suggestions' for running simulations, reviewing incidents, hunting for threats, checking for emerging threats, and providing feedback.

Category	Value
Active devices	0
Simulations executed	3
Alerts in	11
Incidents	4
Actions taken in	0
Investigations	2
Key findings	5

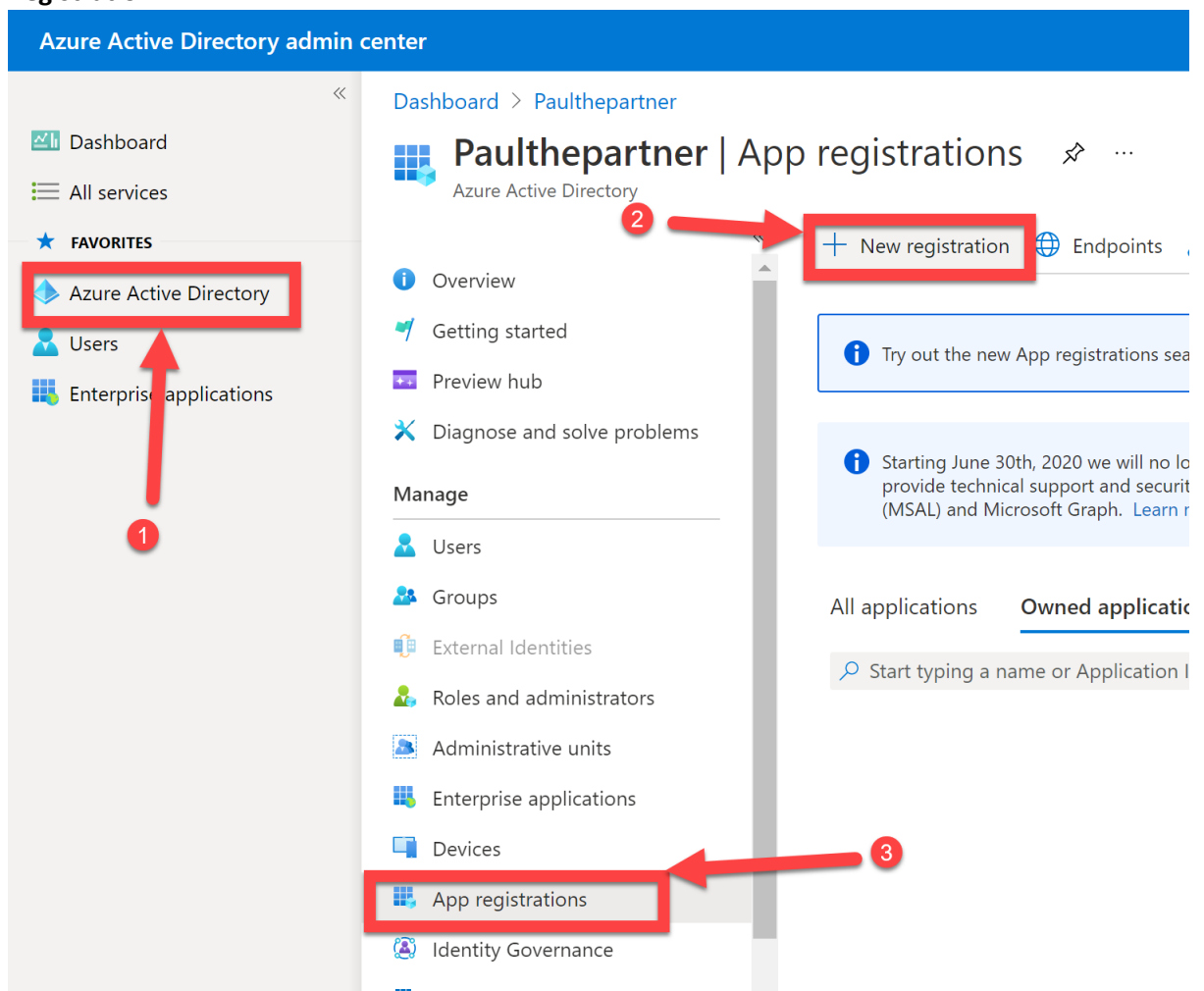
Part 2 – Create a new App Registration with access to APIs

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.

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



4. Enter a suitable **Name** for your App Registration, e.g. "MDE App".
5. 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 and create a multitenant app)
6. Leave the Redirect URI blank – not required for this lab.

Register an application ...

* Name

The user-facing display name for this application (this can be changed later).

MDE App ✓

Supported account types

Who can use this application or access this API?

- ☒ Accounts in this organizational directory only (Northwindtraders only - Single tenant)
- ☐ Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- ☐ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- ☐ Personal Microsoft accounts only

[Help me choose...](#)

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web ▼ e.g. https://example.com/auth

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from [Enterprise applications](#).

By proceeding, you agree to the [Microsoft Platform Policies](#) ↗

Register

7. Click **Register**.
8. 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:
Application (client) ID
b8efd4ec-██████████-49175c3e9b2b
9. Also take a copy of the Directory (tenant) ID and store in notepad.
10. 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.
11. Click **Certificates & secrets**, select **New client secret**, click **Add**.

Defender for Endpoint Management App | Certificates & secrets

Search (Ctrl+/) < Got feedback?

Overview
Quickstart
Integration assistant
Manage
Branding
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles | Preview
Owners
Roles and administrators | Preview
Manifest
Support + Troubleshooting
Troubleshooting

Add a client secret

Description

Expires

☒ In 1 year
☐ In 2 years
☐ Never

Add Cancel

Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

+ New client secret

Description	Expires	Value	ID
No client secrets have been created for this application.			

12. The Client secret will then be generated and will look 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

Password uploaded on Thu Feb 25 2021 2/25/2022 1ql [redacted] vx fffc01e-9545-4b1c-bba2-ef54df07a716

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

Defender for Endpoint Management App | API permissions

Search (Ctrl+/) << Refresh Got feedback?

Overview
Quickstart
Integration assistant

Manage

- Branding
- Authentication
- Certificates & secrets
- Token configuration
- API permissions**
- Expose an API
- App roles | Preview
- Owners
- Roles and administrators | Preview
- Manifest

Support + Troubleshooting

Starting November 9th, 2020 end users will no longer be able to grant consent to newly registered multi-tenant applications. [Learn more](#)

The "Admin consent required" column shows the default value for an organization. However, user consent may not reflect the value in your organization, or in organizations where this app will be used. [Learn more](#)

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of their configuration. To include all the permissions that the application needs, click the "Add a permission" button. [Learn more about permissions and consent](#)

+ Add a permission ✓ Grant admin consent for Paulthepartner

API / Permissions name	Type	Description
Microsoft Graph (1)		
User.Read	Delegated	Sign in and read user profile

To view and manage permissions and user consent, try [Enterprise applications](#).

14. 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).

Request API permissions

Select an API

Microsoft APIs **APIs my organization uses** My APIs

Apps in your directory that expose APIs are shown below

WindowsDefenderATP

Name	Application (client) ID
WindowsDefenderATP	fc780465-2017-40d4-a0c5-307022471b92

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

Delegated permissions

Your application needs to access the API as the signed-in user.

Application permissions

Your application runs as a background service or daemon without a signed-in user.

1 →

Select permissions [expand all](#)

Permission	Admin consent required
> AdvancedQuery	
<input checked="" type="checkbox"/> Alert (1)	
<input checked="" type="checkbox"/> Alert.Read.All ⓘ Read all alerts	Yes

2 → 3 →

16. Scroll down, expand **Score**, select **Score.Read.All**, click **Add permissions**

<input checked="" type="checkbox"/> Score (1)	
<input checked="" type="checkbox"/> Score.Read.All ⓘ Read Threat and Vulnerability Management score	Yes
> SecurityConfiguration	
> SecurityRecommendation	

Add permissions

Discard

3 →

17. Repeat the process to add the following permissions for WindowsDefenderATP:

a. Alert.ReadWrite.All

18. Also, we require the following permissions added for Microsoft Threat Protection:

b. Incident.Read.All

c. Incident.ReadWrite.All

[List incidents API in Microsoft 365 Defender | Microsoft Docs](#)

19. Once Added – click on “Grant Admin Consent for <tenant name>”.

20. If successful, all should look as follows:

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

[+ Add a permission](#) [✓ Grant admin consent for Contoso](#)

API / Permissions name	Type	Description	Admin consent requ...	Status
▼ Microsoft Graph (1)				
User.Read	Delegated	Sign in and read user profile	No	✓ Granted for Contoso
▼ Microsoft Threat Protection (2)				
Incident.Read.All	Application	Read all incidents	Yes	✓ Granted for Contoso
Incident.ReadWrite.All	Application	Read and write all incidents	Yes	✓ Granted for Contoso
▼ WindowsDefenderATP (2)				
Alert.Read.All	Application	Read all alerts	Yes	✓ Granted for Contoso
Alert.ReadWrite.All	Application	Read and write all alerts	Yes	✓ Granted for Contoso

To view and manage permissions and user consent, try [Enterprise applications](#).

21. 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.
22. 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:

Home >

Northwindtraders | Overview ...

Azure Active Directory

Switch tenant Delete tenant + Create

Northwindtraders

Search your tenant

Tenant information

Your role
Global administrator and 1 other roles
[More info](#)

License
Azure AD Premium P2

Tenant ID
602defa2-b842-4355-bd12-49e3... [Copy](#)

Primary domain
northwindtraders.defendermasterclass.cloud

23. 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:
 - a. Windows Defender ATP
 - i. Alert.ReadWrite.All
 - ii. Alert.Read.All
 - b. Microsoft Threat Protection
 - i. Incident.Read.All
 - ii. Incident.ReadWrite.All
 - c. Microsoft Graph (default, ok to leave this):
 - i. 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.

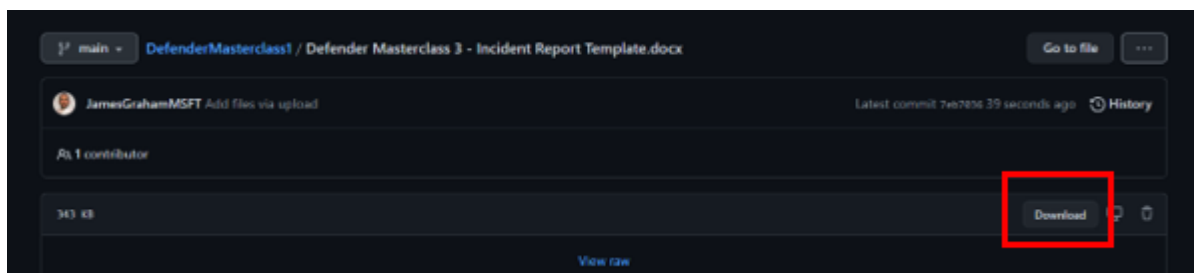
Part 3 – The Incident Report Template

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

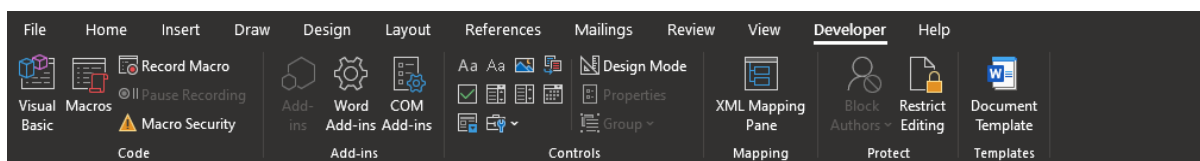
The template is located on the GitHub repo:

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

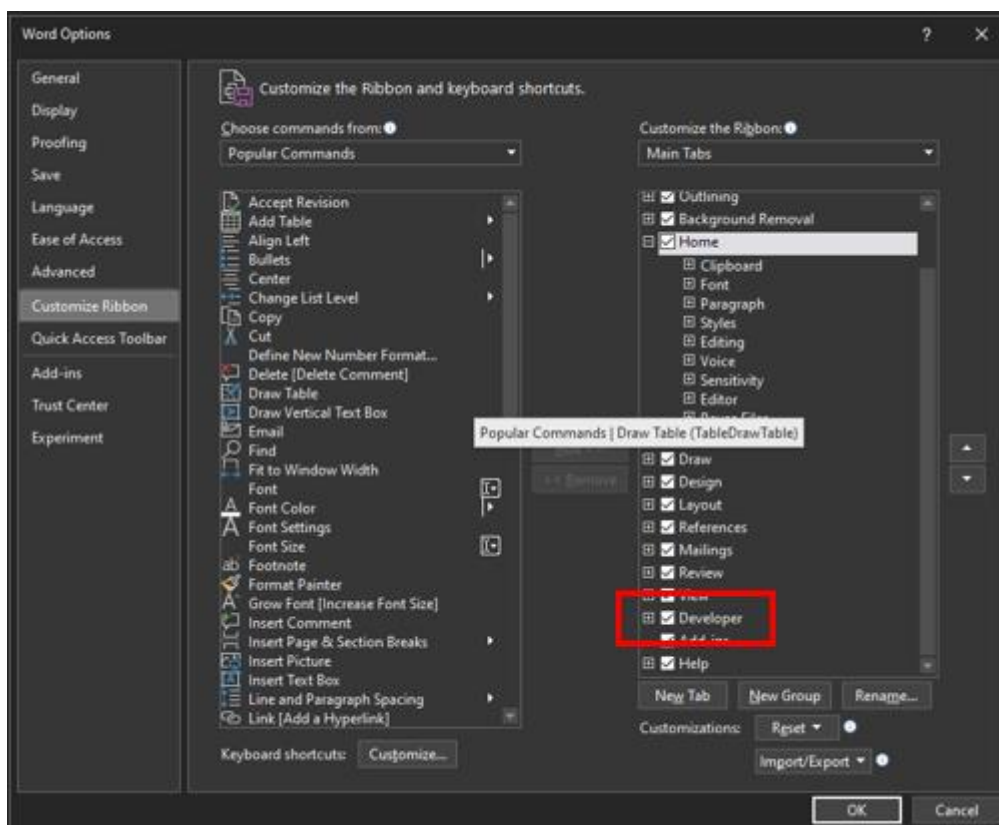
Please download the file.



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



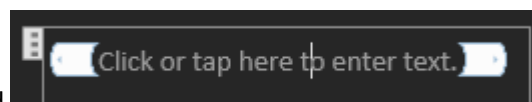
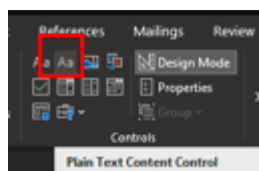
1. Enable Developer mode in Word – Open Word, navigate to Options and enable the Developer tab:



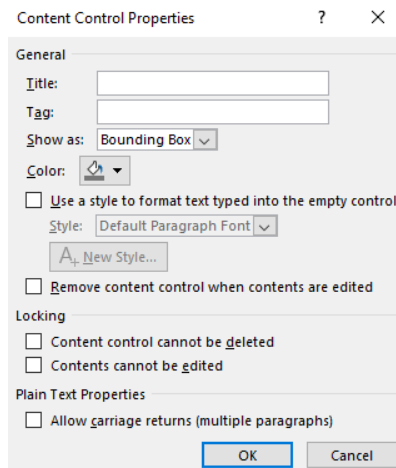
2. Open the Defender Masterclass 3 – Incident Report Template document.
3. Navigate to the Developer tab – enable Design Mode.
4. Once enabled you will be able to see all the fields that I have created.

Field Name	Value
Incident ID	incidentID Click or tap here to enter text. incidentID
Incident Name	incidentName Click or tap here to enter text. incidentName
Created Time	createdTime Click or tap here to enter text. createdTime
Last Update Time	lastUpdateTime Click or tap here to enter text. lastUpdateTime
Assigned To	assignedTo Click or tap here to enter text. assignedTo
Classification	classification Click or tap here to enter text. classification
Determination	determination Click or tap here to enter text. determination
Status	status Click or tap here to enter text. status
Severity	severity Click or tap here to enter text. severity
Comments	comments Click or tap here to enter text. comments
No. of associated Alerts	alerts Click or tap here to enter text. alerts
Link to Incident	incidentLink Click or tap here to enter text. incidentLink

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



- c. This will generate the field
- d. Optional – insert description text within the field.
- e. Whilst the cursor is active inside the field – click on Properties in the Developer Tab.



The 'Content Control Properties' dialog box is shown with the following settings:

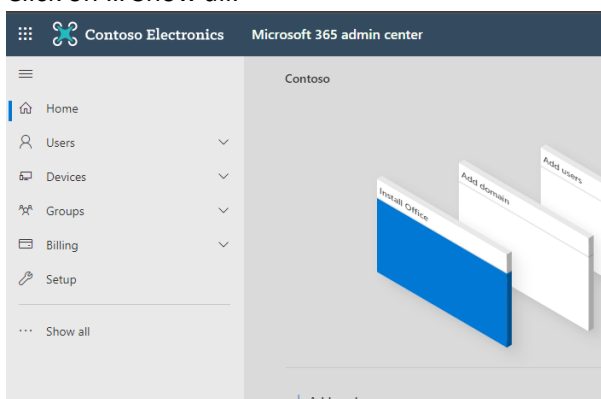
- General:**
 - Title: (empty text box)
 - Tag: (empty text box)
 - Show as: Bounding Box (dropdown)
 - Color: (color picker icon)
 - ☐ Use a style to format text typed into the empty control
 - Style: Default Paragraph Font (dropdown)
 - New Style... (button)
 - ☐ Remove content control when contents are edited
- Locking:**
 - ☐ Content control cannot be deleted
 - ☐ Contents cannot be edited
- Plain Text Properties:**
 - ☐ Allow carriage returns (multiple paragraphs)

Buttons: OK, Cancel

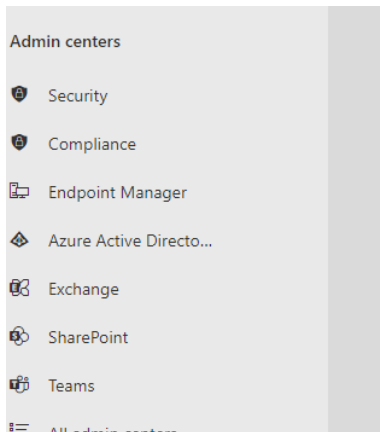
- f. From here we can give the field an appropriate name.
- g. Important - If the output will be spread over multiple lines, we must tick the box at the bottom – Allow carriage returns.

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://myadmin.microsoft.com)
4. Click on ... Show all.



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



6. Within SharePoint Admin Center – navigate to Sites -> Active Sites.
7. We will use the SOC Team site to host our template.

8. Click on SOC Team in the list of Active Sites – on the left tab that opens, click on the URL.

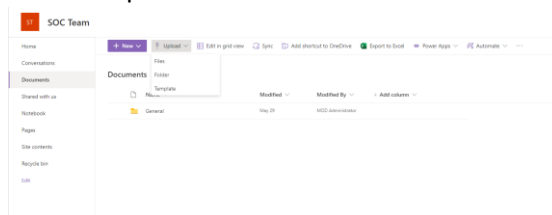
SOC Team

General Activity Permissions Policies

Site name	URL
SOC Team	.../sites/SOC Team
Edit	Edit
Hub association	Template
None	Team site
Edit	
Microsoft 365 Group connected	Domain
Yes	msdx734648.sharepoint.com
Description	
None	

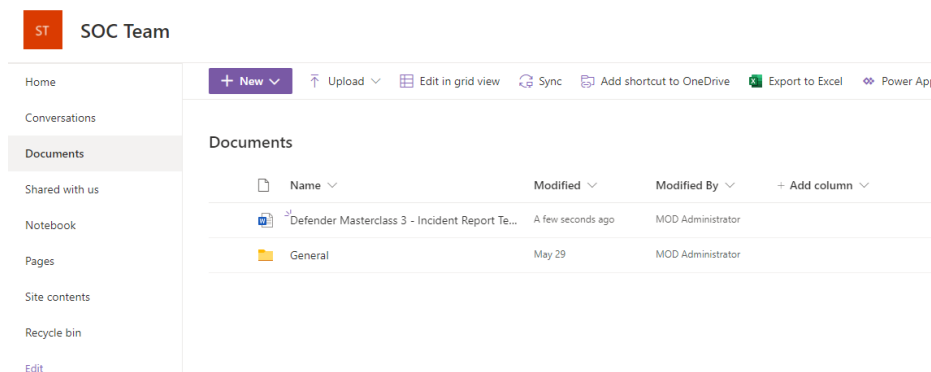
9. Within the SharePoint site – Click on Documents.

10. Click on Upload – Files.



11. Select the Defender Masterclass 3 – Incident Report Template you downloaded earlier.

12. Once completed you should see the template file listed in the Documents:



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.

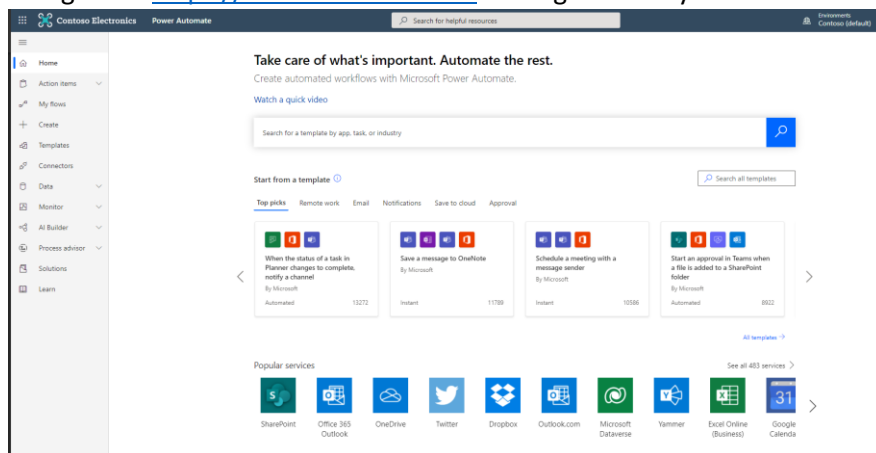
Part 4 – Power Automate

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

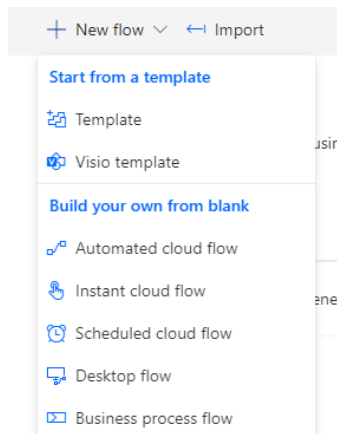
- An M365 Demo Tenant with Microsoft Defender for Endpoint.
- The ability to generate an Alert/Incident within Microsoft Defender for Endpoint.
- An App Registration with access to required APIs.
- A report template stored within a SharePoint site.

Create Flow and Trigger

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



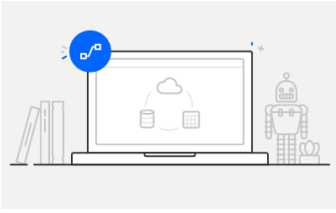
2. Click on My Flows.
3. Click on + New Flow -> Automated cloud flow.



4. Provide a name, e.g. Incident Report Flow.
5. Within the search bar for the flow's trigger, start typing Defender.

6. Click on the Trigger when a new WDATP alert occurs.

Build an automated cloud flow



Free yourself from repetitive work just by connecting the apps you already use—automate alerts, reports, and other tasks.

Examples:

- Automatically collect and store data in business solutions
- Generate reports via custom queries on your SQL database

Flow name

Incident Report Flow

Choose your flow's trigger

defe

Triggers when a new remediation activ...
Microsoft Defender ATP

Triggers - Trigger when new WDATP aL...
Microsoft Defender ATP

Skip Create Cancel

7. Click on Create.

8. Click on the Sign In box within the Flow Trigger – sign in with Administrator credentials.

Microsoft Defender ATP

Sign in to create a connection to Microsoft Defender ATP.

Sign in

Connect with Service Principal

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

Read machine information

Read and write machine information

Isolate machine

Collect forensics

Scan machine

Restrict code execution

Stop and quarantine file

Offboard machine

Read file profiles

Read URL profiles

Read IP address profiles

Read user profiles

Read and write alerts

Read alerts

Sign in and read user profile

☒ Consent on behalf of your organisation

If you accept, this app will get access to the specified resources for all users in your organisation. No one else will be prompted to review these permissions.

Accepting these permissions means that you allow this app to use your data as specified in their Terms of Service and Privacy Statement. The publisher has not provided links to their Terms for you to review. You can change these permissions at <https://myapps.microsoft.com>. Show details

Does this app look suspicious? [Report it here](#)

Cancel Accept

10. 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.

The screenshot shows the 'Initialize variable' configuration window. The title bar is purple with a variable icon {x} and the text 'Initialize variable'. Below the title bar, there are three fields: '* Name' with the value 'AlertDetails', '* Type' with a dropdown menu showing 'String', and 'Value' with a text box containing 'Enter initial value'. There are also help and more options icons in the top right corner.

6. Repeat steps 1-5 for DeviceDetails.

The screenshot shows the 'Initialize variable 2' configuration window. The title bar is purple with a variable icon {x} and the text 'Initialize variable 2'. Below the title bar, there are three fields: '* Name' with the value 'DeviceDetails', '* Type' with a dropdown menu showing 'String', and 'Value' with a text box containing 'Enter initial value'. There are also help and more options icons in the top right corner.

7. Repeat steps 1-5 for UserDetails.

The screenshot shows the 'Initialize variable 3' configuration window. The title bar is purple with a variable icon {x} and the text 'Initialize variable 3'. Below the title bar, there are three fields: '* Name' with the value 'UserDetails', '* Type' with a dropdown menu showing 'String', and 'Value' with a text box containing 'Enter initial value'. There are also help and more options icons in the top right corner.

8. Repeat steps 1-5 for ProcessDetails.

The screenshot shows the 'Initialize variable 4' configuration window. The title bar is purple with a variable icon {x} and the text 'Initialize variable 4'. Below the title bar, there are three fields: '* Name' with the value 'ProcessDetails', '* Type' with a dropdown menu showing 'String', and 'Value' with a text box containing 'Enter initial value'. There are also help and more options icons in the top right corner.

9. Repeat steps 1-5 for RegistryDetails.

{x} Initialize variable 5

* Name

RegistryDetails

* Type

String

Value

Enter initial value

10. Repeat steps 1-5 for FileDetails.

{x} Initialize variable 6

* Name

FileDetails

* Type

String

Value

Enter initial value

11. Repeat steps 1-5 for IPDetails.

{x} Initialize variable 7

* Name

IPDetails

* Type

String

Value

Enter initial value

12. Repeat steps 1-5 for MailMessageDetails.

{x} Initialize variable 8

* Name

MailMessageDetails

* Type

String

Value

Enter initial value

13. Repeat steps 1-5 for MailboxDetails.

{x} Initialize variable 9

* Name

MailboxDetails

* Type

String

Value

Enter initial value

14. Repeat steps 1-5 for MailClusterDetails.

{x} Initialize variable 10

* Name

* Type

String

Value

15. Repeat steps 1-5 for MITRETechniques.

{x} Initialize variable 11

* Name

* Type

String

Value

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.

16. + New Step.
17. Initialize Variable.
18. Name – AlertCount.
19. Type – Integer.
20. Initial Value – 0.

{x} Initialize variable 12

* Name

* Type

Integer

Value

Add dynamic content

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](#)

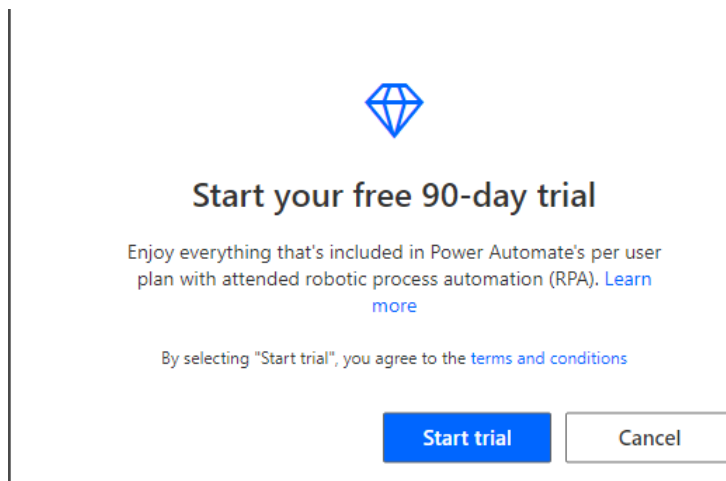
[Create an app to access Microsoft Defender for Endpoint without a user | Microsoft Docs](#)

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.



4. Complete the HTTP Action as follows:
 - a. Method – POST
 - b. URI - `https://login.microsoftonline.com/<insert tenant ID here>/oauth2/v2.0/token`
 - c. Headers (left box) - Content-Type
 - d. Headers (right box) - application/x-www-form-urlencoded
 - e. 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):

The screenshot shows an HTTP client interface with the title "Get Auth token for M365 Defender". It includes a globe icon, a help icon, and a menu icon. The interface is divided into several sections:

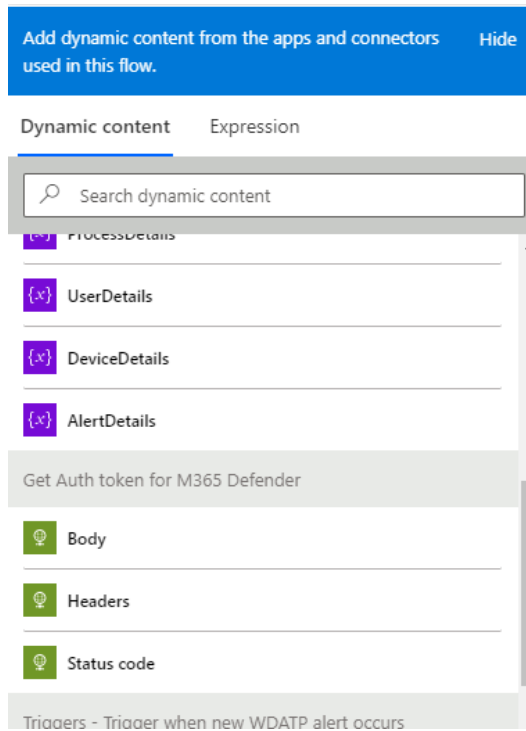
- * Method:** A dropdown menu set to "POST".
- * URI:** A text box containing the URL: `https://login.microsoftonline.com/172b27c6-3be5-4118-aecb-8ff8a30dd655/oauth2/v2.0/token`.
- Headers:** A table with two columns: "Content-Type" and "application/x-www-form-urlencoded". Below it is a row with "Enter key" and "Enter value".
- Queries:** A table with two columns: "Enter key" and "Enter value".
- Body:** A text box containing the following parameters: `client_id=de9f903c-fe91-4d9d-bdc7-b29722bad9fb`, `&scope=https://api.security.microsoft.com/.default`, `&client_secret=43jQ-[REDACTED]Dt233-`, and `&grant_type=client_credentials`.
- Cookie:** A text box with the placeholder "Enter HTTP cookie".

At the bottom, there is a link "Show advanced options" with a downward arrow.

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.

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



9. 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"
    }
  }
}
```

10. Remove the extra line that may get added when you paste the above into the Schema box.

11. It should look as follows:



12. Click on Save at the top right to ensure there are no errors in the Schema – when you click save, error checking takes place.

13. 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.



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.

14. Click on +New Step.

15. Search for https and select the HTTP action.

16. 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:

- Method – POST
- URI - <https://login.microsoftonline.com/<insert tenant ID here>/oauth2/v2.0/token>
- Headers (left box) - Content-Type
- Headers (right box) - application/x-www-form-urlencoded
- Body –
client_id=<insert App ID here>
&scope= https://securitycenter.onmicrosoft.com/windowsatp/service/.default
&client_secret=<insert secret value here>
&grant_type=client_credentials

Should look as follows:

Get Auth token for MDE

* Method: POST

* URI: https://login.microsoftonline.com/172b27c6-3be5-4118-aecb-8ff8a30dd655/oauth2/v2.0/token

Headers:

Content-Type	application/x-www-form-urlencoded
Enter key	Enter value

Queries:

Enter key	Enter value
-----------	-------------

Body:

```
client_id=de9f903c-fe91-4d9d-bdc7-b29722bad9fb
&scope=https://securitycenter.onmicrosoft.com/windowsatpservice/.default
&client_secret=43jQ[redacted]3-
&grant_type=client_credentials
```

Cookie: Enter HTTP cookie

Show advanced options

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

MDEAuth

* Content: Body

* Schema:

```
{
  "type": "integer",
  "ext_expires_in": {
    "type": "integer"
  },
  "access_token": {
    "type": "string"
  }
}
```

Generate from sample

20. 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](#)

[Get incident API | Microsoft Docs](#)

1. Click on +New Step
2. Search for http and select the HTTP action.
3. Complete the HTTP Action as follows:
 - a. Method – GET
 - b. 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.

The screenshot shows the configuration of an HTTP action in a workflow editor. The left pane displays the HTTP action settings: Method is GET, URI is `https://api.securitycenter.windows.com/api/alerts/<AlertID>`, and there are fields for Headers, Queries, Body, and Cookie. The right pane shows the 'Add dynamic content from the apps and connectors' dialog, which is open to the 'Dynamic content' tab. It lists various dynamic content items, including 'ext_expires_in', 'Get Auth token for M365 Defender', 'Body', 'Headers', 'Status code', 'Triggers - Trigger when new WDATP alert occurs', 'Alert Id', and 'Machine Id'. The 'Alert Id' item is selected.

- c. Headers (left box) – authorization
- d. 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.

4. 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.

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

5. Click on +New Step.
6. Search for Parse JSON and select it from the available actions.
7. Select the Content box – the Dynamic Content dialog box will appear.
8. You want to select Body from your Get Alert Data HTTP action.
9. 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": {}
    }
  }
}

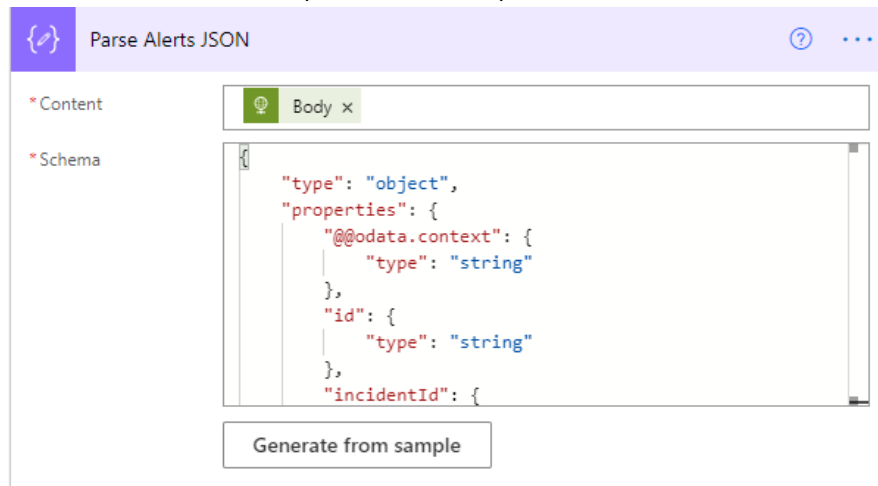
```

```

}
}
}
}
}
}

```

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

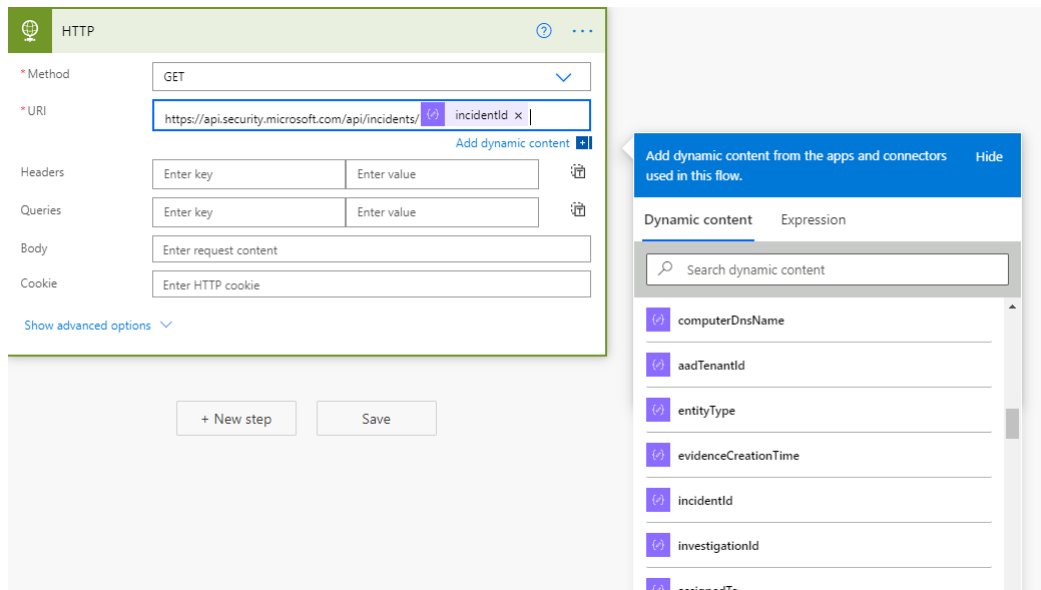


7. Click on Save at the top right to ensure there are no errors in the Schema.

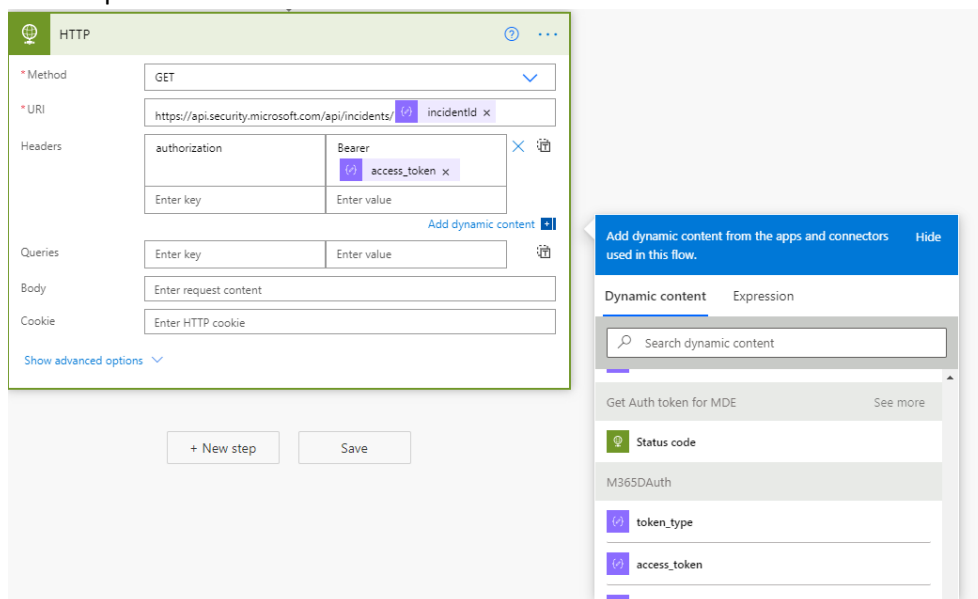
Within the JSON that gets returned will be the Incident ID – we now want to use this to obtain the Incident data.

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

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



- c. Headers (left box) – authorization
- d. 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.



11. 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.

Get Incident Data

* Method: GET

* URI: https://api.security.microsoft.com/api/incidents/{incidentId}

Headers:

authorization	Bearer {access_token}
Enter key	Enter value

Queries:

Enter key	Enter value
-----------	-------------

Body: Enter request content

Cookie: Enter HTTP cookie

[Show advanced options](#)

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.

12. Click on +New Step.
13. Search for Parse JSON and select it from the available actions.
14. Select the Content box – the Dynamic Content dialog box will appear.
15. You want to select Body from your Get Incident Data HTTP action.
16. 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": {},

```

```
"userId": {},  
    "detectionStatus": {},  
    "deviceId": {},  
    "registryHive": {},  
    "registryKey": {},  
    "registryValueType": {},  
    "ipAddress": {},  
    "url": {},  
    "aadUserId": {},  
    "userPrincipalName": {},  
    "mailboxDisplayName": {},  
    "mailboxAddress": {},  
    "clusterBy": {},  
    "sender": {},  
    "recipient": {},  
    "subject": {},  
    "deliveryAction": {},  
    "securityGroupId": {},  
    "securityGroupName": {},  
    "registryValue": {}  
}  
  
}  
  
}  
  
}  
  
}  
  
}  
  
}
```

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



19. 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

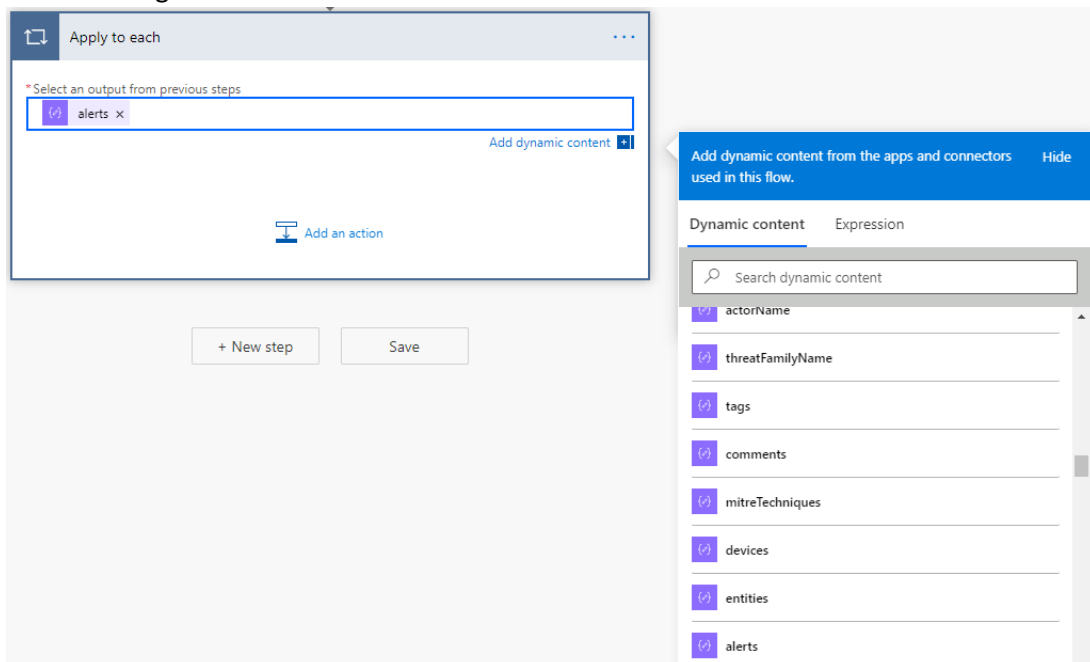
Looking at the Incident API documentation and/or the Schema above we can see the format of the Incident information that will be returned in the JSON.

[List incidents API in Microsoft 365 Defender | Microsoft Docs](#)

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.

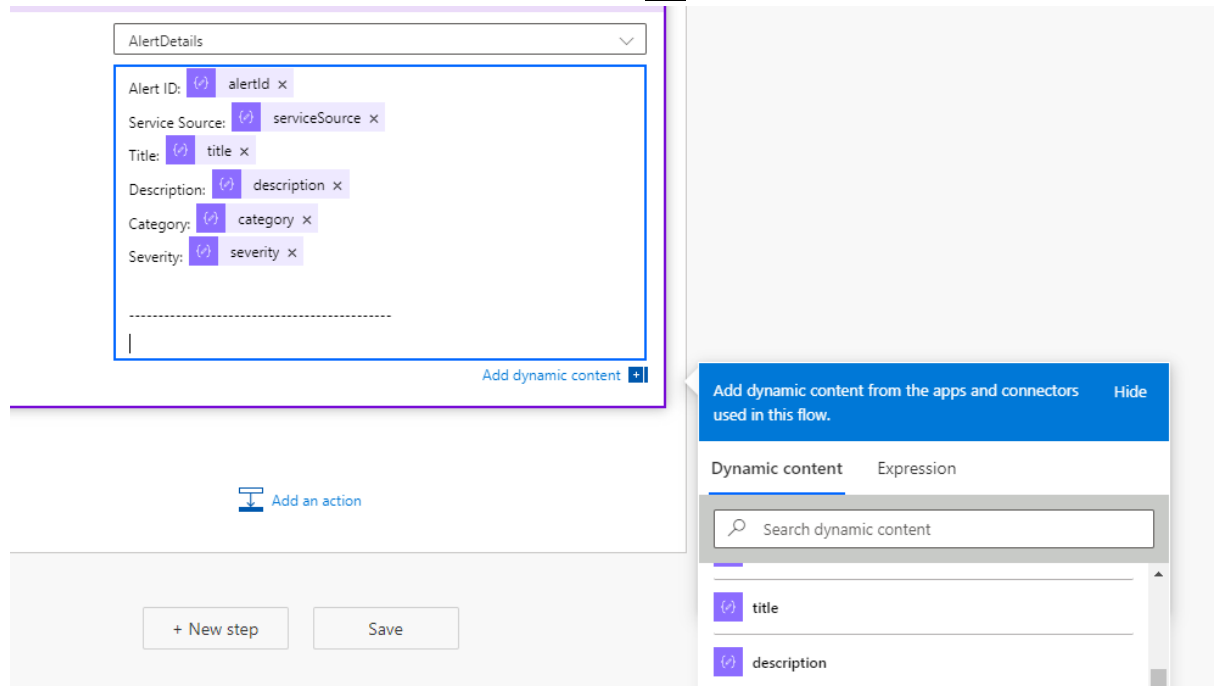
In the next steps we must loop through the Alerts associated with the Incident (and the sub-arrays contained within) and populate the strings we initialised at the start of the Flow.

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.

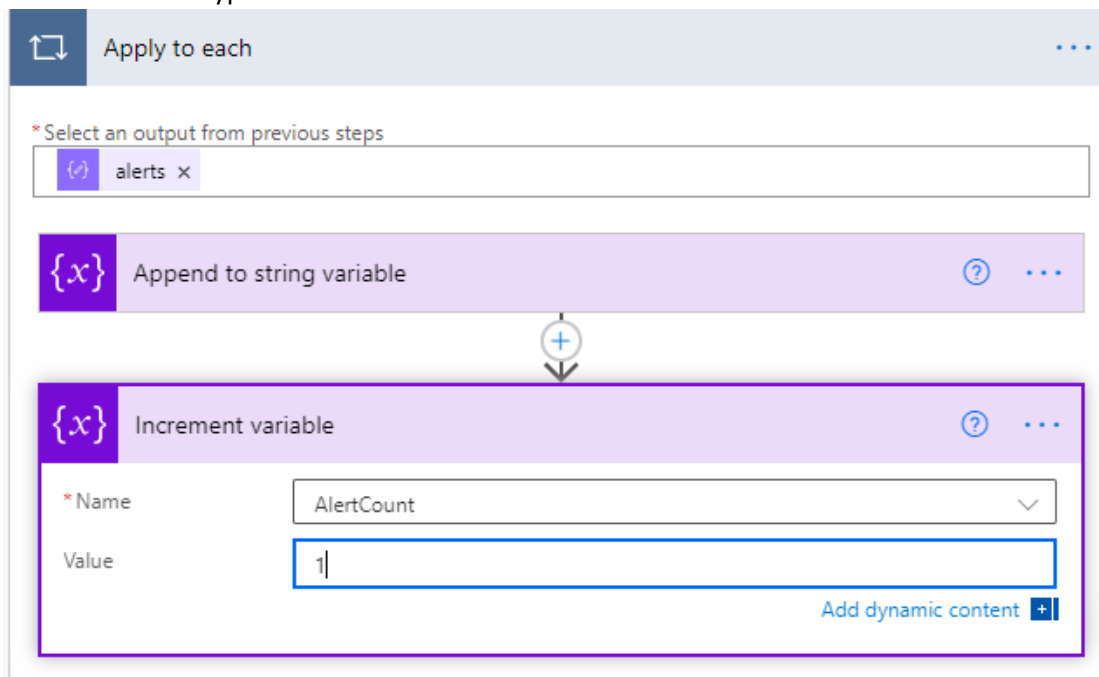


4. We now populate our first variable with Alert Information.
5. Click on Add and action within the Apply to each box.
6. Search for and select “Append to string variable”.
7. In the Name drop down select – AlertDetails.
8. Within the value box – complete as per the screenshot. Ensure you add a new line 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.

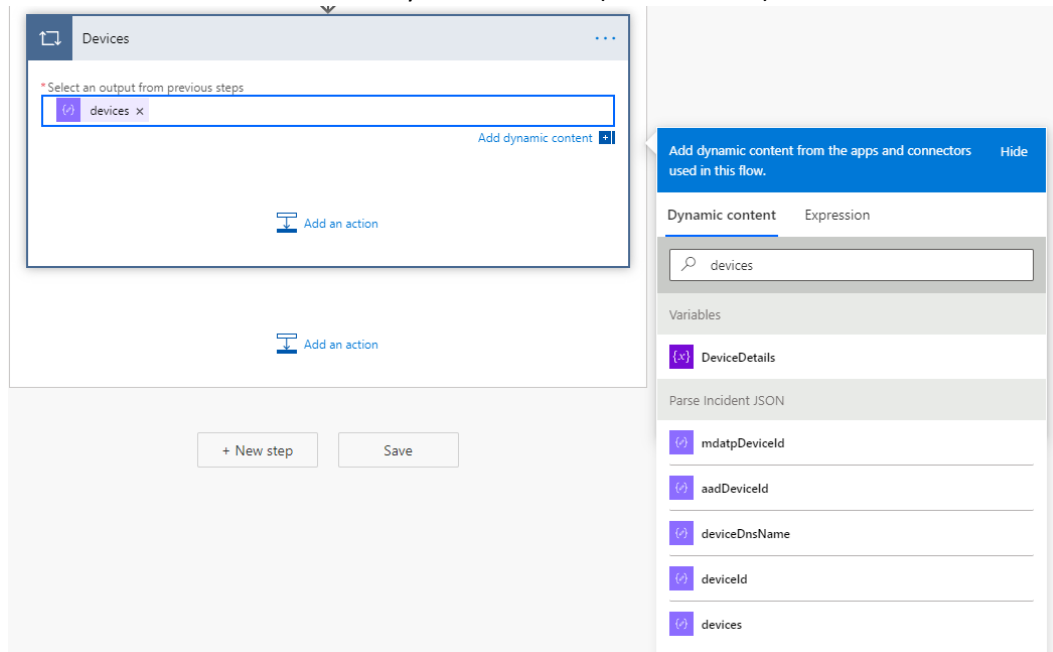


9. Next step is increment the AlertCount variable – this will give us a count of the associated Alerts for this Incident.
10. Click on Add an Action under the previous action.
11. Search for and select “Increment variable”.
12. In the name dropdown select – AlertCount.
13. In the Value box type – 1.



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).



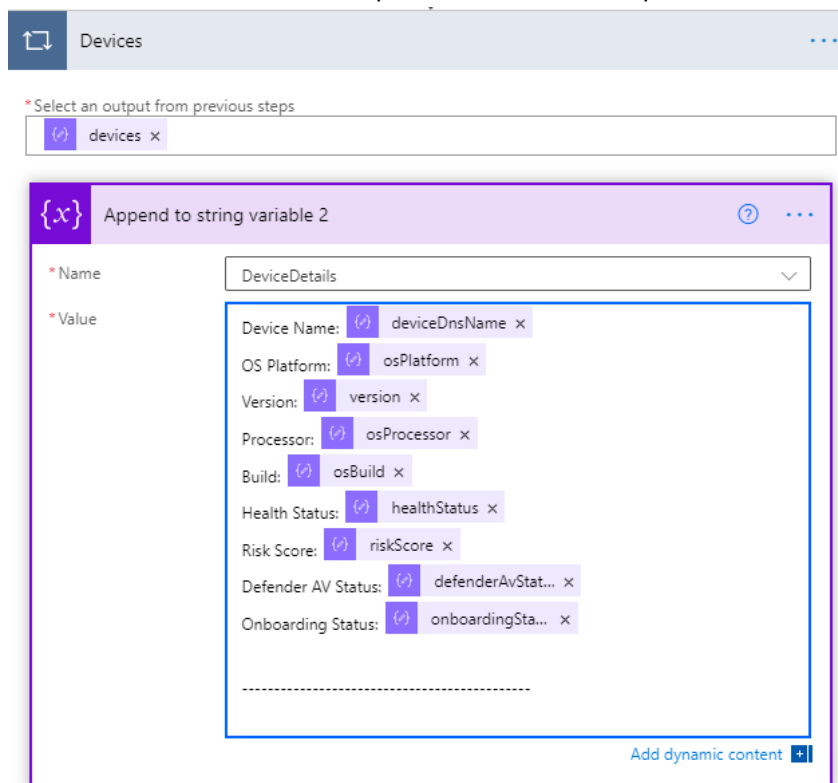
18. Now we must populate the DeviceDetails string.

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.



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

Device Name:

OS Platform:

Version:

Processor:

Build:

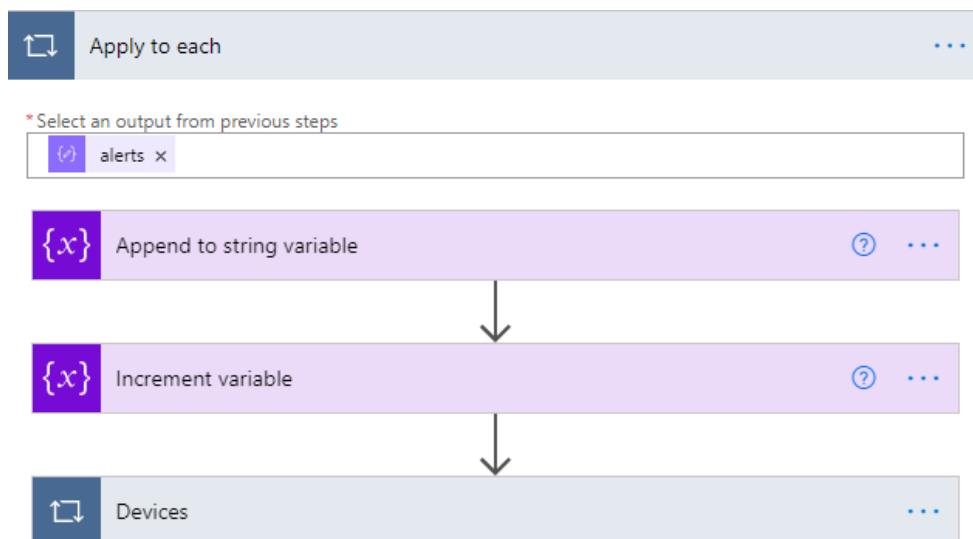
Health Status:


Risk Score:

Defender AV Status:

Onboarding Status:

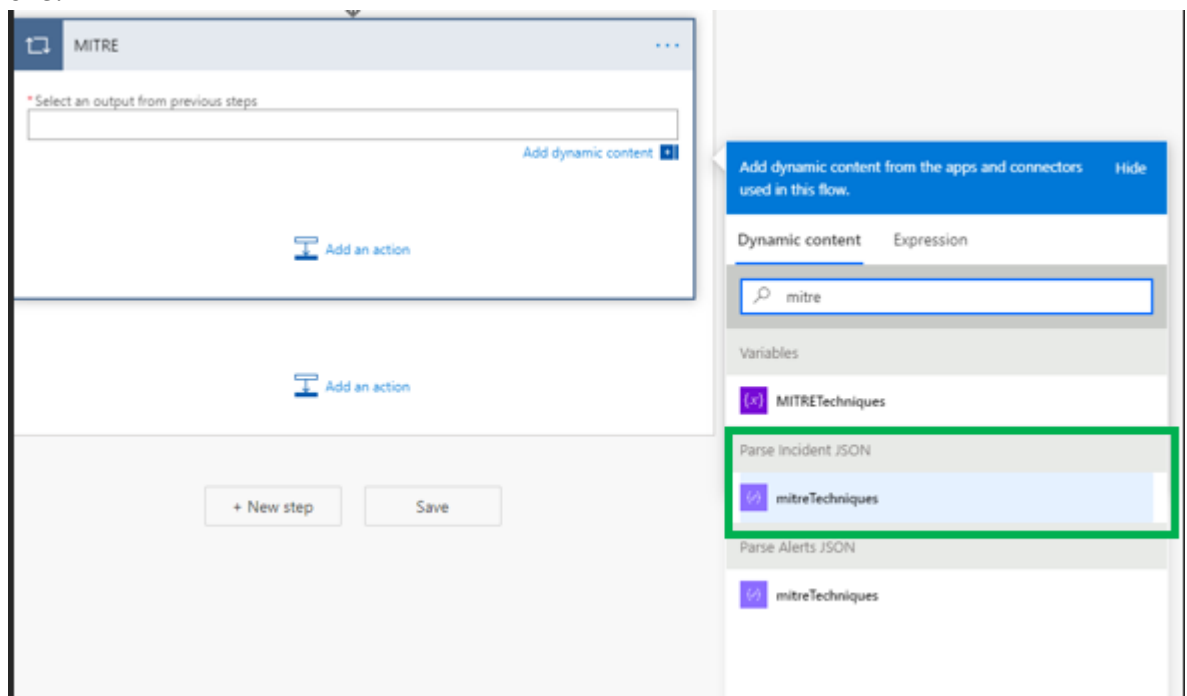
24. That completes the Devices loop – I would suggest you minimise the loop to prevent mistakes in the next step. Click on the top bar of the Devices apply to each box to minimise it:



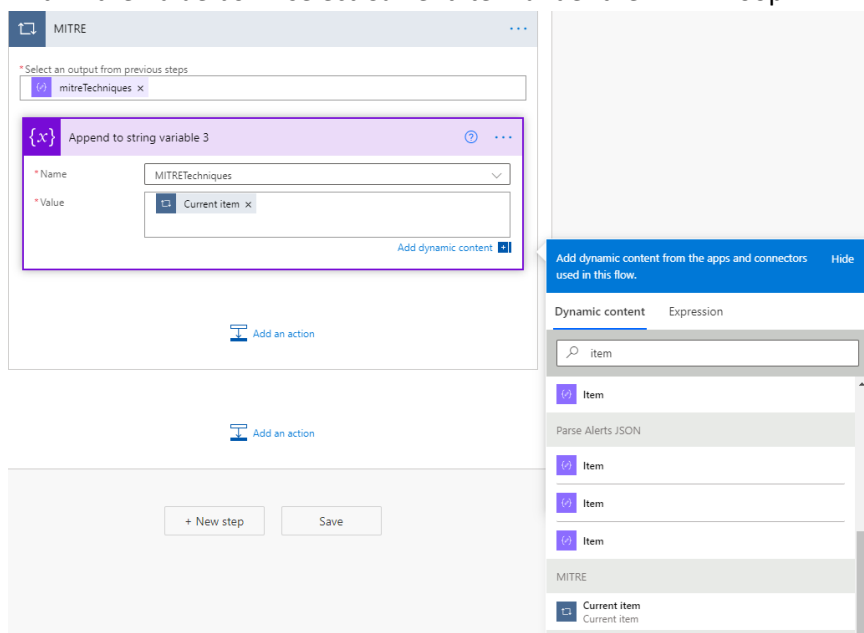
 Add an action

25. Click on Add an action to create a new loop for the MITRE techniques.
26. Search for and select “Apply to each”. Optional – rename it to MITRE.

27. Within the Select an output.. box – select mitreTechnique. Make sure to select the correct one!

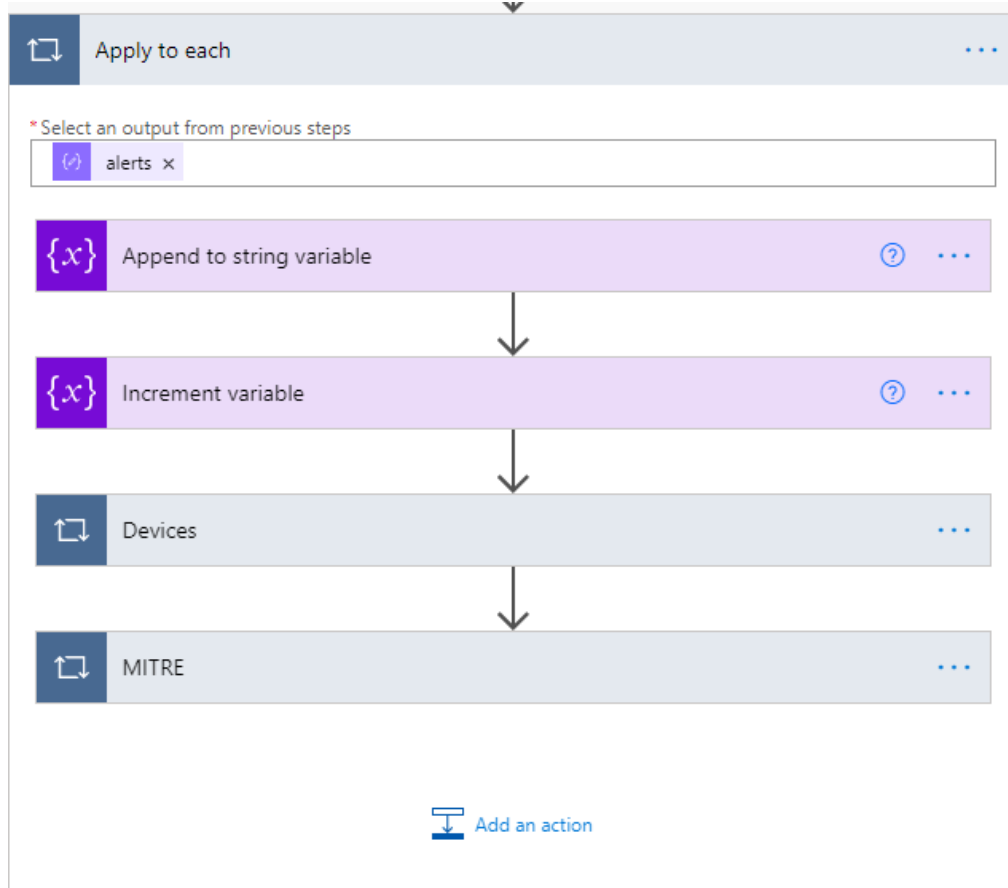


28. Within the MITRE loop – click on Add an action.
29. Search for select “append to string variable”.
30. From the Name dropdown select – MITRETechniques.
31. Within the Value box – select Current item under the MITRE loop.



32. Add a new line under Current Item – this ensures each output will be on a new line in the report.

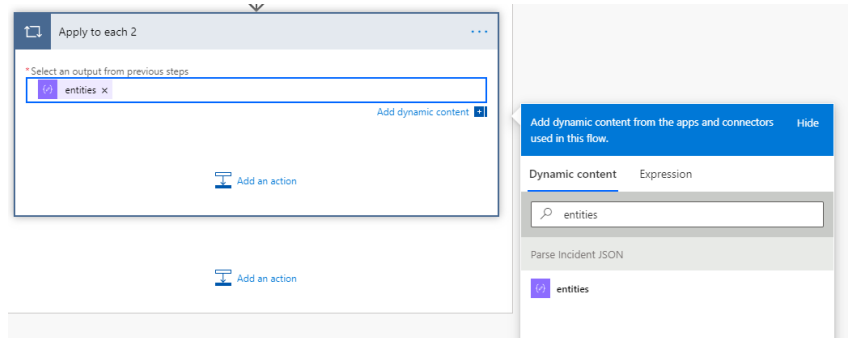
33. Minimise the MITRE Loop by clicking on the top bar.



34. Click on Add an action.

35. One more loop – Search for and select “Apply to each”.

36. Within the output box – select entities from the list of dynamic content.



37. 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.

38. Search for and select the “Switch” control.

39. Within the On box select – entityType from the list of dynamic content un the Parse Incident JSON heading.

40. There are 8 different entity types, so therefore we must create 8 Case Values. We’ll complete one at a time.

41. Case 1 Equals (type the following into the box) – User

42. Within Case 1 – Click on Add an Action.

43. Search for and select “Append to String variable”.

44. From the drop down select – UserDetails.

45. Here are the row headings:

Username:

Domain:

User SID:

AAD SID:

UPN:

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

Case

* Equals

User

{x} Append to string variable 4

* Name: UserDetails

* Value:

```
{
  "Username": "accountName",
  "Domain": "domainName",
  "User SID": "userSid",
  "AAD SID": "aadUserId",
  "UPN": "userPrincipalName"
}
```

Add dynamic content

47. Create a new Case – click on the + to the right of the first Case box.

48. Case 2 – Process

49. Repeat step 43-44 for the ProcessDetails string.

50. 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:



51. Complete as per screenshot:

The screenshot shows a Power Automate flow editor for 'Case 2'. A variable named 'Process' is selected, and the action 'Append to string variable 5' is configured. The 'Name' field is set to 'ProcessDetails'. The 'Value' field contains a list of process details, each preceded by a label and a variable icon: Process ID: processId, sha1: sha1, sha256: sha256, Filename: fileName, File Path: filePath, Process Commandline: processComm..., Process Creation Time: processCreatio..., Parent Process ID: parentProcessId, Parent Process Creation Time: parentProcess..., Account Name: accountName, Domain Name: domainName, and Detection Status: detectionStatus. A dashed line is at the bottom of the list. An 'Add dynamic content' button is at the bottom right.

52. New Case.

53. Case 3 – Registry

54. Here are the headings for the RegistryDetails string:

Registry Hive:

Registry Key:

Registry Value Type:

Registry Value:

55. Complete as per screenshot:

The screenshot shows a Power Automate flow editor for a case named 'Case 3'. At the top, there is a text box labeled 'Registry' with a red asterisk and the word 'Equals' above it. Below this is an action card titled 'Append to string variable' with a variable icon {x} and a help icon. The 'Name' field is set to 'RegistryDetails'. The 'Value' field contains four concatenated variables: 'Registry Hive: registryHive', 'Registry Key: registryKey', 'Registry Value Type: registryValueTy...', and 'Registry Value: registryValue'. Each variable is preceded by a variable icon and has a close button (x). Below the variables is a dashed line and a cursor. At the bottom right of the action card is a link 'Add dynamic content' with a plus icon. Below the action card is a button 'Add an action' with a plus icon.

56. New Case.

57. Case 4 – Mailbox

58. Here are the headings for the MailboxDetails string:

UPN:

Display Name:

Address:

59. Complete as per screenshot:

Case 4

* Equals

Mailbox

{x} Append to string variable 7

* Name MailboxDetails

* Value

UPN: userPrincipalN... x

Display Name: mailboxDisplay... x

Address: mailboxAddress x

|

Add dynamic content +

Add an action

60. New Case.

61. Case 5 – MailCluster

62. Complete as per screenshot:

Case 5

* Equals

MailCluster

{x} Append to string variable 8

* Name MailClusterDetails

* Value

Cluster By: clusterBy x

Add dynamic content +

Add an action

63. New Case.

64. Case 6 – Ip

65. Complete as per screenshot:

The screenshot shows a configuration window for 'Case 6'. At the top, there is a section labeled '* Equals' with a text input field containing 'Ip'. Below this is a purple-bordered box titled '{x} Append to string variable 9'. Inside this box, the '* Name' field is set to 'IPDetails'. The '* Value' field contains a list of items: 'IP Address: ipAddress ×'. Below the list is a dashed line and a button labeled 'Add dynamic content'. At the bottom of the main configuration area, there is a button labeled 'Add an action'.

66. New Case.

67. Case 7 – MailMessage

68. Complete as per screenshot:

The screenshot shows a configuration window for 'Case 7'. At the top, there is a section labeled '* Equals' with a text input field containing 'MailMessage'. Below this is a purple-bordered box titled '{x} Append to string variable 10'. Inside this box, the '* Name' field is set to 'MailMessageDetails'. The '* Value' field contains a list of items: 'Sender: sender ×', 'Recipient: recipient ×', and 'Subject: subject ×'. At the bottom of the main configuration area, there is a button labeled 'Add an action'.

69. New Case. Final one!!

70. Case 8 – File

71. Complete as per screenshot:

Case 8

*Equals

File

{x} Append to string variable

*Name FileDetails

*Value

Filename: fileName x

Path: filePath x

Device ID: deviceId x

Add an action

72. Minimise the Switch and Entities loop, as per screenshot:

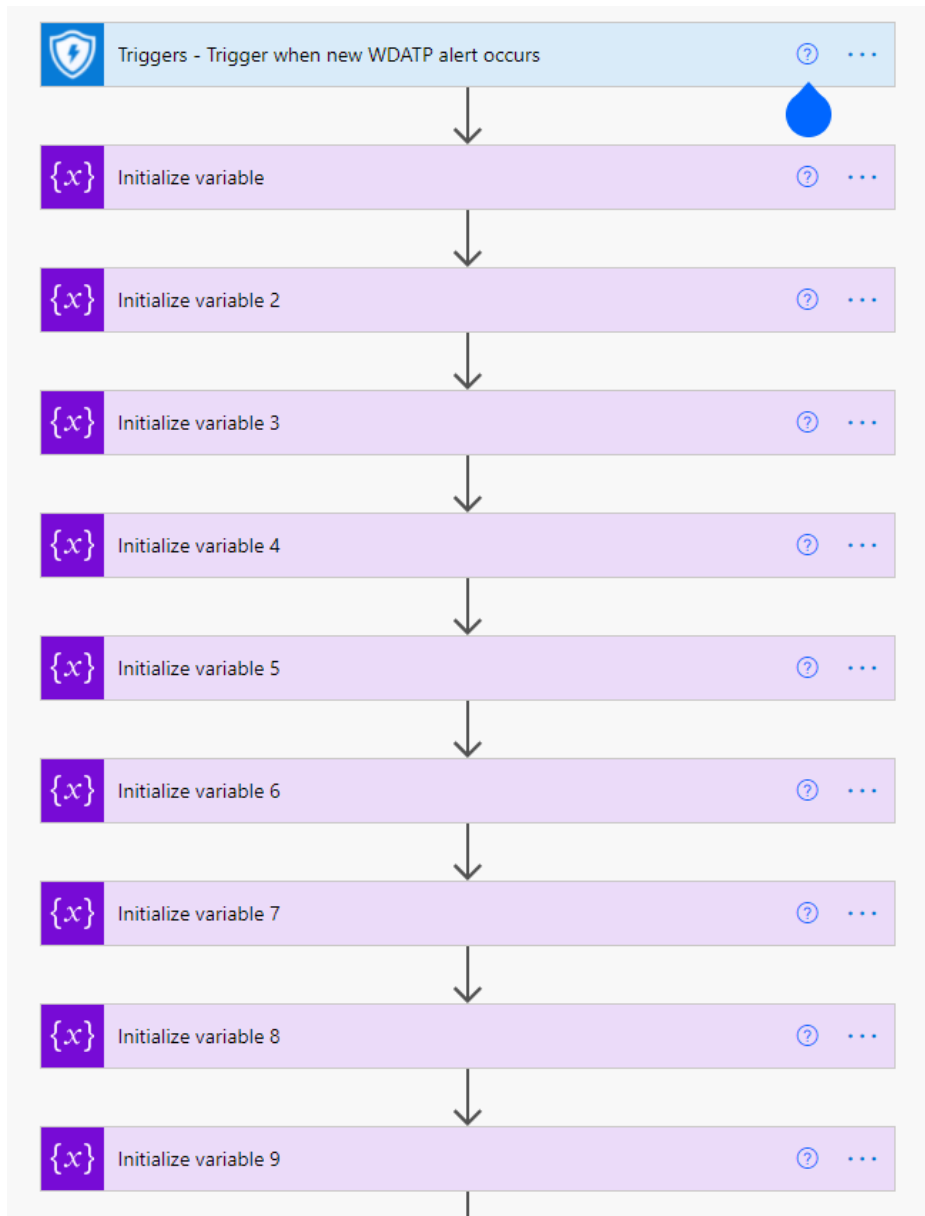


73. That it, no more loops!

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

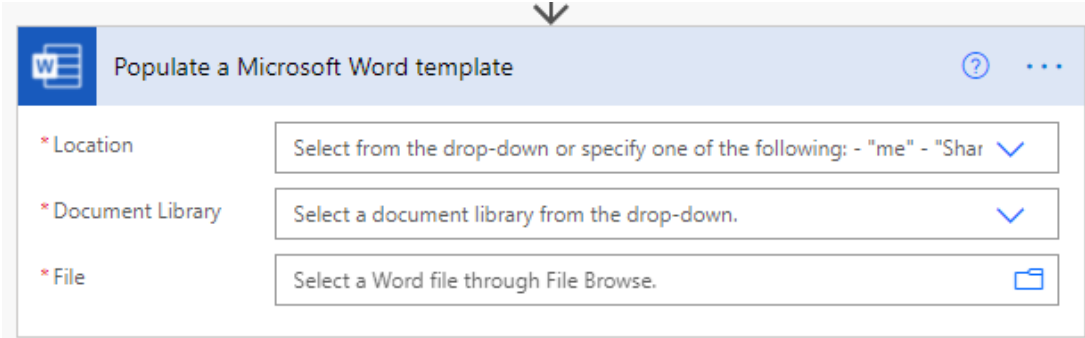
Populate the Incident Report

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





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.



Populate a Microsoft Word template

* Location Select from the drop-down or specify one of the following: - "me" - "Shar" ✓

* Document Library Select a document library from the drop-down. ✓

* File Select a Word file through File Browse. 📁

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.

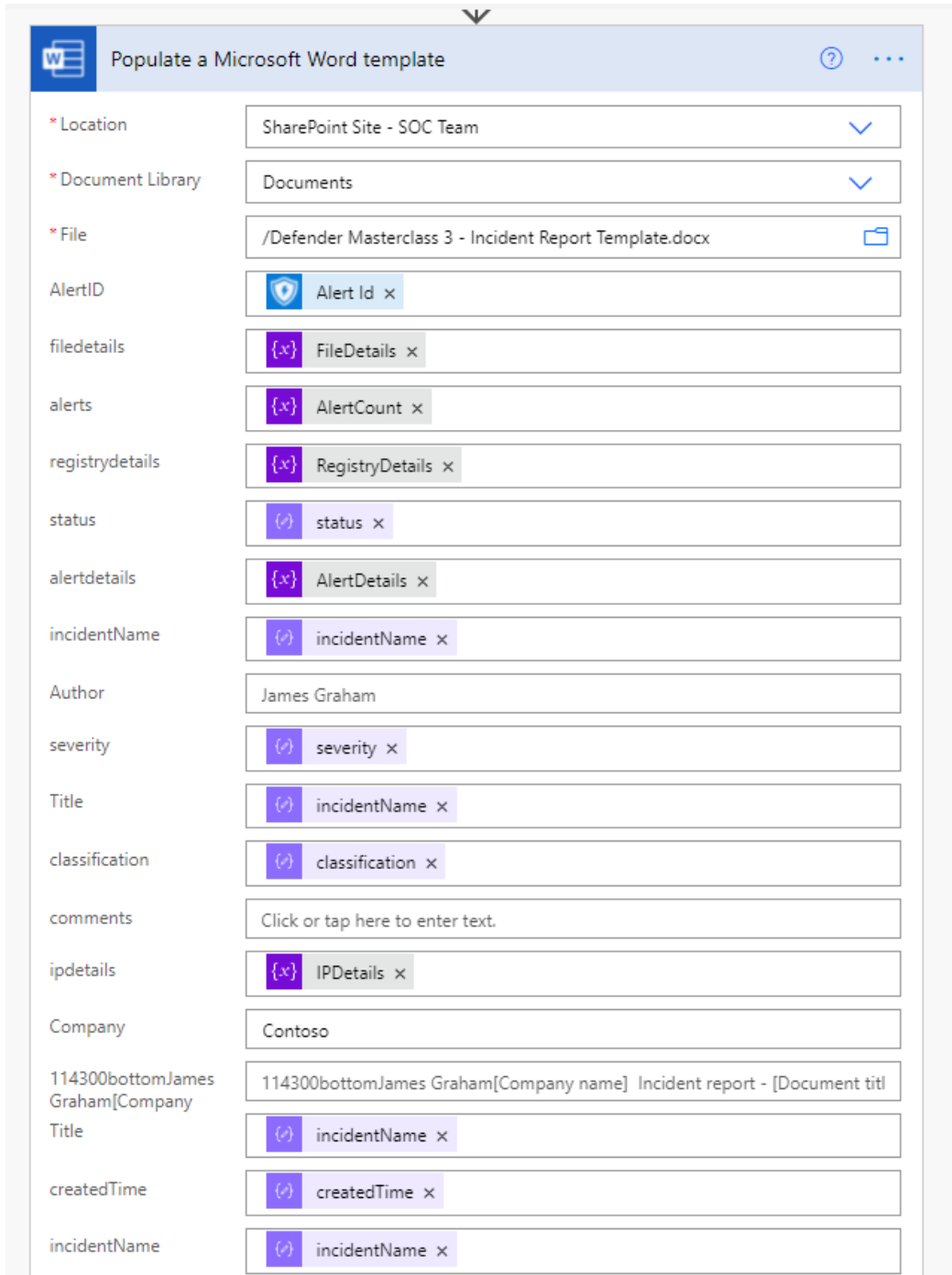
7. After you select the document the action box should expand to show all of our Content Controls that are contained in the document.

Populate a Microsoft Word template	
* Location	SharePoint Site - SOC Team
* Document Library	Documents
* File	/Defender Masterclass 3 - Incident Report Template.docx
Add dynamic content +	
AlertID	Alert ID goes here
filedetails	Click or tap here to enter text.
alerts	Click or tap here to enter text.
registrydetails	Click or tap here to enter text.
status	Click or tap here to enter text.
alertdetails	Click or tap here to enter text.
incidentName	Click or tap here to enter text.
Author	James Graham
severity	Click or tap here to enter text.
Title	[Document title]
classification	Click or tap here to enter text.
comments	Click or tap here to enter text.
ipdetails	Click or tap here to enter text.
Company	[Company name]
114300bottomJames Graham[Company	114300bottomJames Graham[Company name] Incident report - [Document tit
Title	[Document title]
createdTime	Click or tap here to enter text.

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.
 - a. Alert ID = AlertID (under trigger heading).
 - b. Filedetails = FileDetails (string variable).
 - c. Alerts = AlertCount (Integer variable).
 - d. Registrydetails = RegistryDetails (string variable).
 - e. Status = status (from Parse Incident JSON – there are two, pick the top one).
 - f. Alertdetails = AlertDetails (string variable).
 - g. IncidentName = incidentName (from Parse Incident JSON).
 - h. Severity = severity (from Parse Incident JSON – there are two, pick the top one).

- i. Title = incidentName (from Parse Incident JSON).
- j. Classification = classification (from Parse Incident JSON – pick first one).
- k. Comments = Leave blank <not implemented for this lab – would require a loop>.
- l. Ipdetails = IPDetails (string variable).
- m. Company = <Add company name>.
- n. Title = incidentName.
- o. CreatedTime = createdTime (from Parse Incident JSON – pick first one).
- p. IncidentName = incidentName.
- q. IncidentID = incidentID (pick top one).
- r. createdTime = createdTime (same as above).
- s. lastUpdateTime = lastUpdateTime (pick top one).
- t. assignedTo = assignedTo (pick top one).
- u. Determination = determination (pick top one).
- v. IncidentLink = <https://security.microsoft.com/incidents/<incidentID>/overview>
- w. deviceDetails = deviceDetails (string variable).
- x. Accountdetails = UserDetails (string variable).
- y. Processdetails = ProcessDetails (string variable).
- z. MailMessageDetails = MailMessageDetails (string variable).
- aa. Mailboxdetails = MailboxDetails (string variable).
- bb. Mailclusterdetails = MailClusterDetails (string variable).
- cc. Mitre = MITRETechniques (string variable).

10. Well done – should now look as follows:



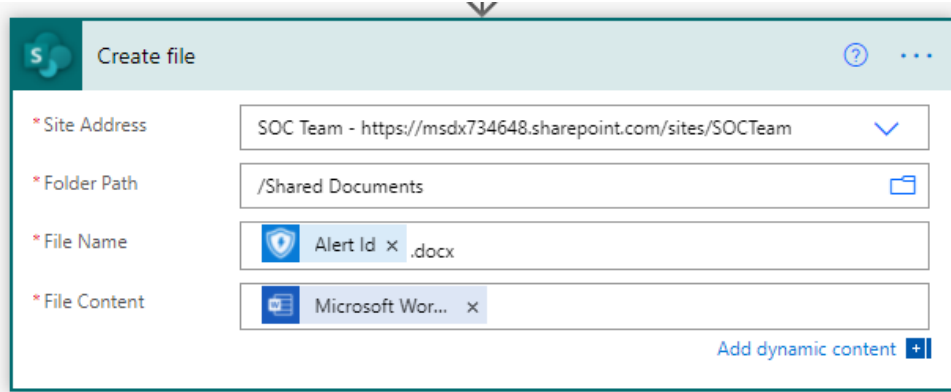
Populate a Microsoft Word template	
* Location	SharePoint Site - SOC Team
* Document Library	Documents
* File	/Defender Masterclass 3 - Incident Report Template.docx
AlertID	Alert Id
filedetails	FileDetails
alerts	AlertCount
registrydetails	RegistryDetails
status	status
alertdetails	AlertDetails
incidentName	incidentName
Author	James Graham
severity	severity
Title	incidentName
classification	classification
comments	Click or tap here to enter text.
ipdetails	IPDetails
Company	Contoso
114300bottomJames Graham[Company	114300bottomJames Graham[Company name] Incident report - [Document titl
Title	incidentName
createdTime	createdTime
incidentName	incidentName

incidentID	{x} incidentId x
createdTime	{x} createdTime x
lastUpdateTime	{x} lastUpdateTime x
assignedTo	{x} assignedTo x
determination	{x} determination x
incidentLink	https://security.microsoft.com/incidents/{x} incidentId x /overview
devicedetails	{x} DeviceDetails x
accountdetails	{x} UserDetails x
processdetails	{x} ProcessDetails x
mailMessageDetails	{x} MailMessageD... x
mailboxdetails	{x} MailboxDetails x
mailclusterdetails	{x} MailClusterDet... x
mitre	{x} MITRETechniqu... x

+ New step
Save

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 variable – 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:



The screenshot shows a 'Create file' dialog box in a SharePoint environment. The dialog has a light blue header with the 'S' logo and the title 'Create file'. Below the header, there are four input fields, each with a red asterisk indicating a required field. The first field is 'Site Address' with the value 'SOC Team - https://msdx734648.sharepoint.com/sites/SOCteam'. The second field is 'Folder Path' with the value '/Shared Documents'. The third field is 'File Name' with the value 'Alert Id .docx'. The fourth field is 'File Content' with the value 'Microsoft Wor...'. At the bottom right of the dialog, there is a blue link that says 'Add dynamic content' with a plus icon.

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](#)

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.

The screenshot shows the 'Your evaluation lab' dashboard in the Microsoft 365 security center. It features three main sections: 'Device allocation', 'Simulations overview', and 'Report overview'. The 'Device allocation' section shows '1 active device' (testmachine2) with a progress bar at 11/12. The 'Simulations overview' section shows '1 simulations executed' with a progress bar and a legend for Failed, Running, and Completed. The 'Report overview' section shows '4 Alerts in', '1 Incidents', '0 Actions taken in', '1 Investigations', and '0 Key findings'. Navigation links for 'View full list', 'Create simulation', 'Go to simulations gallery', and 'View full report' are provided.

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:

The screenshot shows the 'Incident Report Flow' summary dashboard in Microsoft Flow. It includes a 'Details' section with information about the flow (Incident Report Flow), its status (On), and its owner (MOD Administrator). The 'Connections' section lists three connected services: SharePoint Permissions, Microsoft Defender ATP, and Word Online (Business). The 'Owners' section shows the MOD Administrator. A '28-day run history' section is also visible, indicating when the flow runs.

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.

28-day run history ⓘ			All runs
Start	Duration	Status	
Jun 8, 05:13 PM (29 sec ago)	00:00:19	Succeeded	
Jun 8, 05:13 PM (30 sec ago)	00:00:24	Succeeded	
Jun 8, 05:13 PM (41 sec ago)	00:00:22	Succeeded	
Jun 8, 05:13 PM (49 sec ago)	00:00:18	Succeeded	

Also, navigate to the SOC Team SharePoint site to view the Incident Reports that have been generated.

ST

SOC Team

Private g

Home

Conversations

Documents

Shared with us

Notebook

Pages

Site contents

Recycle bin

Edit

+ New

Upload

Edit in grid view

Sync

Add shortcut to OneDrive

Export to Excel

Power Apps

Documents

	Name	Modified	Modified By	+ Add column
	General	May 29	MOD Administrator	
	da637587655975093501_-1810847705.docx	A few seconds ago	MOD Administrator	
	da637587656082530738_884354460.docx	A few seconds ago	MOD Administrator	
	da637587656196992425_-893701879.docx	A few seconds ago	MOD Administrator	
	da637587656201431029_1737577717.docx	A few seconds ago	MOD Administrator	
	Defender Masterclass 3 - Incident Report Te...	Sunday at 6:46 AM	MOD Administrator	

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:



da63758765619699
2425_-893701879.doc

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.