OPSWAT.

Integration Guide

Menlo Security

Menlo Security Integration Guide

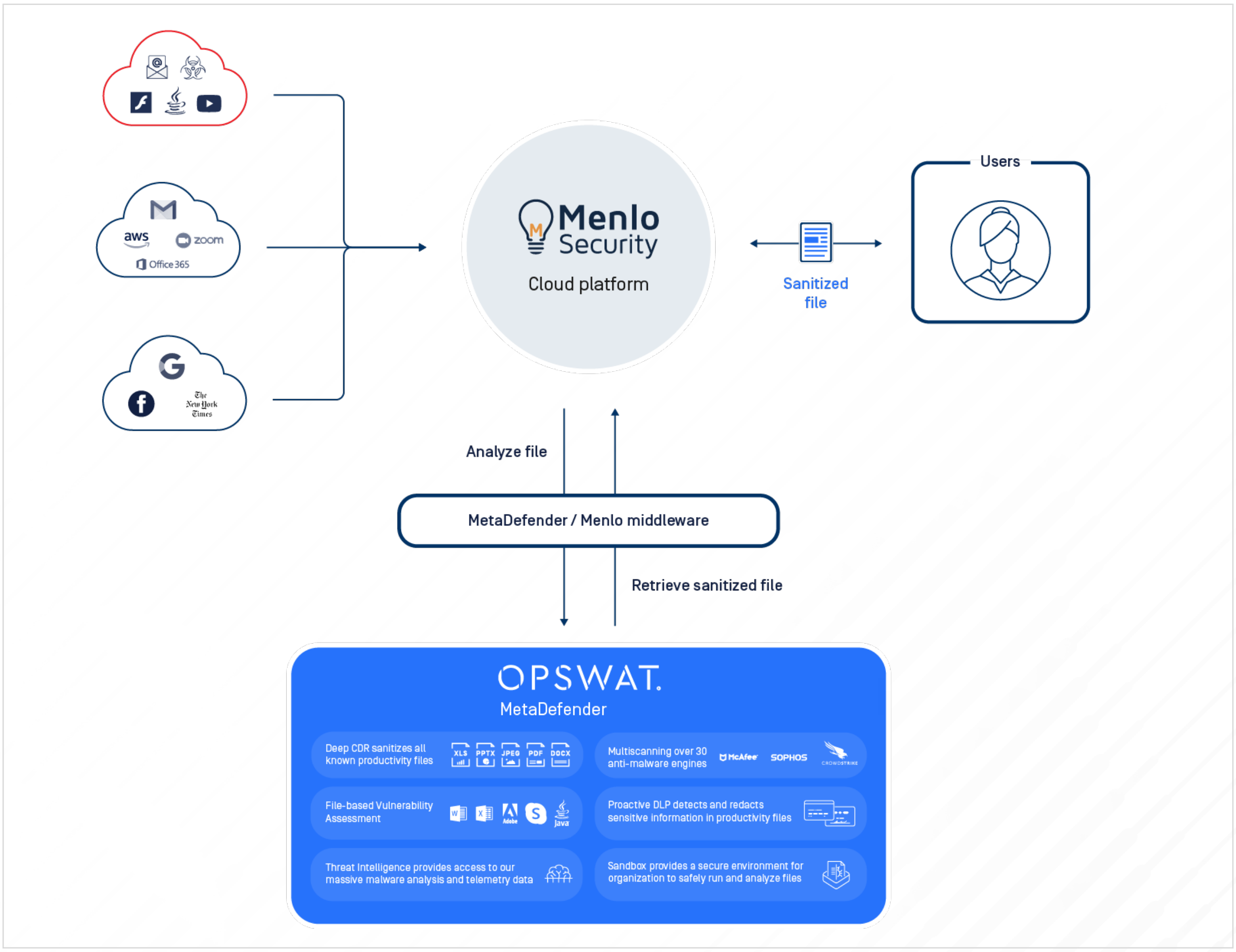
Overview

In this guide we will describe how to integrate OPSWAT's Metadefender Cloud and MetaDefender Core to your Menlo Security Remote Browser Isolation (RBI). Integrating these two solutions helps ensure a very high level of security as your RBI users will be protected from downloading malicious files to their local machines. OPSWAT's industry leading Deep CDR and Multiscanning technologies protect you from known, unknown, and advanced malware threats.

MetaDefender - Menlo Middleware

Introduction

Menlo Security Remote Browser Isolation (RBI) offers the ability to call an External API (called Sanitization API), that will offload the downloaded file to 3rd parties for deeper analysis.


aws
zoom
Office 365
The New York Times
Menlo Security
Cloud platform
Sanitized file
Users
Analyze file
MetaDefender / Menlo middleware
Retrieve sanitized file
OPSWAT.
MetaDefender
Deep CDR sanitizes all known productivity files
XLS
PPTX
JPEG
PDF
DOCX
Multiscanning over 30 anti-malware engines
McAfee
SOPHOS
CROWDSTRIKE
File-based Vulnerability Assessment
Adobe
Java
Proactive DLP detects and redacts sensitive information in productivity files
Threat Intelligence provides access to our massive malware analysis and telemetry data
Sandbox provides a secure environment for organization to safely run and analyze files

This integration will leverage Menlo's Sanitization API and will require a middleware that will translate the Sanitization API into MetaDefender API. This is a python based stateless application that can run on both Windows and Linux. Being lightweight, it is recommended to deploy it in a container environment, using a container orchestration (for e.g., Kubernetes).

If you plan to integrate with MetaDefender Cloud, OPSWAT hosts the middleware and you can connect Menlo Security to it without having to host the middleware yourself. If you plan to use MetaDefender Core you will need to host the middleware. Please view the contents below to host the middleware and integrate the solutions in the most optimal way.

MetaDefender Cloud Setup Guide

In order to connect with the middle-ware, you will have to configure the connection in Menlo's Admin console.
Navigate to Menlo Security (Web Policy > Content Inspection) and edit "Menlo Security Sanitization API"

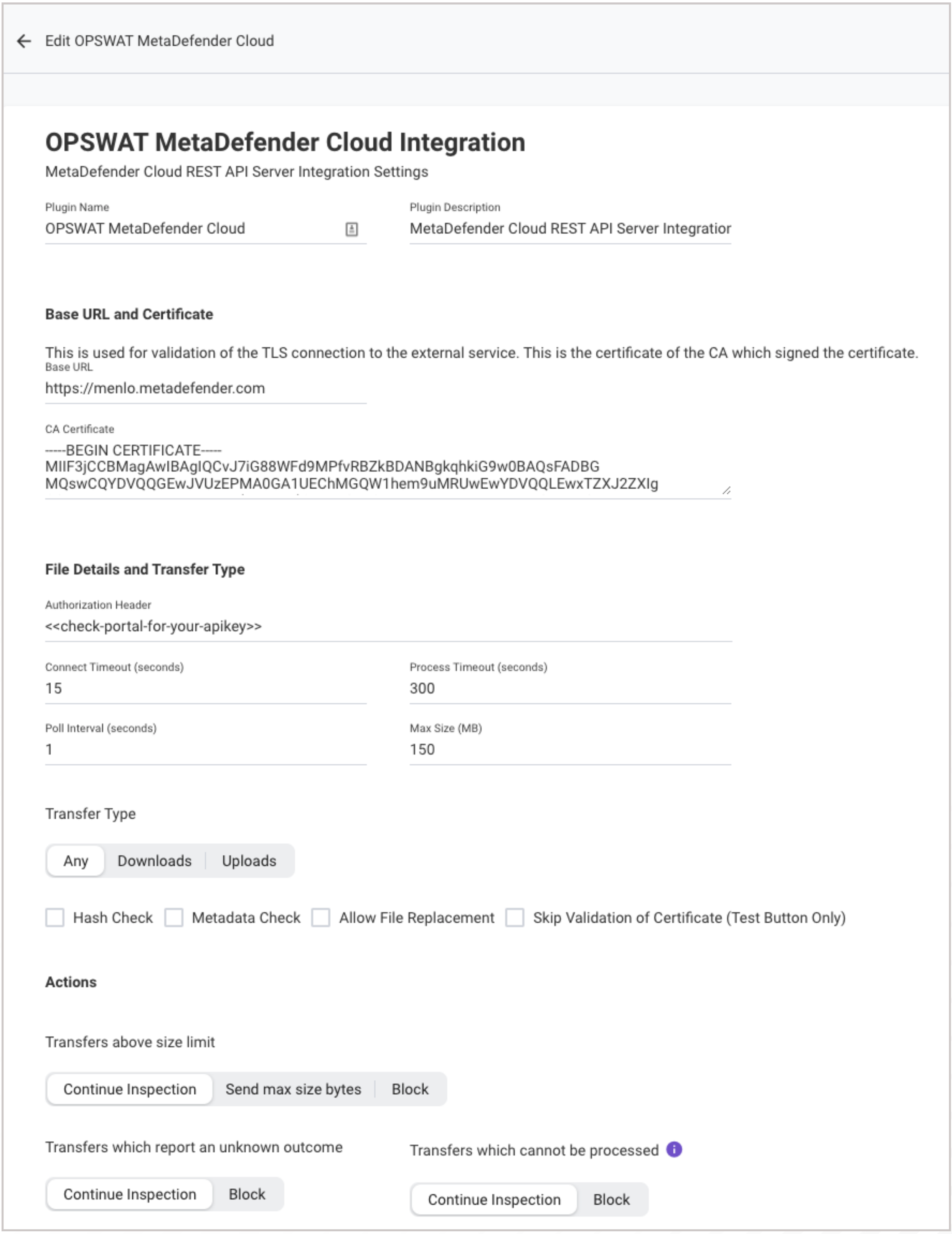
You will need to configure the fields in the modal:

- **Plugin Name:** Rename the plugin to the integration type (OPSWAT MetaDefender Cloud).
- **Plugin Description:** MetaDefender Cloud REST API Server Integration.
- **Base URL:** menlo.metadefender.com
- **CA Certificate:** This should be the **public certificate** for menlo.metadefender.com
- You can retrieve the rootCA of menlo.metadefender.com by using the following command:

```
openssl s_client --showcerts -connect menlo.metadefender.com:443 2>/dev/null </
dev/null |  sed -ne '/-BEGIN CERTIFICATE-/,/-END CERÅTIFICATE-/p'
```

- **Type of Transfer:** Downloads, Uploads or both
- **Authorization Header:** Your MetaDefender Cloud API Key
- **Connection Timeout:** you can leave the default as is.
- **Process Timeout**: in general the processing should happen within 1-2 seconds, but for very large files (or if it is an archive), could take a little longer. Balance this configuration with file size, archive handling and allowed file types configurations. Once the limit is reached, Menlo will consider the file processing to have failed.
- **Poll interval:** How often Menlo will check if the analysis has completed.
- **Max Size:** Set it based on your API Key limits: 140 MB (Personal), 256 MB (Commercial), 1024 MB (Enterprise)
- **Hash Check:** it will provide only the scan report, not the sanitized/redacted file. It is recommended to still submit the file.
- **Metadata Check:** *Not Used*
- **Allow File Replacement:** make sure you **check** this since this will allow you to replace the original file with the sanitized/redacted file.
- Leave **Continue Inspection** for the last 3 items.

See the following example:



Edit OPSWAT MetaDefender Cloud

OPSWAT MetaDefender Cloud Integration

MetaDefender Cloud REST API Server Integration Settings

Plugin Name: OPSWAT MetaDefender Cloud

Plugin Description: MetaDefender Cloud REST API Server Integratior

Base URL and Certificate

This is used for validation of the TLS connection to the external service. This is the certificate of the CA which signed the certificate.

Base URL: https://menlo.metadefender.com

CA Certificate:

```
-----BEGIN CERTIFICATE-----
MIIF3jCCBMagAwIBAgIQCvJ7iG88WFd9MPfvRBZkBDANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1hem9uMRUwEwYDVQQLEwxTZXJ2ZXIg
```

File Details and Transfer Type

Authorization Header: <<check-portal-for-your-apikey>>

Connect Timeout (seconds): 15	Process Timeout (seconds): 300
Poll Interval (seconds): 1	Max Size (MB): 150

Transfer Type: [✓ Any] Downloads Uploads

☐ Hash Check ☐ Metadata Check ☐ Allow File Replacement ☐ Skip Validation of Certificate (Test Button Only)

Actions

Transfers above size limit: [✓ Continue Inspection] Send max size bytes Block

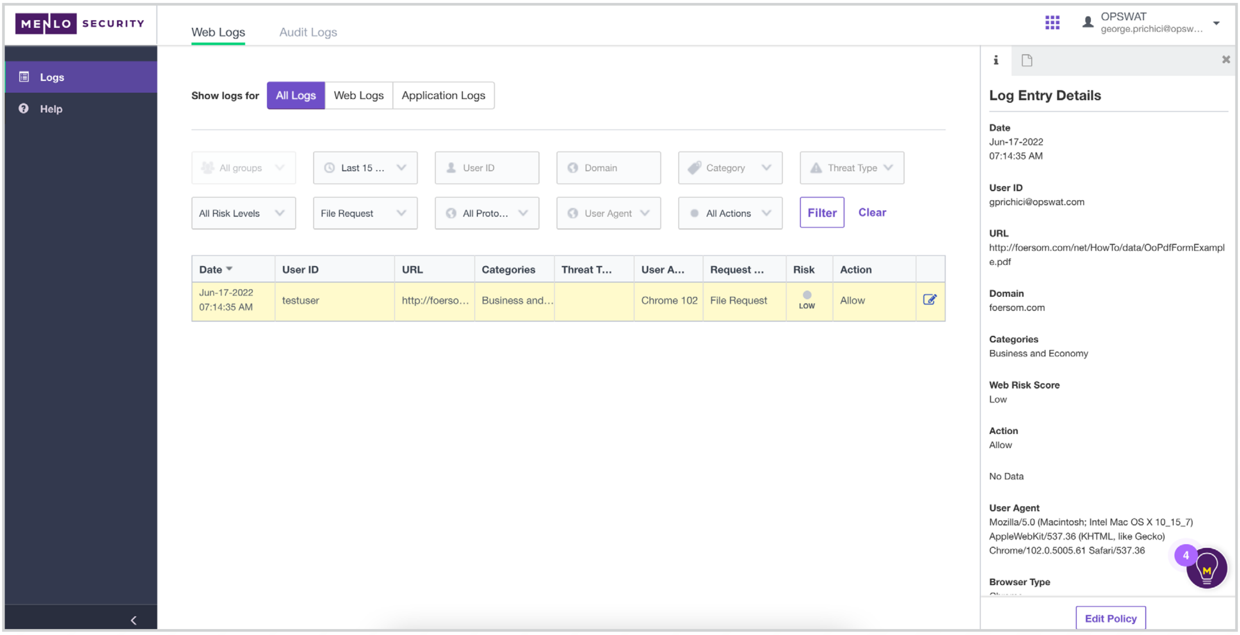
Transfers which report an unknown outcome: [✓ Continue Inspection] Block

Transfers which cannot be processed: [✓ Continue Inspection] Block

Test the MetaDefender Cloud Integration

Navigate to Menlo Security and start browsing. Search for a PDF/Docx on Google and try to open it. That will trigger the Sanitization API and send the file for analysis to MetaDefender.

Go to the Menlo Security Admin console and check the Logs > Web Logs section. Your first entry should have Request Type: File Request. Click on it and scroll down in the right panel to see the MetaDefender inspection results.



Date	User ID	URL	Categories	Threat T...	User A...	Request ...	Risk	Action
Jun-17-2022 07:14:35 AM	testuser	http://foerso...	Business and...		Chrome 102	File Request	LOW	Allow

Log Entry Details

Date
Jun-17-2022
07:14:35 AM

User ID
gprichici@opswat.com

URL
http://foersom.com/net/HowTo/data/OoPdfFormExampl
e.pdf

Domain
foersom.com

Categories
Business and Economy

Web Risk Score
Low

Action
Allow

No Data

User Agent
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/102.0.5005.61 Safari/537.36

Browser Type

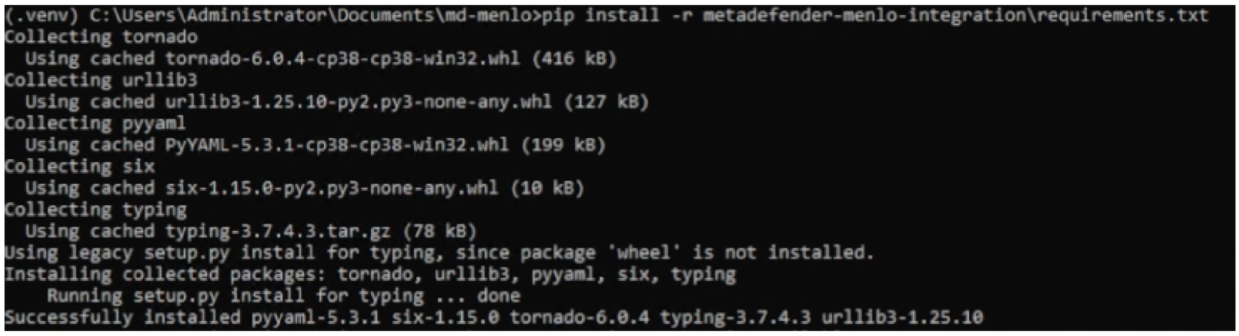
MetaDefender Core Setup Guide

System Requirements

The Middleware was written in python and requires `python 3.5` (released in Sep 2015) or above.
You can install all the required dependencies by running:

```
python -r requirements.txt
```

This will install all the dependencies (preferably in a previously built virtual environment):



```
(.venv) C:\Users\Administrator\Documents\md-menlo>pip install -r metadefender-menlo-integration\requirements.txt
Collecting tornado
  Using cached tornado-6.0.4-cp38-cp38-win32.whl (416 kB)
Collecting urllib3
  Using cached urllib3-1.25.10-py2.py3-none-any.whl (127 kB)
Collecting pyyaml
  Using cached PyYAML-5.3.1-cp38-cp38-win32.whl (199 kB)
Collecting six
  Using cached six-1.15.0-py2.py3-none-any.whl (10 kB)
Collecting typing
  Using cached typing-3.7.4.3.tar.gz (78 kB)
Using legacy setup.py install for typing, since package 'wheel' is not installed.
Installing collected packages: tornado, urllib3, pyyaml, six, typing
    Running setup.py install for typing ... done
Successfully installed pyyaml-5.3.1 six-1.15.0 tornado-6.0.4 typing-3.7.4.3 urllib3-1.25.10
```

Since python's GLI won't allow it to scale vertically, low-end systems are recommended, starting with 2 vCPU and 2GB memory for Linux based systems.

It is recommended to have multiple deployments, to avoid a single point of failure and ensure a high-available and fault-tolerant system.

⚠ HTTPS configuration is required!

Menlo Security will only accept files for analysis over HTTPS. Since these files could contain sensitive information, you wouldn't want it any other way.

With that in mind, make sure to have the right ports open and the right certificates in place. More details available in the Deployment section.

Configurations

The middleware offers the ability to connect to either MetaDefender Cloud or MetaDefender Core.
All the customizations can be done from the `config.yml` file available in the middleware:

```
service: metadefender
api:
  type: core
  params:
    apikey: null
  url:
    core: https://localhost:8008
    cloud: https://api.metadefender.com/v4
server:
  port: 3000
  host: "0.0.0.0"
  api_version: /api/v1
https:
  load_local: false
  crt: /tmp/certs/server.crt
  key: /tmp/certs/server.key
logging:
  enabled: true
  level: INFO
  logfile: /var/log/md-menlo/app.log
  interval: 24
  backup_count: 30
```

MetaDefender Core Configuration
First, select the integration via your own deployment of MetaDefender Core by setting core in the configuration `api > type` in `config.yml.`

For MetaDefender Core:

- `url > core` You'll be required to input the IP/URL to access MetaDefender Core. Depending on your networking configuration, this could be a publicly accessible service, or you can allow it to accept requests only from the middleware.
- If MetaDefender is accessible from outside the organization, it is advisable to use the public IP/URL, instead of internal IPs since this information will be used to list a Report URL in Menlo Security reports.

Server configuration
Under the `server` configuration , you can modify the service's running `port` . Even though by default it is set as `3000`, you can modify it. If the goal is to run HTTPS using 443, it is recommended to place a reverse proxy (like nginx) in front of the middleware cluster.

The current version of Menlo Security Sanitization API is `v1` , so you shouldn't modify the `api_version` configuration.

HTTPS Configuration
As mentioned before, HTTPS is required for the middleware.

This can be set directly in the middleware, by using the configuration under `https` . If you want the middleware to run in HTTPS mode, you'll have to set `load_local: true` . This will force the app to load the `certfile` and `keyfile` from the locations specified in the `crt` and `key` fields. These can be relative paths.

Self-signed certificates are allowed. But are required to be signed by a root CA.

If you would like to use self-signed certificates, it is required to have a server certificate that is signed by a rootCA. The rootCA public certificate will be uploaded in the Menlo Security Admin console. If the wrong certificate is uploaded, the requests will fail.
It is also recommended to use X509 certificates in pem format. Password protected certificates are not supported.

Logging
By default, logging is enabled at `INFO` level.

You should specify the absolute path for the `logfile` and make sure that the path exists and the write permissions are in place, otherwise, the middleware fails to initialize.

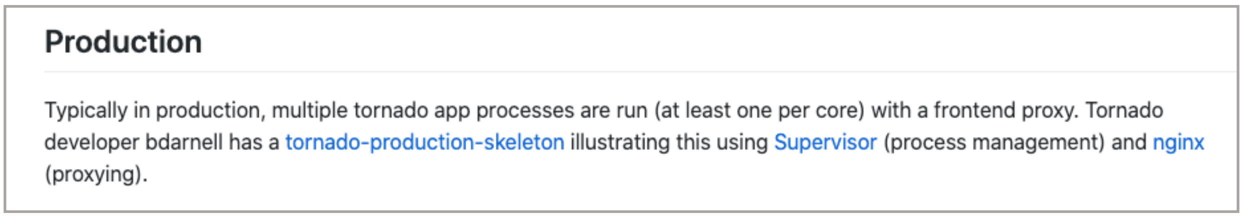
By default log rotation is set for 30 days, with a rotation interval of 1 day (24h).

Deployment

Standalone app
First, build the application.
To run the middleware is as straightforward as:

```
python -m metadefender_menlo
```

Make sure python is version 3.5 and above.
However, in a production environment, Tornado recommends running it behind a front-end proxy, that can also handle HTTPS termination. A great example is nginx:



Production

Typically in production, multiple tornado app processes are run (at least one per core) with a frontend proxy. Tornado developer bdarnell has a tornado-production-skeleton illustrating this using Supervisor (process management) and nginx (proxying).

You'll probably want to run it as daemon/service, to be able to recover and start at boot time. You can review the official Tornado recommendation on how to achieve that.

Another great resource is eklitzke.org/production-tornado-deployment-with-systemd, which explains how to run and leverage multiple python processes and how to put them behind an nginx server.

Kubernetes
First, you're required to build a container. There's a `Dockerfile` in the repo, that you can use to build the container and push it to your registry. Once you have it, you will have to modify `deployment.yaml` and specify your own container image.

Also, check the `deploy.sh` script. It was built specifically for Google Cloud to leverage GKE (Google Kubernetes Engine). But it can be easily adapted to run in any Kubernetes supported environment.

The GCP specific part is building the cluster and (if needed) the static IP.

Modify the deploy script to use your own cluster and certificates as needed. If you don't require self-signed certs, you can remove the entire portion with openssl and jump directly to adding your certificates as Kubernetes secrets.

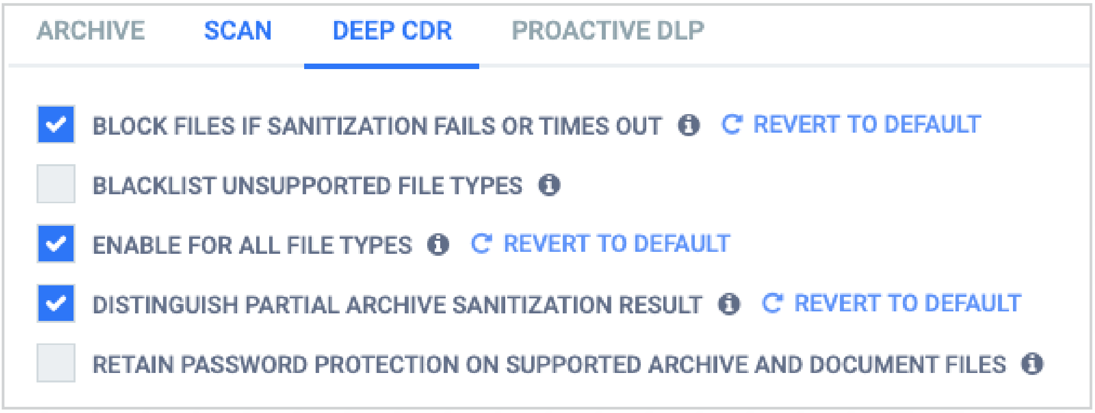
Check `deployment.yaml` and place the real apikey (if you plan to use MetaDefender Cloud), or just remove that environment variable entirely if it is not going to be used. You can also set the apikey in the config file, if you prefer to have it hardcoded in the container image, instead of passing it as an environment variable.

Integration with MetaDefender Core

In the `config.yml` you need to specify that the middleware will connect to MetaDefender Core by setting `api.type: core` and specifying the correct path to MetaDefender Core REST API in `url.core` .

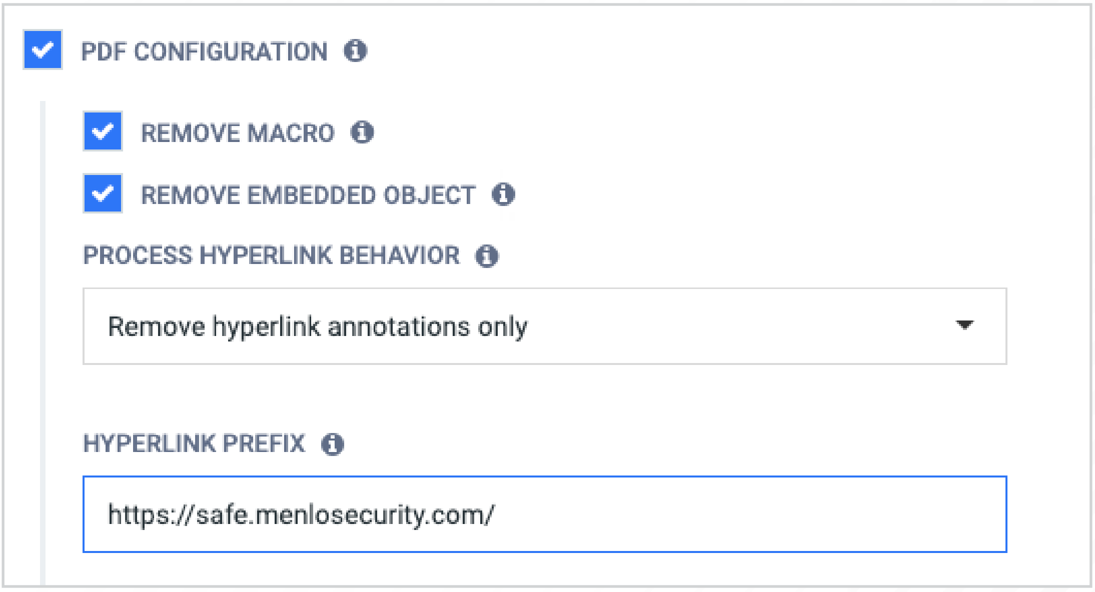
Other than that, you can rely on the existing Workflow Rule in MetaDefender Core. It is recommended to enable the following:

- **Archive processing:** this will increase the detection ratio, including for productivity files (Docx, Xlsx, etc.).
- **Multiscanning:** you can configure the maximum allowed file size for scanning and additional parameters for the scanning process.
- **Deep CDR:** we strongly recommend to check "Enable for all file types" and "Block files if sanitization fails or times out" (high probability that the failure is caused by invalid file structure):



- [✓] BLOCK FILES IF SANITIZATION FAILS OR TIMES OUT
- [ ] BLACKLIST UNSUPPORTED FILE TYPES
- [✓] ENABLE FOR ALL FILE TYPES
- [✓] DISTINGUISH PARTIAL ARCHIVE SANITIZATION RESULT
- [ ] RETAIN PASSWORD PROTECTION ON SUPPORTED ARCHIVE AND DOCUMENT FILES

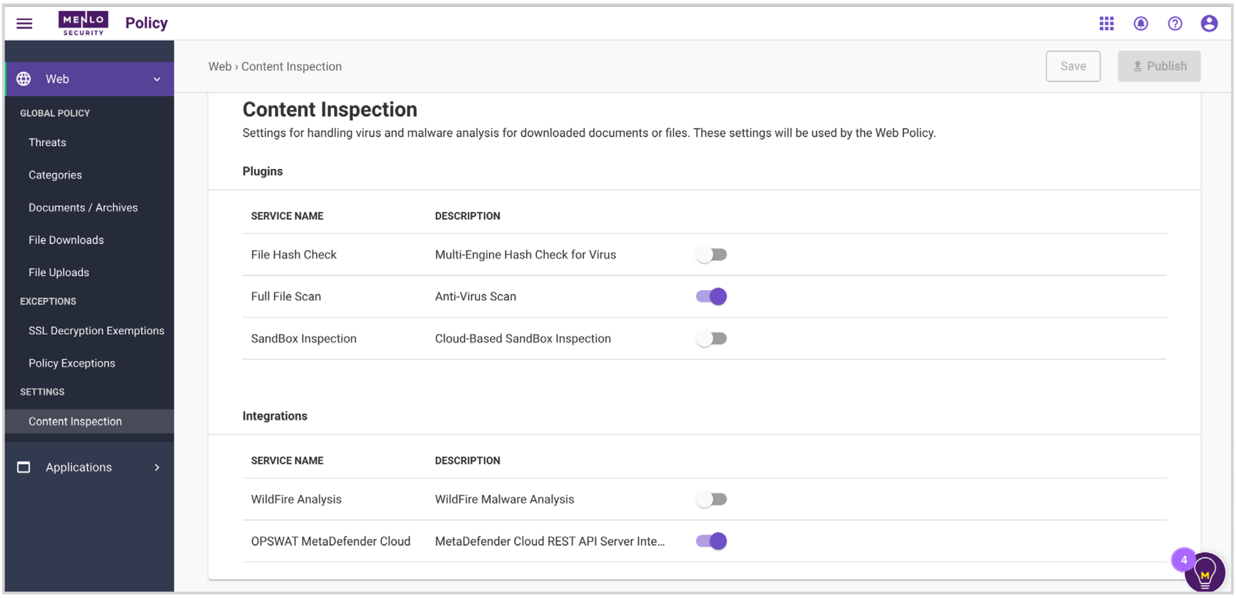
Another very important configuration would be to **sanitize all the hyperlinks** in documents. If a document is being downloaded from the internet and has a potentially bad link, it could bypass the isolation and run on the local machine. You can configure the Deep CDR engine to always sanitize the links and force them to go through Menlo Security to ensure isolation:



- [✓] PDF CONFIGURATION
  - [✓] REMOVE MACRO
  - [✓] REMOVE EMBEDDED OBJECT
  - PROCESS HYPERLINK BEHAVIOR: Remove hyperlink annotations only
  - HYPERLINK PREFIX: https://safe.menlosecurity.com/

Menlo Security Configuration

In order to connect with the middleware, you will have to configure the connection in Menlo's Admin console. Navigate to admin.menlosecurity.com/#/policy/inspection and edit Menlo Security Sanitization API (Web Policy > Content Inspection > Menlo Security Sanitization API):



Content Inspection

Settings for handling virus and malware analysis for downloaded documents or files. These settings will be used by the Web Policy.

Plugins

SERVICE NAME	DESCRIPTION	
File Hash Check	Multi-Engine Hash Check for Virus	off
Full File Scan	Anti-Virus Scan	on
SandBox Inspection	Cloud-Based SandBox Inspection	off

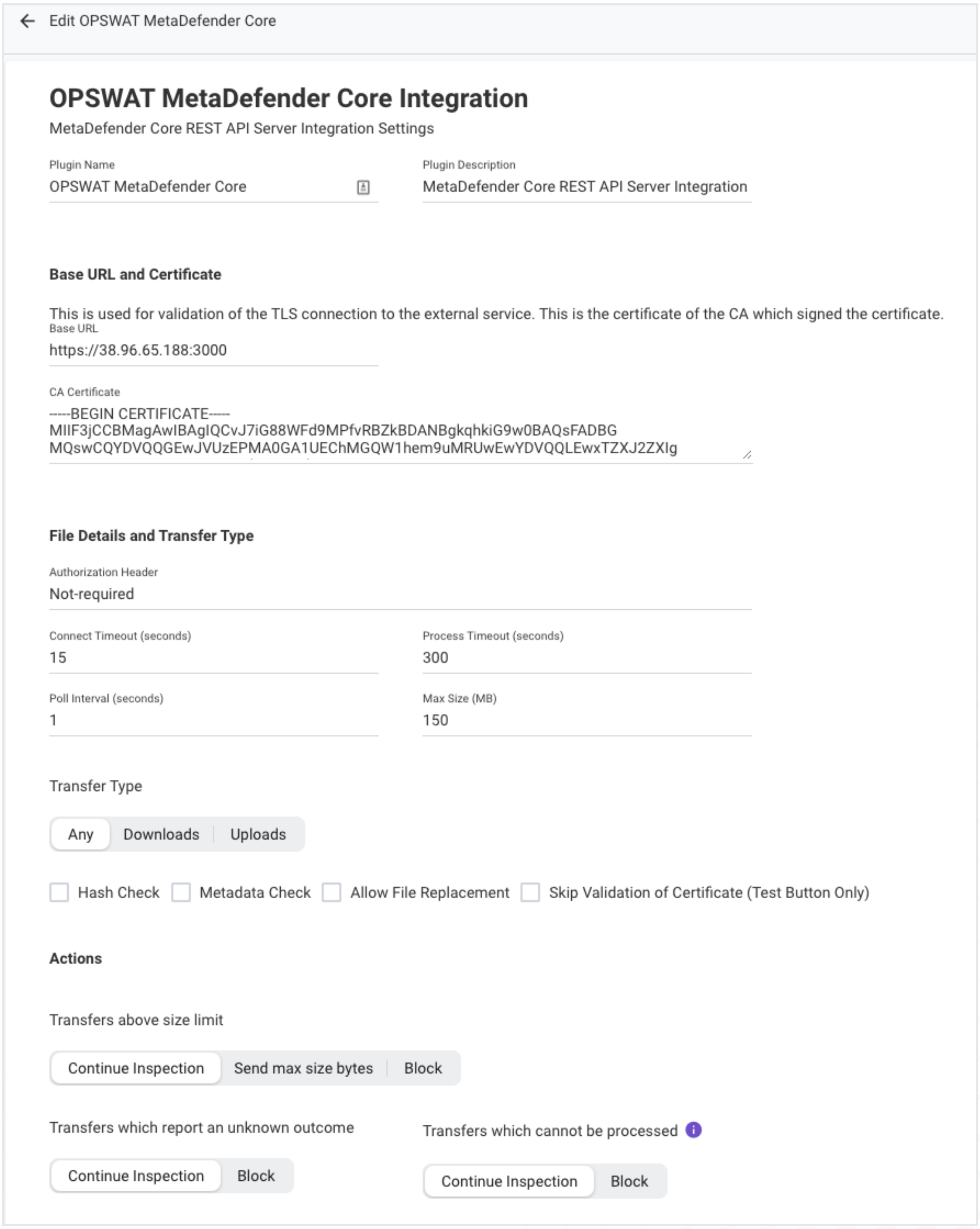
Integrations

SERVICE NAME	DESCRIPTION	
WildFire Analysis	WildFire Malware Analysis	off
OPSWAT MetaDefender Cloud	MetaDefender Cloud REST API Server Inte...	on

You will need to configure the fields in the modal:

- **Plugin Name:** Rename the plugin to the integration type (OPSWAT MetaDefender Core).
- **Plugin Description:** MetaDefender Core REST API Server Integration.
- **Base URL:** This is the middleware URL (needs to be an HTTPS URL).
- **Certificate:** This should be the rootCA public certificate, that was used to sign the middleware certificate.
- **Type of Transfer**: Downloads, Uploads or both.
- **Connection Timeout:** you can leave the default as is.
- **Process Timeout:** in general the processing should happen within 1-2 seconds, but depending on the file size (or if it is an archive), could take a little longer. Balance this configuration with file size, archive handling and allowed file types configurations.
- **Poll interval:** How often Menlo will check if the analysis is completed.
- **Max Size:** For MetaDefender Core could be higher values
- **Allow File Replacement:** make sure you check this since this will allow you to replace the original file with the sanitized/redacted file.
- **Unused fields:**
  - **Authorization Header**
  - Metadata Check
  - Leave **Continue Inspection** for the last 3 items.

See the following example:



OPSWAT MetaDefender Core Integration

MetaDefender Core REST API Server Integration Settings

Plugin Name: OPSWAT MetaDefender Core

Plugin Description: MetaDefender Core REST API Server Integration

Base URL and Certificate

This is used for validation of the TLS connection to the external service. This is the certificate of the CA which signed the certificate.

Base URL: https://38.96.65.188:3000

CA Certificate:

```
-----BEGIN CERTIFICATE-----
MIIF3jCCBMagAwIBAgIQCvJ7iG88WFd9MPfvRBZkBDANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1hem9uMRUwEwYDVQQLEwxTZXJ2ZXIg
```

File Details and Transfer Type

Authorization Header: Not-required

Connect Timeout (seconds): 15

Process Timeout (seconds): 300

Poll Interval (seconds): 1

Max Size (MB): 150

Transfer Type: [✓ Any] Downloads Uploads

☐ Hash Check ☐ Metadata Check ☐ Allow File Replacement ☐ Skip Validation of Certificate (Test Button Only)

Actions

Transfers above size limit: [✓ Continue Inspection] Send max size bytes Block

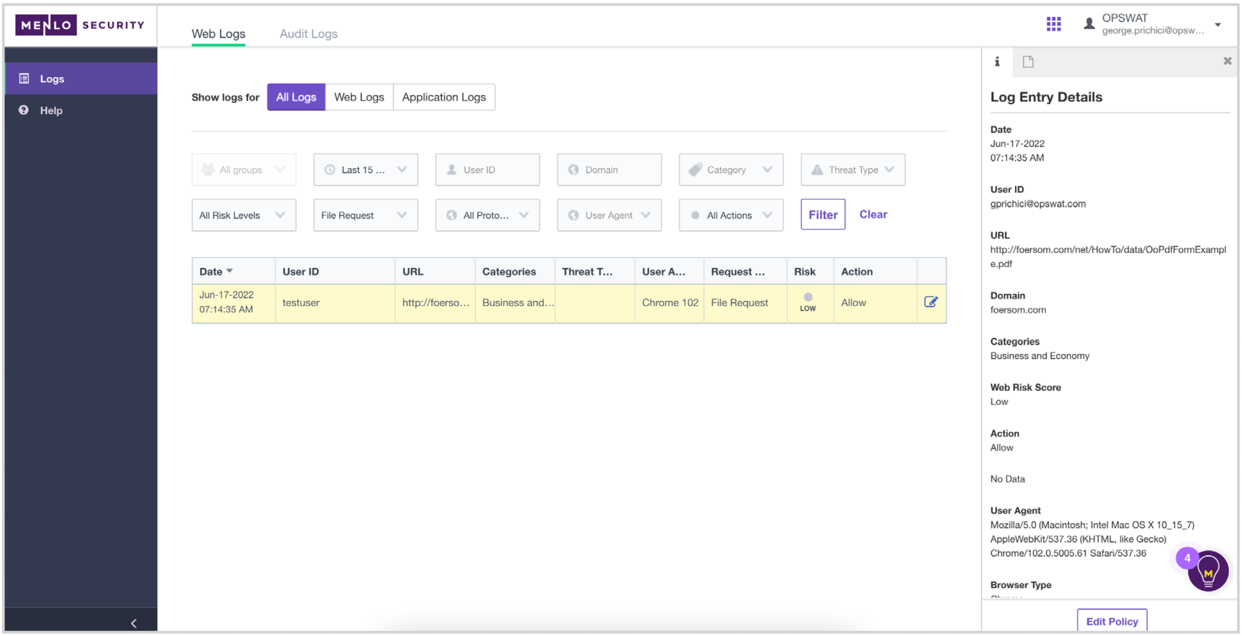
Transfers which report an unknown outcome: [✓ Continue Inspection] Block

Transfers which cannot be processed: [✓ Continue Inspection] Block

Test the integration

Navigate to safe.menlosecurity.com and start browsing. Search for a PDF/Docx on Google and try to open it. That will trigger the Sanitization API and send the file for analysis to MetaDefender.

If you enabled 'watermark' in MetaDefender Core, your rendered/downloaded file should contain a watermark. If not, go to the Menlo Security Admin console and check the Logs > Web Logs section and your first entry should have Request Type: File Request. Click on it and scroll down in the right panel to see the MetaDefender inspection results.



Date	User ID	URL	Categories	Threat T...	User A...	Request ...	Risk	Action
Jun-17-2022 07:14:35 AM	testuser	http://foerso...	Business and...		Chrome 102	File Request	LOW	Allow

Log Entry Details

Date
Jun-17-2022
07:14:35 AM

User ID
gprichici@opswat.com

URL
http://foersom.com/net/HowTo/data/OoPdfFormExample.pdf

Domain
foersom.com

Categories
Business and Economy

Web Risk Score
Low

Action
Allow

No Data

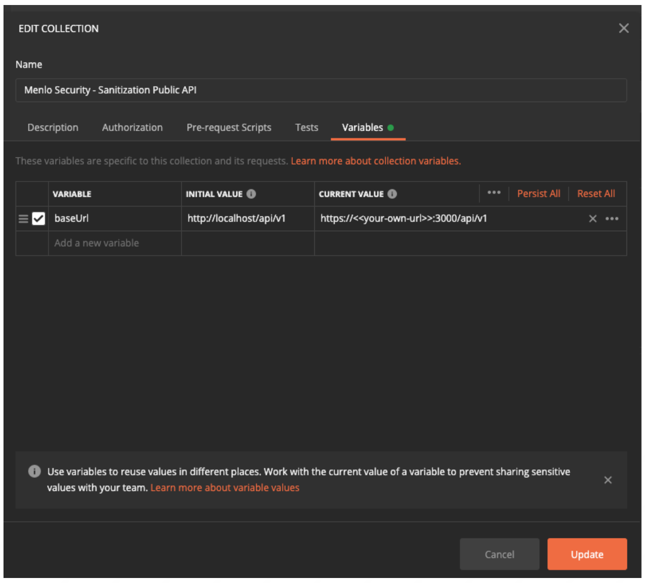
User Agent
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.61 Safari/537.36

Browser Type

Troubleshooting Guide

How can I check the Middleware is running properly?

You have a postman collection in the repository.
Import it in your Postman app and set the proper baseURL.


EDIT COLLECTION

Name: Menlo Security - Sanitization Public API

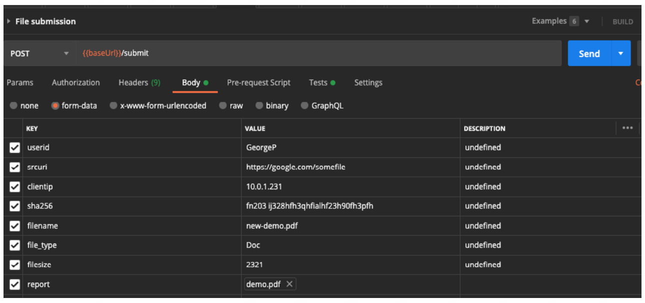
These variables are specific to this collection and its requests. Learn more about collection variables.

VARIABLE	INITIAL VALUE	CURRENT VALUE
✓ baseUrl	http://localhost/api/v1	https://<<your-own-url>>:3000/api/v1
Add a new variable		

Use variables to reuse values in different places. Work with the current value of a variable to prevent sharing sensitive values with your team. Learn more about variable values

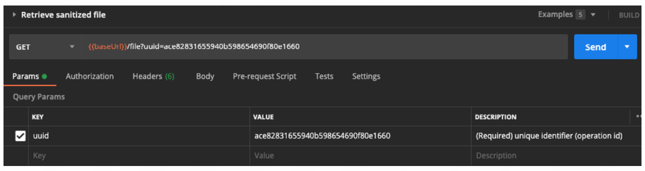
You should either import the certificate in *Settings > Certificates > CA Certificates* or disable *SSL certificate verification in Settings > General*

Then do a quick check to see that file submission and analysis results work properly. You should also try to retrieve the sanitized file, to make sure that the MetaDefender configuration was set properly and CDR configuration is enabled.


File submission

POST {{baseUrl}}/submit

	KEY	VALUE	DESCRIPTION
✓	userid	GeorgeP	undefined
✓	srcuri	https://google.com/somefile	undefined
✓	clientip	10.0.1.231	undefined
✓	sha256	fn203 ij328hfh3qhfialhf23h90fh3pfh	undefined
✓	filename	new-demo.pdf	undefined
✓	file_type	Doc	undefined
✓	filesize	2321	undefined
✓	report	demo.pdf	


Retrieve sanitized file

GET {{baseUrl}}/file?uuid=ace82831655940b598654690f80e1660

Query Params

	KEY	VALUE	DESCRIPTION
✓	uuid	ace82831655940b598654690f80e1660	(Required) unique identifier (operation id)
	Key	Value	Description

If any of the requests fail, check the middleware app logs.

Middleware is available, but it shows "Skipped" in Menlo logs

You can try to validate the integration by trying to mimic Menlo's behavior.
In order to make sure the request is completed successfully, you should try to run the following (or similar):

```
curl --cacert rootCA.crt https://middleware.your-corporate.com:
<<port>>/api/v1/health
```

In case the following request fails, you probably either specified the wrong URL or provided the wrong rootCA.crt. Check the error message for hints.

Make sure that:

- You are using X509 certificates;
- CA Cert is in PEM format.

How do I get the CA certificate?

You can try to retrieve the rootCA by using the following command:

```
openssl s_client -showcerts -servername server -connect server:443 > cacert.pem
```

Resources

Find out more about Menlo Security at www.menlosecurity.com.

Connect with an OPSWAT technical expert at www.opswat.com/get-started to schedule a demo and see how the Metadefender platform can help enhance the security of your organization. Learn more at www.opswat.com.

OPSWAT.
Protecting the World's Critical Infrastructure