## IntelMQ :

### Installation and Configuration Manual :

# Setting of testing environment:

* Install Virtual box

Install Ubuntu 22.04:

Memory: 6GB

Ram: 6

Disk: 50 GB

* Configure network to use both NAT (internet access) and Host only (to connect from the host machine) with dhcp enabled for testing
* Write a python script to automate basic security hardening refer CIS benchmark
* The VM does not pick ip address for both interfaces by default in ubuntu VM, so configure the second interface manually:
  + # cd /etc/netplan
  + # vi 00-installer-config.yaml
* Initial config:

network:

ethernets:

enp0s3: # This is your first adapter (NAT)

dhcp4: true

#Add the lines in red to configure second adapter

enp0s8: # This is your second adapter (host only)

dhcp4: true

version: 2

***\*\*\*\*\*\* Notes: since this is a yaml file, be mindful of space***

* Create a snapshot of VM and download a copy
* Now you can ssh from your host terminal and install the tools to be tested.
* *Consider hardening the OS once tools are finalized*

# Installation

There are three options for installation:

* Native: Preferred choice of installation
* Pypi : can lead to dependencies challenges
* Docker: Not stable yet, not recommended to use in production

$ sudo apt update

$ sudo apt upgrade

$ echo 'deb http://download.opensuse.org/repositories/home:/

:/intelmq/xUbuntu\_22.04/ /' | sudo tee /etc/apt/sources.list.d/home:sebix:intelmq.list

$ curl –fsSL https://download.opensuse.org/repositories/home:sebix:intelmq/xUbuntu\_22.04/Release.key | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/home\_sebix\_intelmq.gpg > /dev/null

$ sudo apt update

$ sudo apt install intelmq

1. Manager:
   1. Install
   2. Create api user
   3. Configure database

Configure Runtime configuration file based on the bots that you would like to configure for collection, parsing, deduplication, enrichment and storage:

To start with following is the design of data collection bot:

Collector ---> Parser---> Deduplicator--> enrich with taxonomy-->extract additional info for url->add geolocation info -->extract ip to fqdn-->add whois data-->write the enriched data to storage

There is a sequence of steps involved in collecting and making the threat feeds usable. The collector acquires the initial data, which is then passed on to the parser. The parser's role is to analyze and interpret the data, ensuring its accuracy and correctness. Next, the deduplicator identifies and removes any duplicate entries, streamlining the dataset. The data is then enriched with more details including, ip if only fqdn is known or vice versa and adding additional who is details. Once the data is enriched it's written to an output bot for storage, visualization or integration with existing tools.

Following bots have been chosen so far after using the feed selection criteria, more to be explored and included in future:

Collectors:

* AbusechURLHaus: bot not available, customized using generic url collector
* AbusechFeodoTracker: bot not available, customized using generic url collector
* Spamhausdrop: spamhaus collector

Parser:

* AbusechURLHaus: bot not available, customized using generic csv parser
* AbusechFeodoTracker: use AbusechFeodoTracker parser
* Spamhaus: use SpamhausDrop parser

Expert:

* Use deduplicator expert to remove deplicate entries uses hashes to detect duplicates
* Use Taxonomy expert to apply eCSIRT Taxonomy to all events for standardizing the data
* Use URL-Expert : to obtain additional information from the url for both source and destination including fqdn, ip, port, urlpath...
* Use Gethostbyname to resolve IP address for the FQDN if the IOC is an IP address.
* Use Abuseix to enrich data with abuse contact.

Output:

* Use File output bot: For the current purpose, the data is written to a file within the system.
* Logstash output bot: To write data to redi

**Configuration:**

1. **Enable redis to run on start of operating sysebstem and start the service**

# systemctl enable redis.service  
# systemctl start redis.service

1. **Note important paths for configuration management and troubleshooting**

Since we use native packages for installation the configurations are stored in standard Linux paths (LSB paths)

# /var/log/intelmq/ # log files

# /etc/intelmq/ # all core configuration files

# /var/lib/intelmq/ # data /templates storage

# /var/run/intelmq/ # list of currently configured bots

1. **Bot Configuration File:**

To configure the bots, you need to define and configure each bots in runtime.yaml in /etc/intelmq/ directory It is recommended to check configuration templates at the [Data Feeds](https://intelmq.readthedocs.io/en/latest/user/feeds.html) documentation page. Also use the following command to view the currently available bots, which you can start configuring as per your needs:

# intelmqctl list bots

* 1. The configuration for the bots for this system is recorded at [runtime configuration.txt](https://deakin365.sharepoint.com/:t:/r/sites/HardhatEnterprises2/Shared%20Documents/%F0%9F%91%BE%20Malware%20Visualisation/T3,%202023/Threat%20Intel/IntelMQ/runtime%20configuration.txt?csf=1&web=1&e=3b9i5j) for future reference and further development,

\*\*\*\*\*note it is not recommended to copy configuration files to shared drives or any unsecured storage since it can contain sensitive information including credentials.\*\*\*\*\*

1. Following is the template taken from intelMQ official documentation page :

**<bot ID>:**  
 **group:** <bot type (Collector, Parser, Expert, Output)>  
 **name:** <human-readable bot name>  
 **module:** <bot code (python module)>  
 **description:** <generic description of the bot>  
 **parameters:**  
 **<parameter 1>:** <value 1>  
 **<parameter 2>:** <value 2>  
 **<parameter 3>:** <value 3>

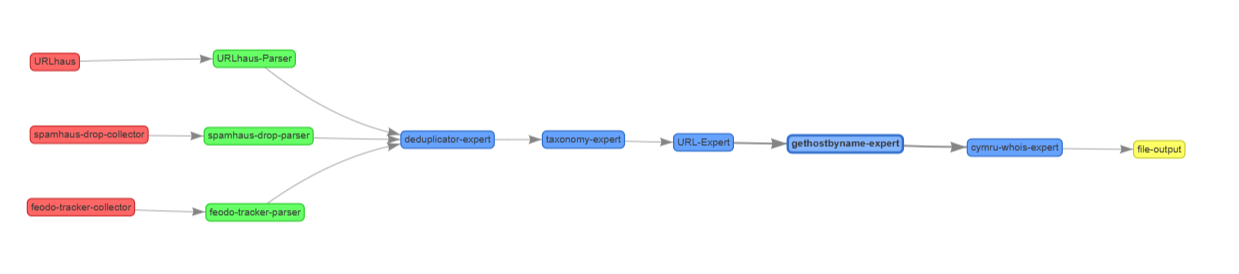
Once the bots are configured the bots, use following command to manage the bots:

\*\*\*Ensure the bot-id matches the one you assigned while configuring the bots\*\*\*

* Starting a bot:
  + intelmqctl start *bot-id*
    - Eg: # intelmqctl start URLhaus
      * ensure the id matches the id given when creating the bot
* Stopping a bot:
  + intelmqctl stop *bot-id*
    - Eg: # intelmqctl stop URLhaus

Likewise you can perform following operations when needed:

* + Reloading a bot:
    - intelmqctl reload *bot-id*
  + Restarting a bot:
    - intelmqctl restart *bot-id*
  + Get status of a bot:
    - intelmqctl status *bot-id*
* Current Bots map :



Next week 4/5:

Threat feeds of 14 different types from 10 different providers have been collected.

