PasteHunter Documentation

Release 1.0

Kevin Breen

Oct 07, 2019

Contents:

PasteHunter is a python3 application that is designed to query a collection of sites that host publicliy pasted data. For all the pasts it finds it scans the raw contents against a series of yara rules looking for information that can be used by an organisation or a researcher.

CHAPTER 1

Installation

There are a few ways to install

1.1 Local Installation

1.1.1 Pastehunter

If you want to run the latest stable version grab the latest release from https://github.com/kevthehermit/PasteHunter/releases. If you want to run the development version clone the repository or download the latest archive.

Pastehunter has very few dependancies you can install all the python libraries using the requirements.txt file and `sudo pip3 install -r requirements.txt`

1.1.2 Yara

Yara is the scanning engine that scans each paste. Use the official documentation to install yara and the python3 library. https://yara.readthedocs.io/en/latest/gettingstarted.html#compiling-and-installing-yara

All yara rules are stored in the YaraRules directory. An index.yar file is created at run time that includes all additional yar files in this directory. To add or remove yara rules, simply add or remove the rule file from this directory.

1.1.3 Elastic Search

If you want to use the elastic search output module you will need to install elastic search. Pastehunter has been tested with version 6.x of Elasticsearch. To install follow the offical directions on https://www.elastic.co/guide/en/elasticsearch/reference/current/deb.html.

You will also need the elasticsearch python library which can be installed using `sudo pip3 install elasticsearch`.

1.1.4 Kibana

Kibana is the frontend search to Elasticsearch. If you have enabled the Elasticsearch module you probably want this. To install follow the offical directions on https://www.elastic.co/guide/en/kibana/current/deb.html.

1.2 Docker Installation

You will find a Dockerfile that will build the latest stable version of PasteHunter.

This can be used with the included docker-compose.yml file. A sample podspec for kubernets is coming soon.

1.3 Configuration

Before you can get up and running you will need to set up the basic config. Copy the settings.json.sample to settings.json and edit with your editor of choice.

1.3.1 Yara

- **rule_path**: defaults to the YaraRules directory in the PasteHunter root.
- **blacklist**: If set to true, any pastes that match this rule will be ignored.
- **test_rules**: Occasionaly I release some early test rules. Set this to `true` to use them.

1.3.2 log

Logging for the application is configured here.

- **log_to_file**: true or false, default is stdout.
- **log_file**: filename to log out to.
- **logging_level**: numerical value for logging level see the table below.
- **log_path**: path on disk to write log_file to.
- **format**: python logging format string - https://docs.python.org/3/library/logging.html#formatter-objects

Level	Numerical
CRITICAL	50
ERROR	40
WARNING	30
INFO	20
DEBUG	10
NETSET	0

1.3.3 general

General config options here.

- **run_frequency**: Sleep delay between fetching list of inputs to download. This helps rate limits.

For Input, Output and Postprocess settings please refer to the relevant sections of the docs.

1.4 Starting

You can run pastehunter by calling the script by name.

`python3 pastehunter.py`

1.4.1 Service

You can install pastehunter as a service if your planning on running for long periods of time. An example systemd service file is show below

Create a new service file `/etc/systemd/system/pastehunter.service`

Add the following text updating as appropriate for your setup paying attention to file paths and usernames.:

```
[Unit]
Description=PasteHunter

[Service]
WorkingDirectory=/opt/PasteHunter
ExecStart=/usr/bin/python3 /opt/PasteHunter/pastehunter.py
User=localuser
Group=localuser
Restart=always

[Install]
WantedBy=multi-user.target
```

Before starting the service ensure you have tested the pastehunter app on the command line and identify any errors. Once your ready then update systemctl `systemctl daemon-reload` enable the new service `systemctl enable pastehunter.service` and start the service `systemctl start pastehunter`

CHAPTER 2

Inputs

This page details all the configuration options per input.

There are a few generic options for each input. - **enabled**: This turns the input on and off. - **store_all**: ignore the only store on matching rule. - **module**: This is used internally by pastehunter.

2.1 Pastebin

To use the pastebin API you need an API key. These need to be purchased and are almost always on some sort of offer! https://pastebin.com/pro The API uses your IP to authenticate instead of a key. You will need to whitelist your IP at https://pastebin.com/api_scraping_faq

- **api_scrape**: The URL endpoint for the list of recent paste ids.
- **api_raw**: The URL endpoint for the raw paste.
- **paste_limit**: How many pasteids to fetch from the recent list.
- **store_all**: Store all pastes regardless of a rule match.

2.2 Github Gists

Github has an API that can be used at no cost to query recent gists. There are two options here.

- Without an access key - You will have a low rate limit.
- With an access key - You will have a higher rate limit.

The unauthenticated option is not suitable for pastehunter running full time. To create your key visit https://github.com/settings/tokens

YOU DO NOT NEED TO GIVE IT ANY ACCESS PERMISSIONS

- **api_token**: The token you generated.

- **api_limit**: Rate limit to prevent being blocked.
- **store_all**: Store all pastes regardless of a rule match.
- **user_blacklist**: Do not process gists created by these usernames.
- **file_blacklist**: Do not process gists that match these filenames.

2.3 Slexy

Slexy has some heavy rate limits on it.

- **store_all**: Store all pastes regardless of a rule match.
- **api_scrape**: The URL endpoint for the list of recent pastes.
- **api_raw**: The URL endpoint for the raw paste.
- **api_view**: The URL enpoint to view the paste.

2.4 StackExchange

The same API is used to query them all. Similar to github there is a public API which has a reduced rate limit or an App API which has a higher cap. There is a cap on 10,000 requests per day per IP, so pulling all would be impractical. Generate a key at https://stackapps.com/.

There are over 170 exchanges that form stackexchange. The following list is the most likly to expose privldidged information.

- stackoverflow
- serverfault
- superuser
- webapps
- webmasters
- dba
- **site_list**: List of site shorttitles that will be scraped.
- **api_key**: API App key as generated above.
- **store_filter**: This is the stackexchange filter that determines what fields are returned. It must contain the body element.
- **pagesize**: How many questions to pull from the latest list.
- **store_all**: Store all pastes regardless of a rule match.

CHAPTER 3

Outputs

This page details all the confiuration options for the output modules/ There are a few generic options for each input.

- **enabled**: This turns the input on and off.
- **module**: This is used internally by pastehunter.
- **classname**: This is used internally by pastehunter.

3.1 Elasticsearch

Elasticsearch was the default output. Storing all pastes and using Kibana as a graphical frontend to view the results

- **elastic_index**: The name of the index.
- **weekly_index**: Use a numbered index for each week of the year instead of a single index.
- **elastic_host**: Hostname or IP of the elasticsearch.
- **elastic_port**: Port number for elasticsearch default is 9200
- **elastic_user**: Username if using xpack / shield or basic auth.
- **elastic_pass**: Password if using xpack / shield or basic auth.
- **elastic_ssl**: True or false if Elasticsearch is served over SSL.

3.2 Splunk

Splunk output is similar to Elasticsearch. All the data is put into Splunk and then Splunk can be used for graphical frontend and querying.

- **splunk_host**: Hostname of IP of your Splunk instance.
- **splunk_port**: The Splunk management port. (Usually port 8089)

- **splunk_user**: Username of your Splunk user.
- **splunk_pass**: Password for your Splunk user.
- **splunk_index**: The name of the Splunk index to store the data in.
- **store_raw**: Include the raw paste in the data sent to Splunk.

3.3 JSON

This output module will store each paste in a json file on disk. The name of the file is the pasteid.

- **output_path**: Path on disk to store output files.
- **store_raw**: Include the raw paste in the json file. False jsut stores metadata.
- **encode_raw**: Ignored, Reserved for future usage.

3.4 CSV

The CSV output will append lines to a CSV that contains basic metadata from all paste sources. The raw paste is not included.

- **output_path**: Path on disk to store output files.

Stored elements are

- Timestamp
- Pasteid
- Yara Rules
- Scrape URL
- Pastesite

3.5 Syslog

Using the same format as the CSV output this writes paste metadata to a syslog server. The raw paste is not included.

- **host**: IP or hostname of the syslog server.
- **port**: Port number of the syslog server.

3.6 SMTP

This output will send an email to specific email addresses depending on the YaraRules that are matched. You need to set up an SMTP server.

- **smtp_host**: hostname for the SMTP server.
- **smtp_port**: Port number for the SMTP Server.
- **smtp_security**: One of `tls`, `starttls`, `none`.
- **smtp_user**: Username for SMTP Authentication.

- **smtp_pass**: Password for SMTP Authentication.
- **recipients**: Json array of recipients and rules. - **address**: Email address to send alerts to. - **rule_list**: A list of rules to alert on. Any of the rules in this list will trigger an email. - **mandatory_rule_list**: List of rules that *MUST* be present to trigger an email alert.

3.7 Slack

This output will send a Notification to a slack web hook. You need to configure the URL and the channel in Slack. Head over to https://api.slack.com/apps?new_app=1

Create a new Slack App with a Name and the workspace that you want to send alerts to. Once created under Add Features and Functionality select Incoming Webhooks and toggle the Active button to on. At the bottom of the page select *Add New Webhook to Workspace* This will show another page where you select the Channel that will receive the notifications. Once it has authorized the app you will see a new Webhook URL. This is the URL that needs to be added to the pastehunter config.

- **webhook_url**: Generated when creating a Slack App as described above.
- **rule_list**: List of rules that will generate an alert.

CHAPTER 4

PostProcess

There are a handful of post process modules that can run additional checks on the raw paste data.

There are a few generic options for each input.

- **enabled**: This turns the input on and off.
- **module**: This is used internally by pastehunter.

4.1 Email

This postprocess module extracts additional information from data that includes email addresses. It will extract counts for:

- Total Emails
- Unique Email addresses
- Unique Email domains

These 3 values are then added to the meta data for storage.

- **rule_list**: List of rules that will trigger the postprocess module.

4.2 Base64

This postprocess will attempt to decode base64 data and then apply further processing on the new file data. At the moment this module only operates when the full paste is a base64 blob, i.e. it will not extract base64 code that is embedded in other data.

- **rule_list**: List of rules that will trigger the postprocess module.

See the Sandboxes documentation for information on how to configure the sandboxes used for scanning decoded base64 data.

4.3 Entropy

This postprocess module calculates shannon entropy on the raw paste data. This can be used to help identify binary and encoded or encrytped data.

- **rule_list**: List of rules that will trigger the postprocess module.

4.4 Compress

Compresses the data using LZMA(lossless compression) if it will reduce the size. Small pastes or pastes that don't benefit from compression will not be affected by this module. Its outputs can be decompressed by base64-decoding, then using the xz command.

- **rule_list**: List of rules that will trigger the postprocess module.