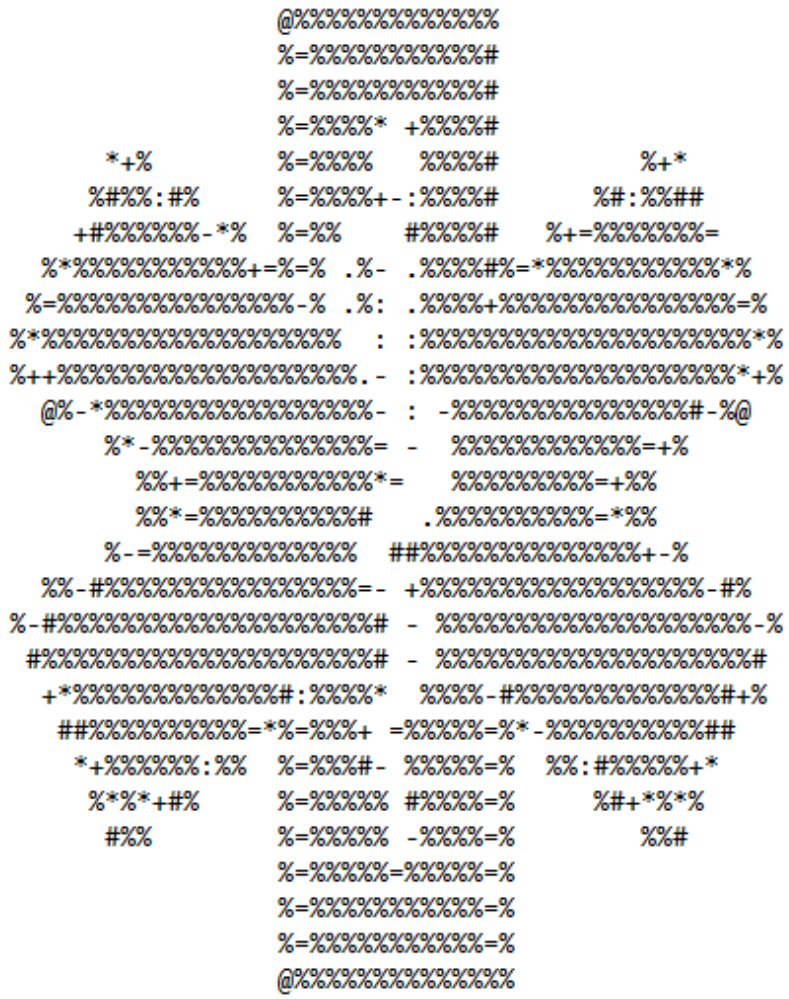
EMT User Guide

Elastic Malware Benchmark for Empowering Researchers (EMBER)
Malware Tactical Decision Aid



```
EEEEE  M     M  TTTTT
E      MM   MM    T
EEE    M M M M    T
E      M  M  M    T
EEEEE  M     M    T
```

Installation

The EMT Framework is composed of 5 parts. The Model Trainer, Model Evaluator, EMT Framework Main, VirusTotal Integrator, and Post Processing Module. All of the parts are available as Jupyter notebooks except for the EMT Framework Main which is available as a python file. These can be downloaded at https://github.com/jmeoak/EMTFramework

Jupyter:

The 4 Jupyter notebook parts can be run in any environment setup to run jupyter. In order to use the VirusTotal Integrator, a VirusTotal API key is needed. A version is included with free accounts, but is limited to 4 requests per minute and 500 per day so testing may need to be spread over multiple days if using the free version. The environment must satisfy the following dependencies:

- EMBER: https://github.com/elastic/ember (the setup script currently does not work. When installing, updated the requirements document line `lief==0.9.0` to be `lief>=0.9.0`). The other dependencies (tqdm, numpy, pandas, lightgbm, and scikit-learn) are required by ember module and most are used in the ember framework
- TensorFlow: https://www.tensorflow.org/
    - In jupyter execute `!pip install tensorflow`
- VirusTotal API: https://github.com/VirusTotal/vt-py
- Standard libraries already available in most jupyter environments: joblib, re, matplotlib

Python:

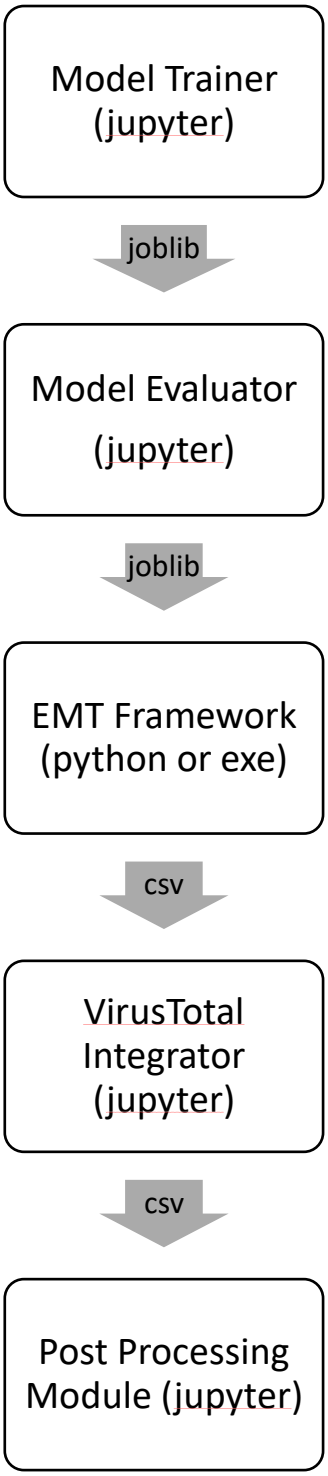
EMT Framework Main runs as a python file. Recommended usage is in a python venv for research purposes. Future releases will have a separate version available to create an all-in-one executable using only standard Linux dependencies.

The python file has the following dependencies:

- EMBER: https://github.com/elastic/ember (the setup script currently does not work. When installing, update the requirements document line `lief==0.9.0` to be `lief>=0.9.0`).
- Joblib: https://joblib.readthedocs.io/en/stable/ (Already installed with most python distributions
- Additional Models: For any models implemented, the associated library needs to be installed on the system (sklearn, tensorflow, etc.)

Usage

The full pipeline can be seen on the right. Each of the tools is designed to be very user friendly and self-explanatory.



Jupyter Notebooks:

Model Trainer:

This is the largest of the notebooks in the EMT framework. It is used to train new models for use as the machine learning engine. Examples are shown along with comments on the code. The EMBER training data must be in vectorized form. The https://github.com/elastic/ember has instructions on doing this, and it is recommended to do the vectorization in a command line environment or separate notebook as it only needs to be done once.

Model Evaluator:

This notebook uses the testing samples from the EMBER dataset to evaluate each model and output standard machine learning metrics. It is recommended to be careful of models with extremely high accuracy scores as these have been shown to be overfit to the EMBER dataset for use in modern malware detection.

VirusTotal Integrator:

This notebook takes in a csv file output by the EMT Framework Main, queries VirusTotal using the SHA256 hash and creates a new csv file including all of the original data in addition to adding the VirusTotal score corresponding to each sample. A VirusTotal API key is needed for this notebook.

Post Processing Module:

This notebook accepts either the csv file output by the EMT Framework Main or the VirusTotal Integrator and shows accuracy metrics to include normal accuracy and adjusted accuracy using predicted FPR.

EMT Framework Main:

The EMT Framework Main runs as a stand-alone python script. It has 2 run modes, normal and advanced. It uses the following folder structure by default where there is one folder at the same level as the python script containing the joblib models and another containing all of the malware samples.

- `EMT_Main.py`
- `models/`
- `malware/`

Standard Mode: Provides prompts for each option and runs a single model against 1 or more binaries with the option to save to csv. Designed for use in incident response

Advanced Mode: Shown in screenshots below. This mode allows detection of DLLs and can run different models against DLLs in addition to the ability to run multiple models at once. Designed for research use cases.

Starting Screen

Options Using Advanced Mode:

Model: The 'all' options will use every model in the model folder provided. The 4 names included in list shown will run already trained Random Forest, Decision Tree, EMBER Pretrained LightGBM, or Extremely Random Forest. Manual allows a custom name to input corresponding to a new trained model in joblib format.

Cut point: Some models provide a likelihood score between 1 and 0. This number cut point is used in the case of likelihood. 0.9 means the models needs to be 90% confident that the sample is malicious for classification as malware.

Path to malware folder: Custom path of /root/venv/Malware/pentesting/msf/stager used in the screenshot. Auto detection of PE files is done so this can be run against folders containing a mix of PE, doc, pdf, or other types of malware and will only run against PE files.

Name of malware subset: This will be the attack/group/tool family. Included as first column of csv document.

Run different models against DLLs: This is best when running multiple models as it will ask for a new DLL sub model to run with every loop of a new model. The better option is to use this with execution of a single model.

```
Use Adv Mode? (Dll Detection, Multi-Model) (y/N): y
Model Directory Path (Def: ./models):
Choose a model [rf, tree, lgbm, ET, all, manual]: all
This will run every model in the directory:
['GBC1.joblib', 'TR1.joblib', 'RF9_DLLW.joblib', 'ET1.joblib', 'RF9.joblib', 'ET1_DLL.joblib', 'ADB_RF
9.joblib', 'lgbm.joblib']
Continue? (Y/n):
Cut point for rounding prediction 0-1 (def: 0.9): 0.9
Path to Malware (Def ./malware): /root/venv/Malware/pentesting/msf/stager
/root/venv/Malware/pentesting/msf/stager
['stager_bind_named_pipe_32_exe', 'stager_reverse_tcp_32_exe', 'stager_bind_tcp_32_exe_service', 'stag
er_bind_named_pipe_64_dll', 'stager_reverse_tcp_32_exe_service', 'stager_reverse_https_32_exe', 'stage
r_reverse_http_32_exe_service', 'stager_reverse_https_32_dll', 'stager_reverse_https_32_exe_service',
'stager_reverse_https_64_dll', 'stager_reverse_http_32_dll', 'stager_bind_tcp_64_exe', 'stager_reverse
_https_64_exe_service', 'stager_reverse_tcp_64_dll', 'stager_bind_named_pipe_64_exe', 'stager_reverse_
http_64_dll', 'stager_reverse_http_64_exe_service', 'stager_bind_named_pipe_32_exe_service', 'stager_b
ind_named_pipe_64_exe_service', 'stager_reverse_http_32_exe', 'stager_bind_named_pipe_32_dll', 'stager
_reverse_https_64_exe', 'stager_bind_tcp_64_exe_service', 'stager_reverse_tcp_64_exe_service', 'stager
_bind_tcp_32_exe', 'stager_reverse_tcp_64_exe', 'stager_bind_tcp_32_dll', 'stager_bind_tcp_64_dll', 's
tager_reverse_tcp_32_dll', 'stager_reverse_http_64_exe']
Name of malware subset? (Def = dir name): Meterpreter_stagers
Meterpreter_stagers
Run Submodel Against DLLs? (y/N): N
```

Output Per Model (0 = benign, 1 = malicious)

```
Running GBC1.joblib
/root/venv/Malware/models/GBC1.joblib
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:   lief version 0.14.1-bae887e0 found instead. There may be slight inconsistencies
WARNING:   in the feature calculations.
stager_bind_named_pipe_32_exe 1
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:   lief version 0.14.1-bae887e0 found instead. There may be slight inconsistencies
WARNING:   in the feature calculations.
stager_reverse_tcp_32_exe 1
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:   lief version 0.14.1-bae887e0 found instead. There may be slight inconsistencies
WARNING:   in the feature calculations.
stager_bind_tcp_32_exe_service 1
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:   lief version 0.14.1-bae887e0 found instead. There may be slight inconsistencies
WARNING:   in the feature calculations.
stager_bind_named_pipe_64_dll 0
WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
WARNING:   lief version 0.14.1-bae887e0 found instead. There may be slight inconsistencies
WARNING:   in the feature calculations.
```

Writing to CSV File: This format will be as shown below with attack/group/tool provides, model used, SHA256 hash of file, filename, detected or not, and filetype (exe or dll). After writing to CSV the framework can be run again and if the same filename is provided, the framework will append onto it the same file to combine the results of multiple runs.

```
Write to csv? (Y/n):
csv name (def: malware subset name): NewMalware
Run Again? (Y/n)
```

	A	B	C	D	E	F
1	Attack/Group	Model	SHA256 Hash	File Name	Detected	FileType
2	stageless	GBC1.joblib	8b7eaa551a15174caed8d44cb9bf9e3773350d039125cd238c21439e83437d8e	meterpreter_bind_named_pipe_32_exe	1	exe
3	stageless	GBC1.joblib	7137de0974395e5d135ff2f039e084dc73c778650ccb351ea7fa6fab7a06273a	meterpreter_bind_tcp_32_exe	1	exe
4	stageless	GBC1.joblib	59e39e8802cd1a4baf6f03df6795a6e7230c6cc7a951e55d1dd3aada6ddf8e61	meterpreter_bind_named_pipe_64_exe	1	exe
5	stageless	GBC1.joblib	add2a70519e48e6bbc291115a6addc0cc63bd6dd2bf89ec6b2a80d4e45e9ba84	meterpreter_reverse_tcp_32_exe	1	exe