

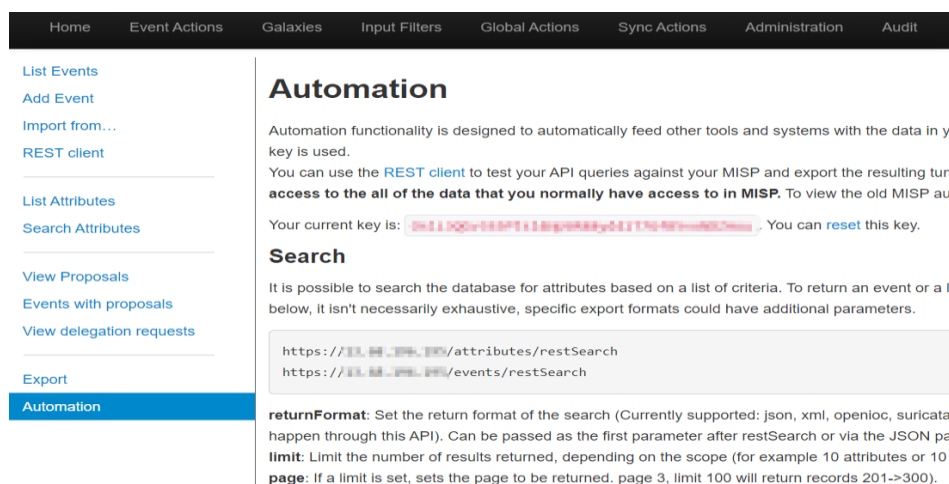
MISP TI Feeds integration with Microsoft Sentinel

Create an Azure AD App Registration

1. The first step is configuring an Azure AD App Registration that MISP will use to push IoCs into the Azure Security Graph.
2. In Azure Active Directory, navigate to App Registrations, and click New Registration. Name the Application: "MISP – Threat Intel Platform" and click Register.
3. Configure permission "ThreatIndicators.ReadWrite.OwnedBy" to the app and grant admin consent.
4. Save the Application id , Client secret and Tenant id.

MISP REST API

You need to get the MISP API key to use in the Export Script. In the MISP web interface, navigate to Event Actions > Automation, and get the API key to connect to the MISP API. You will need this in the next step.



Setup Export to Azure Graph

The next step is to configure a script to push the indicators from MISP into the Azure Security Graph. Run the following commands to download and configure the script.

```
sudo apt-get install python3-venv

python3 -m venv mispToSentinel

cd mispToSentinel

source bin/activate

git clone https://github.com/microsoftgraph/security-api-solutions

cd security-api-solutions/Samples/MISP/

pip install -r requirements.txt
```

** if you get the error while installing requirements.txt, open the txt file and get the list of packages needs to be installed for configuration and install the packages separately.

```
MISP@MISP: ~/mispToSentinel/security-api-solutions/Samples/MISP
$ cat requirements.txt
asn1crypto==0.24.0
awscli==1.16.20
botocore==1.12.10
certifi==2018.11.29
cffi==1.11.5
chardet==3.0.4
cryptography==2.4.2
idna==2.8
pycparser==2.19
pyOpenSSL==18.0.0
PySocks==1.6.8
requests==2.21.0
requests-futures==0.9.9
rsa==3.4.2
six==1.12.0
urllib3==1.24.2

"requirements.txt" 17L, 267C
```

Configure the Script

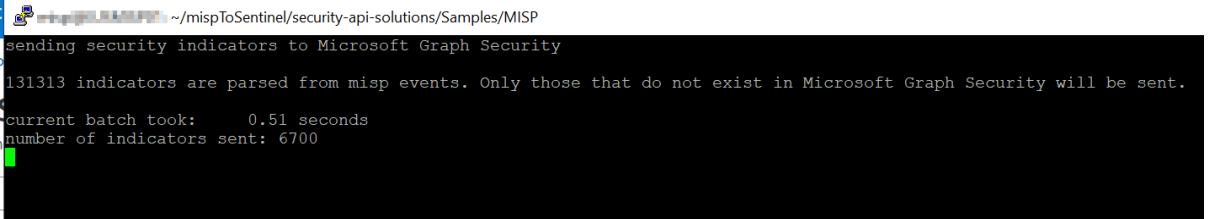
Run nano config.py to edit the Python script.

```
GNU nano 4.8
graph_auth = {
    'tenant': '<add tenant id>',
    'client_id': '<add client id >',
    'client_secret': '<add secret >',
}
targetProduct = 'Azure Sentinel'
misp_event_filters = {
    # 'org': '',
    # 'category': ''
}
action = 'alert'
passiveOnly = False
days_to_expire = 5
misp_key = 'your Azure Sentinel MISP Security API Key'
misp_domain = 'https://10.0.0.4/'
misp_verifycert = False
```

- Put Tenant , client id , client secret of application saved.
- If you have org. name and category in MISP instance put that in script , otherwise comment out the event filter section.
- Put MISP key and MISP domain address and save the config.py file.

Execute Script

Run `python script.py` to execute the export script. It will probably take several minutes to run.

- A terminal window with a black background and green text. The prompt is `~/mispToSentinel/security-api-solutions/Samples/MISP`. The output shows: `sending security indicators to Microsoft Graph Security`, `131313 indicators are parsed from misp events. Only those that do not exist in Microsoft Graph Security will be sent.`, `current batch took: 0.51 seconds`, and `number of indicators sent: 6700`.

```
~/mispToSentinel/security-api-solutions/Samples/MISP
sending security indicators to Microsoft Graph Security
131313 indicators are parsed from misp events. Only those that do not exist in Microsoft Graph Security will be sent.
current batch took: 0.51 seconds
number of indicators sent: 6700
```

Schedule Script: Schedule script to run every day to get updated feeds.

- Run `crontab -e` to edit your Crontab Entries.
- Add the following entry to execute the export script every day at 2 AM.
- `0 2 * * * /opt/MISP/jobs/Sample/python script.py`

Configure Azure Sentinel:

- Go to connector Section
- Search Threat Intelligence Platform connector and open connector page.
- Click on Connect button to connect the connector.