Towards a unified XAI-based framework for digital forensic investigations

By:
Zainab Khalid, Farkhund Iqbal, Benjamin C.M. Fung

From the proceedings of
The Digital Forensic Research Conference
DFRWS APAC 2024
Oct 22-24, 2024

Forensic Science International: Digital Investigation 50 (2024) 301806

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS APAC 2024 - Selected Papers from the 4th Annual Digital Forensics Research Conference APAC

Towards a unified XAI-based framework for digital forensic investigations

Zainab Khalid [a,*], Farkhund Iqbal [b], Benjamin C.M. Fung [c]

[a] National University of Sciences and Technology (NUST), School of Electrical Engineering and Computer Science (SEECS), Islamabad, Pakistan
[b] College of Technological Innovation, Zayed University, Dubai, United Arab Emirates
[c] School of Information Studies, McGill University, Canada

ARTICLE INFO

Keywords:
Explainable artificial intelligence
XAI
Digital forensics
XAI-DF framework
UNSW-NB15 dataset
LIME
SHAP

ABSTRACT

Explainable Artificial Intelligence (XAI) aims to alleviate the black-box AI conundrum in the field of Digital Forensics (DF) (and others) by providing layman-interpretable explanations to predictions made by AI models. It also handles the increasing volumes of forensic images that are impossible to investigate via manual methods; or even automated forensic tools. A holistic, generalized, yet exhaustive framework detailing the workflow of XAI for DF is proposed for standardization. A case study examining the implementation of the framework in a network forensics investigative scenario is presented for demonstration. In addition, the XAI-DF project lays the basis for a collaborative effort from the forensics community, aimed at creating an open-source forensic database that may be employed to train AI models for the digital forensics domain. As an onset contribution to the project, we create a memory forensics database of 27 memory dumps (Windows 7, 10, and 11) simulating malware activity and extracting relevant features (specific to processes, injected code, network connections, API hooks, and process privileges) that may be used for training, testing, and validating AI models in keeping with the XAI-DF framework.

1. Introduction

The adoption of Artificial Intelligence (AI) in the mainstream has grown exponentially in recent years mainly because it can solve complex problems, process vast amounts of data, and perform tasks previously thought to be exclusive to human intelligence only. AI's prevalence is also credited to the fact that the technology '*feeds on itself*' in progression Yampolskiy and S (2016)]. Everyday businesses, industries, and critical sectors such as healthcare, finance, defense and cybersecurity, etc. make use of AI to achieve efficiency in their workflows in various manners Baggili and Behzadan (2019). Consequently, AI's utilization in high-stakes situations, such as those involved in cybersecurity and Digital Forensics (DF) for justice courts, raises questions about the assurance, reliability, and validity of its performance and results. Since the model's decisions or predictions directly impact individual and collective human lives, it becomes crucial to develop trust in AI models through interpretability i.e. through Explainable AI (XAI).[1] This is especially an important consideration when the subject model is *closed-box* or *black-box*.[2]

Failures in AI systems have been documented on many accounts; inevitably, because all machines/codes have bugs or loopholes Yampolskiy and S (2016). AI failures are specifically attributed to algorithmic biases which are more closely related to the training data rather than the technical details of data processing Solanke et al. (2022). Inadequately trained AI models may generate predictions influenced by unintended features present in the training dataset Hall et al. (2022). Examples of accidents caused by AI software or robots are numerous, such as robotic financial advisors giving bad advice to intelligent AI stock trading software causing trillion-dollar crashes Yampolskiy and S (2016). In 2015, at a Volkswagen plant, a robot that was programmed to work with automobile parts seized and crushed a worker against a metal plate, which resulted in him being killed Yampolskiy and S (2016); Docterman (2015). Likewise, multiple road accidents involving self-driving cars like Tesla have been reported as well Yampolskiy and S (2016); Levin and Woolf (2017).

AI failures may also impact DF processes. For example, a computer vision system intended to categorize images of tanks but instead learned to differentiate the backgrounds of these images Yampolskiy and S

* Corresponding author.
E-mail address: zkhalid.msis18seecs@seecs.edu.pk (Z. Khalid).

[1] In the context of XAI, explainability and interpretability are used interchangeably throughout the text of this paper.

[2] A black box AI model's internal processes are opaque and not easily interpretable, making it difficult to understand how it arrives at its decisions/predictions.

https://doi.org/10.1016/j.fsidi.2024.301806

Available online 18 October 2024

(2016); Yudkowsky (2008).

With this premise that AI systems, like all machines are bound to fail at one point or another, in the context of DF, we need to be able to prove that predictions made by AI systems for digital evidence are up to legal standards i.e. verifiable, and based on an *admissible* timeline of events. The end goal is to use the evidence as expert evidence in court. While AI does make increasing volumes of data/forensic images more manageable, the interpretability requirements of AI models hold prime significance. To this end, XAI helps demonstrate a model's '*impartiality in decision-making*', by identifying how the prediction was made and if the subject features are relevant or suitable for contributing to predictions while also determining unknown biases in training datasets Hall et al. (2022). It is pertinent to note that explanations given by XAI should be easy to understand by a layman at the very least (while *truly explainable* systems would be an idealistic case) Arrieta et al. (2020).

XAI implementations and tools for DF must have the flexibility to be contextualized to multiple forensic scenarios and pertinent data under investigations that may involve multiple file formats, OSs, etc. from multiple sources like disk, memory, and network after aggregation Hall et al. (2022). It is also important to utilize the right model for the right task, e.g. intrinsically explainable (or self-explainable) AI models like Decision Trees (DT) may especially be used for well-structured forensic data. On the other hand, relevant interpretable models (or post-hoc explainable models) may be used for unstructured data like image/-audio/video Solanke et al. (2022). Also, multiple methods of explanation can be employed as well, such as *local* vs. *global*[3] explanations Alam and Altiparmak (2024).

Considerable research is being done in the XAI-DF domain in particular (discussed further in Section "2"). However, a standard framework, that generalizes yet details a workflow through set modules that can be applied to various DF sub-domains exhaustively, still needs to be outlined. In this context, our research study aims to propose and implement an XAI framework for DF and also propagate collaborative research efforts in the domain. Three major contributions of this study are as follows.

- A holistic and general XAI-DF framework, that is comprehensive, adaptable, and explainable is proposed.
- The XAI-DF collaborative project is initiated with a memory forensics database of 27 dumps (simulating malware activity) for XAI in the DF domain. In addition, we extract process-centric memory features from the dumps for *explained* classification. The project aims to build a vast database that may facilitate research and development in XAI-DF.
- A case study implementing the practical workflow of the XAI-DF framework (utilizing the UNSW-NB15 network database) is presented.

The rest of this paper is structured as follows. Section II discusses previous research and other related contributions. Section III details the proposed XAI-DF framework. Section IV explains the XAI-DF project. Section V presents an implementation of the framework using a case study utilizing the Network Intrusion Detection Systems (NIDS) database. Section VI discusses the final comments, conclusion, and possible future directions in the domain.

2. Related work

Hall et al. present a proof of concept implementation of XAI in IT forensics Hall et al. (2022). Using a database of 23 VHD forensic images which are sourced to extract multimedia (images and videos) and file metadata to be input into a training model, the classification results are processed via LIME for explanations. The classifications for multimedia were based on a 16-digit hex code embedded into the target images and videos. The LIME explanations for the results of image classification in specific divulged that the model was making predictions of target multimedia based on features other than the hex codes. This analysis reinforces the fact that explainability needs to be a standard module in forensic investigations that utilize AI models to avoid coming to conclusions based on faulty inferences. Also, the training sets need to be considerably large in order to sufficiently train the models.

Hall et al. discuss current AI solutions integrated into digital forensics tools that mainly assist in multimedia forensics Hall et al. (2021). For example, Griffeye[4] is a tool that uses AI to classify images. Currently, such AI-integrated forensic tools are completely opaque and offer no explanations as to how they perform classifications and predictions. A *human-in-the-loop* must validate the results to be acceptable as verified outputs.

Solanke discusses the limitations of closed-box AI models and explores methods for making AI-based digital forensics investigations more interpretable given that courts, legal practitioners, and the general public are skeptical about using AI for digital evidence extraction due to concerns about transparency and understandability Solanke et al. (2022). Inaccurate interpretations are said to be likely caused by "*erroneous algorithms/code, skewed or disproportionate datasets, and defective functional components of the system* (e.g., *OS, distributed platforms, etc.)*" Solanke et al. (2022).

Dunsin et al. propose the MADIK framework, which can be referenced to highlight the proposition that multiple AI agents can be used for *specific* forensics purposes, i.e., an AI algorithm can be trained to analyze just the Windows Registry, others can be trained for file/directory paths' analysis, timestamp analysis, etc. Dunsin et al. (2022). AI models may perform more efficiently when trained and tested for such specific tasks. All agents' findings may be combined to produce corroborative results and predictions finally.

Kalutharage et al. make *antemortem* utilization of XAI as opposed to validity and assurance in postmortem forensic law i.e. to help detect DDoS attacks as part of intrusion detection Kalutharage et al. (2023). They determine influential features from (local and global) explanations of individual anomalous instances and correlate them with a list of the most informative DDoS attack detection features. This streamlined the most important DDoS attack features enabling more efficient detection than Deep Neural Network (DNN), Random Forest (RF), and DT.

3. The Explainable Artificial Intelligence for Digital Forensics (XAI-DF) framework

The integration of XAI into DF is meant to prioritize the interpretability needs of critical forensics contexts; since potential AI failures in DF, in a worst-case scenario, may lead to inaccurate verdicts in court that can gravely impact human lives. The proposed XAI-DF framework generalizes the holistic workflow of a digital forensics investigation employing XAI for predictions and interpretable explanations, yet offering an exhaustive/comprehensive and adaptable structure that may be used for: (a) any digital forensics domain, (b) utilizing any suitable existing AI model (or designing custom models for specialized use), and (c) finally sourcing any explainability method for interpretability.

Fig. 1 illustrates the abstract/high-level XAI-DF framework, while Fig. 2 details the framework ontology in-depth. It is composed of three main modules (or phases): (1) Forensic Data Collection, (2) Artificial Intelligence Model, and (3) Explainable AI. Before reaching any conclusions, it is an efficient practice that a *human-in-the-loop* cross-checks and verifies the AI's decisions at each stage in the XAI-DF framework's processes. As with any DF investigation, results obtained via the

[3] Local explanations interpret decisions for one input or instance in a dataset, while global explanations provide information about all inputs as a whole Alam and Altiparmak (2024).

[4] https://www.griffeye.com/.

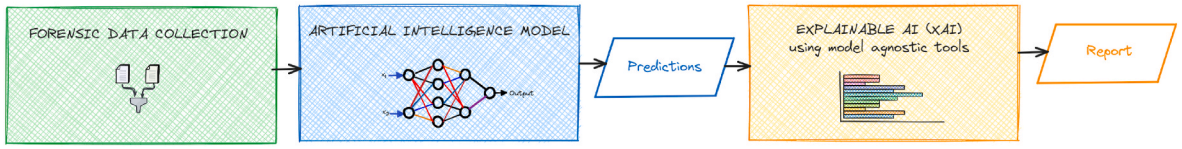

Fig. 1. XAI-DF holistic framework.

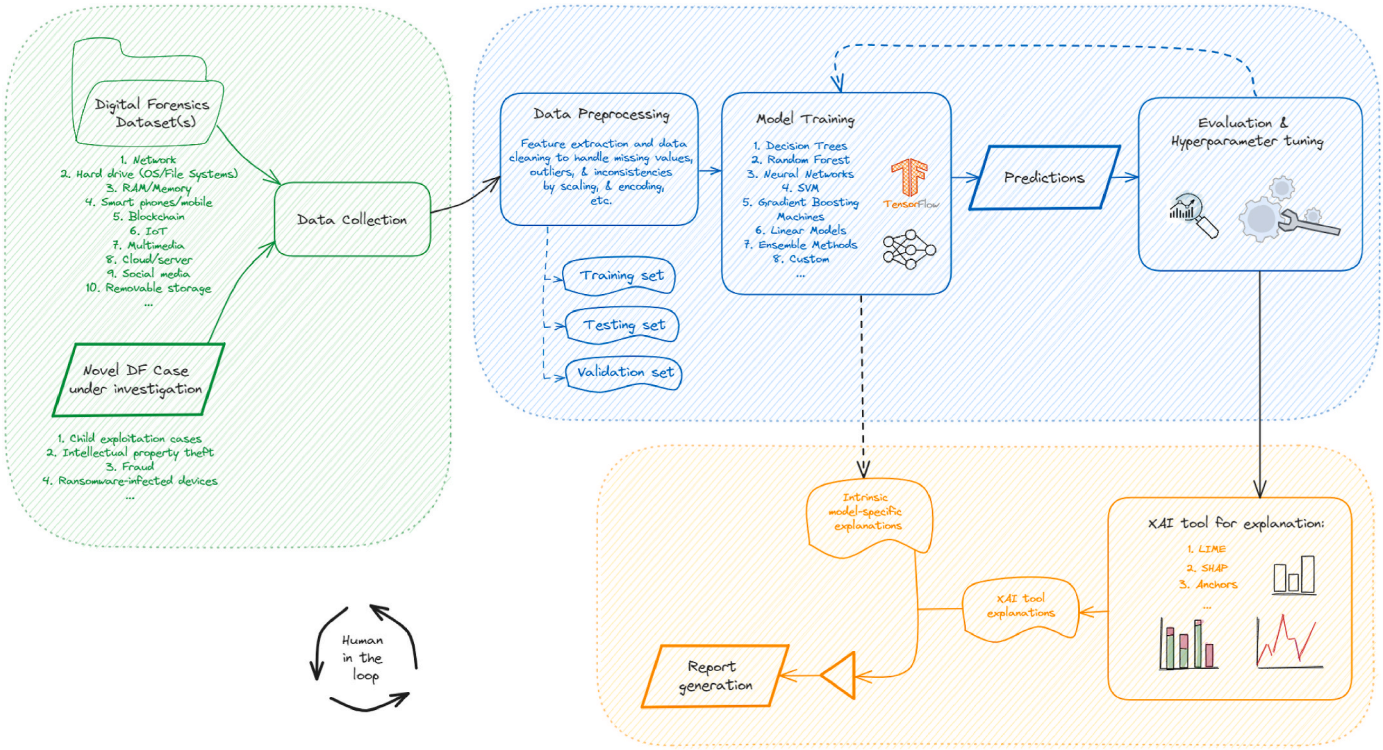

Fig. 2. XAI-DF framework ontology.

XAI-DF framework must be done so following the established chain-of-custody protocols, and pertinent legal and ethical considerations to ensure the admissibility of the extracted artifacts as expert evidence.

3.1. Forensic Data Collection

The data collection module sources dataset(s)/database(s) of a digital forensics sub-domain like (1) network, (2) hard drive (Operating System (OS)/file system), (3) RAM/memory, (4) mobiles/smartphones, (5) Internet of Things (IoT), (6) blockchain, (7) cloud/server, (8) social media, (9) multimedia (images, audio, video), (10) removable storage, etc. This includes a pre-prepared dataset of forensic material (such as memory dumps, network traffic captures, or hard drive forensic images, etc. depending on the sub-domain) that is used for training AI models.

In addition, case material of a novel digital forensics investigation at hand (such as child exploitation cases, intellectual property theft, fraud or ransomware investigations, etc.) may be input for both training and/or testing. This ensures real-time postmortem DF analysis capabilities are included in the framework. The pertinent data is used to extract meaningful *features* in the feature extraction step of the next module.

Forensic tools may be used to extract and aggregate information from databases/case material that contain forensic images in raw form i.e. bytes. This is done to convert data into a more readable form before feeding it to an AI model. For example, Volatility may be used to parse the memory for running processes and other registry or network artifacts, etc. Autopsy may be used to view and extract different OS or user files from *bit-by-bit* hard drive forensic images, etc. and Wireshark or NetworkMiner may be used to analyze traffic captured from networking hardware.

3.2. Artificial Intelligence Model

The AI model requires data to be preprocessed before training to handle discrepancies in the datasets. *Data cleaning* caters for missing/null values, outliers, and inconsistencies. The *feature extraction* step captures relevant information from the raw data (or after it has been processed via forensic tools) and represents it in a form that is more suitable for learning by AI models. Then, *transformation* encodes categorical features, scales numerical ones, and handles text data preprocessing. The data is finally split into *testing*, *training*, and *validation* sets after preprocessing.

Following preprocessing, a fitting AI model (like Decision Tree, Random Forest, Neural Networks, Support Vector Machines, Gradient Boosting Machines, Linear Models, Ensemble Methods, or any customized/designed model) that is *suitable* and *compatible* with the subject dataset is identified. In terms of DF, the AI models may help perform (1) network traffic analysis, (2) event/timeline reconstruction through file system analysis, (3) registry analysis, (4) log analysis, (5) database analysis, (6) browser and cloud/server analysis (7) classification of malicious memory processes, (8) multimedia analysis, (9) text analysis, etc.

The training set is used to train the model which then gives predictions on test data. The performance of the trained models is evaluated based on metrics such as accuracy, precision, recall, F1-score, Mean

Absolute Error (MAE), Mean Squared Error (MSE), etc. and the validation set may be utilized to tune hyperparameters.

It is pertinent to note that some AI models are *intrinsically* explainable, in that they may provide *model-specific* explanations of how they came to certain conclusions. Such model-specific *glass-box* explanations (e.g., from classic Machine Learning (ML) models like Decision Trees, rule-based, linear models, etc.) may later be compared or combined with the *model-agnostic*[5] explanations obtained from XAI tools in the next module.

Usually, multiple AI models are used for a dataset to determine which works best in terms of efficiency, performance, etc.

3.3. Explainable Artificial Intelligence

The Explainable AI module entails obtaining explanations for the black-box model's predictions using external tools (i.e. model-agnostic explanations). Explanations may be local/global, textual, or visualization-based, etc. As previously mentioned, these explanations may be corroborated with intrinsic/model-specific explanations for more clarity, if the AI model under use is intrinsically explainable.

XAI tools create explanations for predictions through various methods, like model-agnostic approaches that perturb the input data and fit a simple, interpretable model locally to approximate the complex model's behavior, highlighting feature importance. Local Interpretable Model-agnostic Explanations (LIME) is one such tool that may be used to explain single instances (or a subset of instances) in datasets; a *local* explanation. Tools like SHapely Additive exPlanations (SHAP), Saliency map, Counterfactual, etc. also produce local explanations. SHAP, however, uses game theory to assign each feature an importance value, explaining the contribution of each feature to the prediction. Some of the *global* explanation algorithms are Partial Dependence Plot (PDP), Individual Conditional Expectation (ICE), Global Sensitivity Analysis (GSA), and Submodular Pick LIME (SP-LIME) Alam and Altiparmak (2024). Anchors and LORE are post-hoc as well Alam and Altiparmak (2024).

The implementation of the XAI-DF framework in different DF scenarios such as memory forensics and network forensics are presented in the following sections to demonstrate its practical application.

4. A memory forensics database for the XAI-DF project

The XAI-DF project is initiated as an open-source resource of digital forensic images to be utilized for (but not limited to) training XAI models used in DF investigations. For this purpose, we created an initial memory database of 27 dumps,[6] detailed below, that focuses on malware activity in the memory. With the progression of time, this is intended to be used as a base to build upon a more vast database consisting of forensic images of both memory and other sub-disciplines such as network, hard drive/disk space, smartphones, IoT, multimedia, blockchain, etc.

Since the XAI-DF project is a collaborative effort, members of the forensics community are urged to contribute with forensic images (of all types) by uploading them to the project. This will help achieve its intended purpose of providing researchers and practitioners with a vast database of DF images for XAI. Forensics research, in general, may also greatly benefit from such a database in many ways.

4.1. Memory database creation

Virtual Machines (VMs), operated via a controlled VMware Workstation Pro environment, and created with various Windows OSs' *.iso* images, i.e. Windows 7 Professional, Windows 10 Home, and Windows 11 Home, were allotted 2 GB RAM, and 60 GB disk space. Table 1 logs the experimental environment details.

Table 1
Environment—memory database creation.

Characteristic	Description
Virtualization Software	VMware Workstation 16 Pro, 16.2.5 build-20904516
Windows 7	Windows 7 Professional, Service Pack 1, 32-bit OS
Windows 10	Windows 10 Home, 19042.631, 64-bit OS
Windows 11	Windows 11 Home, 64-bit OS
Memory acquisition	AccessData FTK Imager 4.5.0.3
Feature extraction	Volatility (versions 2.6 and 3)
Classification	Weka 3.8.6 (DT, RF, Naive Bayes), TensorFlow (DT, RF)
XAI-DF	LIME, SHAP

VMs were used as testbeds to simulate malicious activity. Since malware can infect a machine through various methods, a random combination of activities was conducted for each VM to achieve an infected machine like careless online surfing (visiting suspicious websites, clicking questionable pop-ups, downloading ambiguous games), or directly downloading and executing malware samples from resources such as Zeltser (2021) and various GitHub repositories like Malware2.0Database,[7] and malware-samples[8] etc.

Raw memory images, each 2 GB in size, were taken using the AccessData FTK Imager by suspending the VMs and creating duplicates of the *.vmem* file pertinent to each VM. In addition to memory dumps of malicious activity, some benign memory dumps consisting of normal user activity/benign running processes were also captured for each OS. Table 2 logs the characteristics of the memory database in detail.

4.2. Feature extraction

Memory features were extracted from raw memory dumps of Windows 7 and 10, in particular, using the Volatility Framework 2.6 and 3 Volatility (2024). Note that since current tools do not support Windows 11 analysis, it is omitted from the feature extraction stage for now. Information extracted from outputs of various Volatility plugins (specified below for each feature) was largely done manually which contributed to the curation of process-centric memory features. Some network connections, API hooks, injected code, and process privilege features were also extracted. A total of 55 features (with numerical and categorical values) are presently extracted from the database. To label the dataset, each process in the memory dump was individually extracted via the Procdump Volatility plugin and scanned on VirusTotal.[9]

The memory features can be accessed via the GitHub repository of the project[10]. *Memory_Features_Separate.xlsx* logs features of each memory dump separately while *Memory_Features_Combined_CSV.csv* logs the same features combined altogether. The names with descriptions of the extracted features are elaborated in detail below.

- `OSVersion_Win7SP1x86`: Indicates (by 1/0) if the OS version is Win7SP1x86
- `OSVersion_Win10Homex64`: Indicates (by 1/0) if the OS version is Win10Homex64
- `Process_Name`: Name of the running process as seen in memory (via Pslist, Psscan, and Psxview plugins)
- `PID`: Process ID (via Pslist plugin)
- `PPID`: Parent Process ID (via Pslist plugin)
- `Hidden_Process`: Indicates (by 1/0) whether or not the subject process was hidden in memory (via Psscan and/or Psxview plugins)

[5] Model-agnostic explanations of AI predictions are obtained without relying on their internal structures, and applicable to any model type.

[6] Memory database can be accessed at the link provided in the GitHub repository of the project: https://github.com/znbkhld/XAI-DF-Project.

[7] https://github.com/pankoza2-pl/Malware2.0Database.

[8] https://github.com/fabrimagic72/malware-samples.

[9] https://www.virustotal.com/gui/home/upload.

[10] https://github.com/znbkhld/XAI-DF-Project.

Table 2
Memory database characteristics.

Characteristic	Description
Size	**54 GB** raw bytes captured in (*.vmem*) memory dump files
Memory dumps per OS	**Windows 7 SP1**: 12 memory dumps **Windows 10 Home**: 8 memory dumps **Windows 11 Home**: 7 memory dumps
Memory attack type	Malware activity
Features	**55** (Process-centric features, Injected code features, API hooks features, Network connections features, Process privileges features)

- `Threads`: Number of open threads (via Pslist plugin)
- `Handles`: Number of open handles (via Pslist plugin)
- `DLLs`: Number of DLLs (via DLLlist plugin)
- `Session_ID`: Session ID (via Pslist plugin)
- `Wow64`: Indicates (by 1/0) whether or not the process is a Wow64 process (i.e. it uses a 32-bit address space on a 64-bit kernel) (via Pslist plugin)
- `Start_Time`: Process' start time (via Pslist plugin)
- `Exit_Time`: Process' exit time (in case closed) (via Pslist and Psscan plugins)
- `Injected_Code`: Indicates (by 1/0) whether or not the process contains injected code (via Malfind plugin)
- `APIhooks Features`: Indicates the number of API hooks of subject type (via APIhooks plugin)
  - `APIhooks_ImportAddressTable(IAT)`
  - `APIhooks_Inline/Trampoline`
  - `APIhooks_NTSyscall`
- `Network_Connection`: Indicates (by 1/0) whether or not the subject process established a network connection (via Netscan plugin)
- `Network_Protocol_TCP`: Indicates (by 1/0) whether or not the subject process communicated via TCP protocol (via Netscan plugin)
- `Network_Protocol_UDP`: Indicates (by 1/0) whether or not the subject process communicated via UDP protocol (via Netscan plugin)
- `Process Privileges Features`: Indicates (by 1/0) whether or not the subject process had the specified privilege (description of each privilege can be referenced from *Memory_Features_Separate.xlsx*)
  - `CreateTokenPrivilege`
  - `AssignPrimaryTokenPrivilege`
  - `LockMemoryPrivilege`
  - `IncreaseQuotaPrivilege`
  - `MachineAccountPrivilege`
  - `TcbPrivilege`
  - `SecurityPrivilege`
  - `TakeOwnershipPrivilege`
  - `TakeOwnershipPrivilege`
  - `LoadDriverPrivilege`
  - `SystemProfilePrivilege`
  - `SystemtimePrivilege`
  - `ProfileSingleProcessPrivilege`
  - `IncreaseBasePriorityPrivilege`
  - `CreatePagefilePrivilege`
  - `CreatePermanentPrivilege`
  - `BackupPrivilege`
  - `RestorePrivilege`
  - `ShutdownPrivilege`
  - `DebugPrivilege`
  - `AuditPrivilege`
  - `SystemEnvironmentPrivilege`
  - `ChangeNotifyPrivilege`
  - `RemoteShutdownPrivilege`
  - `UndockPrivilege`
  - `SyncAgentPrivilege`
  - `EnableDelegationPrivilege`
  - `ManageVolumePrivilege`
  - `ImpersonatePrivilege`
  - `CreateGlobalPrivilege`
  - `TrustedCredManAccessPrivilege`
  - `RelabelPrivilege`
  - `IncreaseWorkingSetPrivilege`
  - `TimeZonePrivilege`
  - `CreateSymbolicLinkPrivilege`
  - `DelegateSessionUserImpersonatePrivilege`
- `Label`: Malicious vs. Benign

4.3. Classification results

Classification of the memory database (*Memory_Features_Combined_CSV.csv*) using Weka's DT (J48), DT (LMT), DT (Hoeffding), Random Forest, and Naive Bayes gave accuracy scores of 93.75 %, 94.55 %, 92.16 %, 95.35 %, and 91.07 %, respectively.

The classification was also done using Python's TensorFlow library; DT, RF, and DNN models were used. In addition, other libraries were used including Pandas for data manipulation, NumPy for numerical computations, Matplotlib for visualization, and scikit-learn modules for preprocessing, modeling, and evaluation. *Memory_Features_Combined_CSV.csv* was loaded using Pandas, followed by preprocessing steps which included separating features and label, and catering for categorical and numerical features. After splitting the dataset into testing and training sets, DT, RF, and DNN models were defined and trained using scikit-learn and used for classification. Accuracy scores for DT, RF, and DNN models were 93.11 %, 95.28 %, and 93.47 % respectively. The implementations of all three models are available via GitHub.

4.4. LIME and SHAP explanations for interpretability

Subsequently, LIME was used to generate local explanations for the models' predictions. This involved initializing a LIME tabular explainer object, randomly choosing an instance from the test set to explain, and using LIME to explain the model's prediction for that particular instance, i.e., plotting feature importances. Fig. 3 illustrates such an explanation in a bar plot form. The same explanation can be obtained as a graph format (Fig. 4). From the graph plot, the sample's prediction probability indicates it is malicious, and top contributing features are `APIhooks_ImportAddressTable` with 16 % feature importance score, `Threads` with 15 %, while `TrustedCredManAccessPrivilege`, `Wow64`, `RestorePrivilege`, and `Network_Protocol_TCP` features with 4 % importance scores. Other contributing features include `DelegateSessionUserImpersonatePrivilege`, `PID`, `Network_Protocol_UDP`, `IncreaseWorkingSetPrivilege`, `SecurityPrivilege`, `Hidden_Process`, etc.

As opposed to explanations of single instances that LIME produces, SHAP provides global explanations as well. Fig. 5 illustrates features (in order of importance) with the most impact on the classification across *all* instances. For the memory database in its current stage, top features include `Handles`, `PPID`, `Threads`, `PID`, and `APIhooks_ImportAddressTable`, etc.

5. Case study: implementation of XAI-DF framework in a network forensics scenario

As another implementation of the XAI-DF framework, we detail our experiments conducted for the classification of various network attacks of the UNSW-NB15 dataset using DT, RF, and DNN models and interpreting the predictions using LIME model-agnostic XAI tool Moustafa and Slay (2015a,b); Ribeiro et al. (2016).

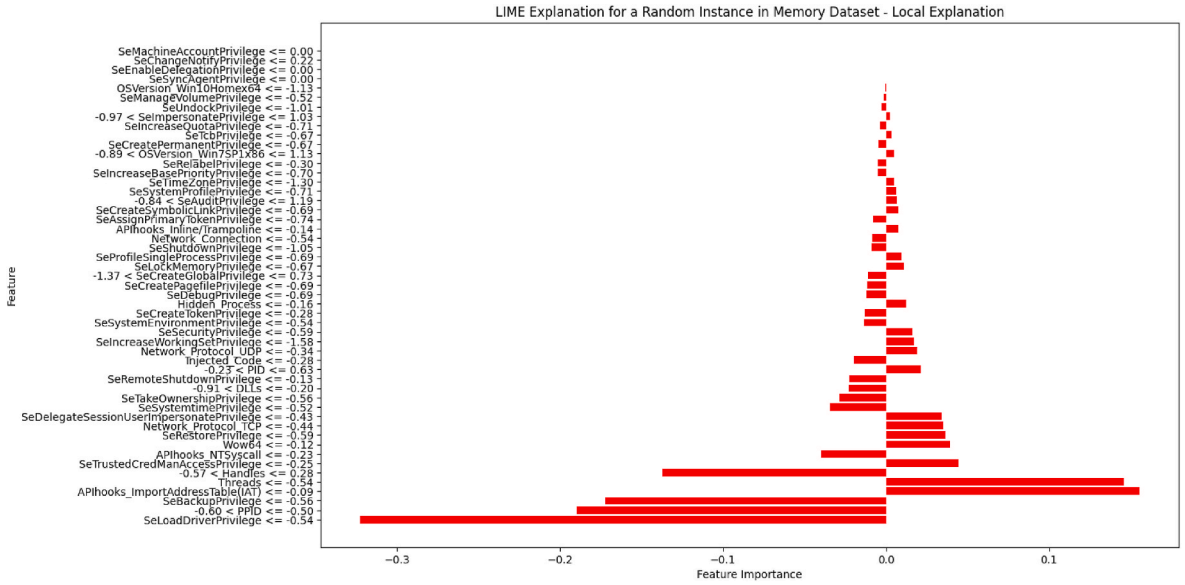

Fig. 3. LIME local explanation (bar plot)—features' importance for a random instance from memory dataset, DT implementation.

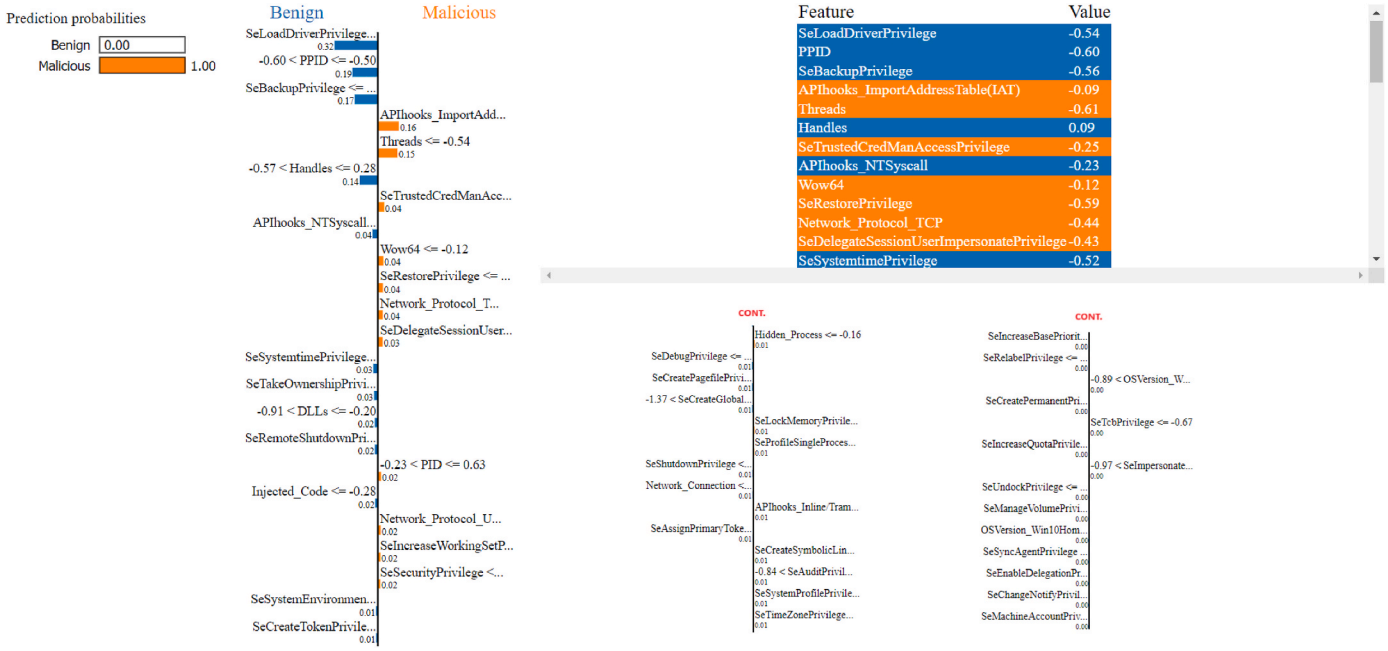

Fig. 4. LIME local explanation (graph plot)—features' importance for a random instance from memory dataset, DT implementation.

5.1. *UNSW-NB15 database*

The UNSW-NB15 dataset is a widely used benchmark dataset in the field of network security Moustafa and Slay (2015a,b). Developed by researchers at the University of New South Wales (UNSW) in Australia, the dataset contains network traffic data generated using the IXIA PerfectStorm tool in a controlled lab environment, simulating various types of low-footprint network attacks and normal network activities.

The dataset contains 9 different network attack families and includes 49 features extracted from network packets, such as source and destination IP addresses, port numbers, transaction protocols, transaction bytes, etc. Moustafa and Slay (2015a,b). The dataset comprises a total of 2,540,044 records across four CSV files (*UNSW-NB15-[1–4].csv*). From these records, a subset is dedicated for training and testing purposes: *UNSW_NB15_training-set.csv* and *UNSW_NB15_testing-set.csv* containing 175,341 and 82,332 records, respectively. These records encompass different types of network activities, including both normal traffic and various forms of attacks.

The UNSW-NB15 dataset is often used for evaluating and testing NIDSs. Its diverse range of attack scenarios makes it valuable for training and validating ML models for detecting network intrusions and anomalies. For our implementation of the XAI-DF framework, we use the dataset in the context of a cybercrime forensic investigation, aiming to perform binary classification (to determine normal and attack traffic) and multiclass classification (to determine the various attack families). Table 3 details the characteristics of the UNSW-NB15 dataset.

5.2. *Artificial Intelligence Model(s) for classification*

DT, RF, and DNN models were used to perform binary and multiclass

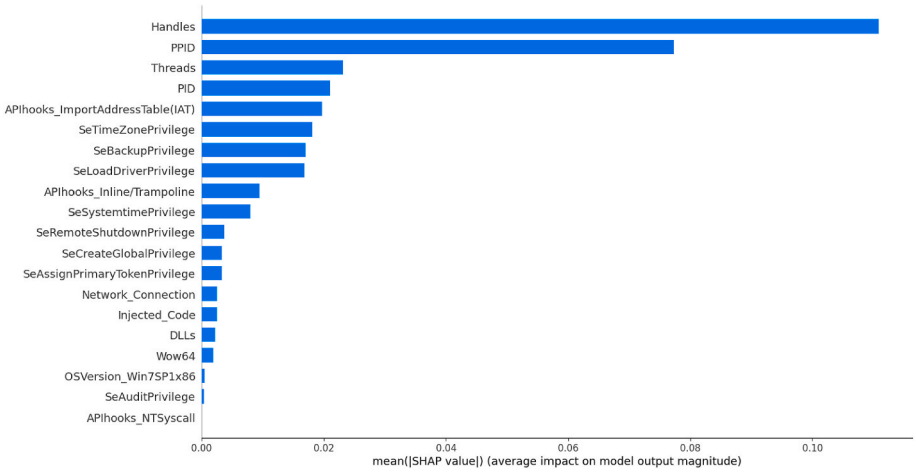

Fig. 5. SHAP global explanation—average impact of each feature on DT model's output.

Table 3
UNSW-NB15 dataset.

Characteristic	Description
Size	**100 GB** raw network data captured in *.pcap* files
Network attack types	**9** (DoS: 0, Fuzzers: 1, Generic: 2, Exploits: 3, Reconnaissance: 4, Analysis: 5, Shellcode: 6, Worms: 7, Backdoor: 8, Normal: 9)
Features	**49** (Flow features, Basic features, Content features, Time features, Additional generated features, Labelled features)
Total no. of records	**2,540,044** (4 CSV files)
Training set	**175,341** records
Testing set	**82,332** records

classifications of the dataset using Python and TensorFlow. The '*label*' feature in the dataset (which had 2 outcomes: 0 for normal traffic, 1 for abnormal traffic) was used as the target label for binary classification. The '*attack_cat*' feature (with 9 possible outcomes representing the 9 attack families specified in Table 3) was used as the target label for multiclass classification.

Loading the training and testing sets' CSV files using Pandas, preprocessing steps included combining the datasets, separating features and target labels, encoding categorical targets into numerical labels, and preprocessing categorical and numerical columns separately. The dataset was then split into training and testing sets. A DT classifier was then defined and trained which then made predictions on the testing set and performance was evaluated using accuracy score and classification report metrics (precision, recall, f-score, etc). The accuracy for DT binary and multiclass classifications was 98.4 % and 85.1 %, respectively. While accuracy for RF binary and multiclass classifications was 97.6 % and 85.25 %, respectively. Similarly, multiclass classification accuracy for DNN was 81 %. Classification reports detailing precision, recall, and f1-scores for DT and RF multiclass implementations are shown in Figs. 6 and 7, respectively. Note that 0–9 identifiers in the Figures represent 9 attack categories plus normal traffic (detailed mapping of identifiers to attack categories used in implementations can be referenced from Table 3). The DT, RF, and DNN implementations are available via GitHub.

5.3. LIME explanations for interpretability

LIME was used to generate a local explanation for a random instance (Fig. 8). Fig. 9 illustrates a more specific explanation that details feature importances for two attack categories, i.e., 4 (Reconnaissance) and 9 (Normal).

```
Making predictions
Accuracy: 0.8511691083729505

Classification Report:
              precision    recall  f1-score   support

           0       0.37      0.40      0.38      3292
           1       0.82      0.81      0.81      4831
           2       0.99      0.98      0.99     11537
           3       0.73      0.72      0.73      9080
           4       0.76      0.75      0.76      2852
           5       0.21      0.22      0.21       519
           6       0.59      0.60      0.60       309
           7       0.61      0.59      0.60        34
           8       0.16      0.18      0.17       468
           9       0.98      0.98      0.98     18613

    accuracy                           0.85     51535
   macro avg       0.62      0.62      0.62     51535
weighted avg       0.85      0.85      0.85     51535
```

Fig. 6. Accuracy and classification report of Decision Tree multiclass implementation (clipped CMD output).

```
Accuracy: 0.8525856214223343


Classification Report:
              precision    recall  f1-score   support

           0       0.39      0.37      0.38      3292
           1       0.80      0.78      0.79      4831
           2       0.99      0.98      0.99     11537
           3       0.74      0.78      0.76      9080
           4       0.79      0.76      0.77      2852
           5       0.19      0.17      0.18       519
           6       0.68      0.60      0.64       309
           7       0.67      0.18      0.28        34
           8       0.11      0.11      0.11       468
           9       0.96      0.97      0.97     18613

    accuracy                           0.85     51535
   macro avg       0.63      0.57      0.59     51535
weighted avg       0.85      0.85      0.85     51535
```

Fig. 7. Accuracy and classification report of Random Forest multiclass implementation (clipped CMD output).

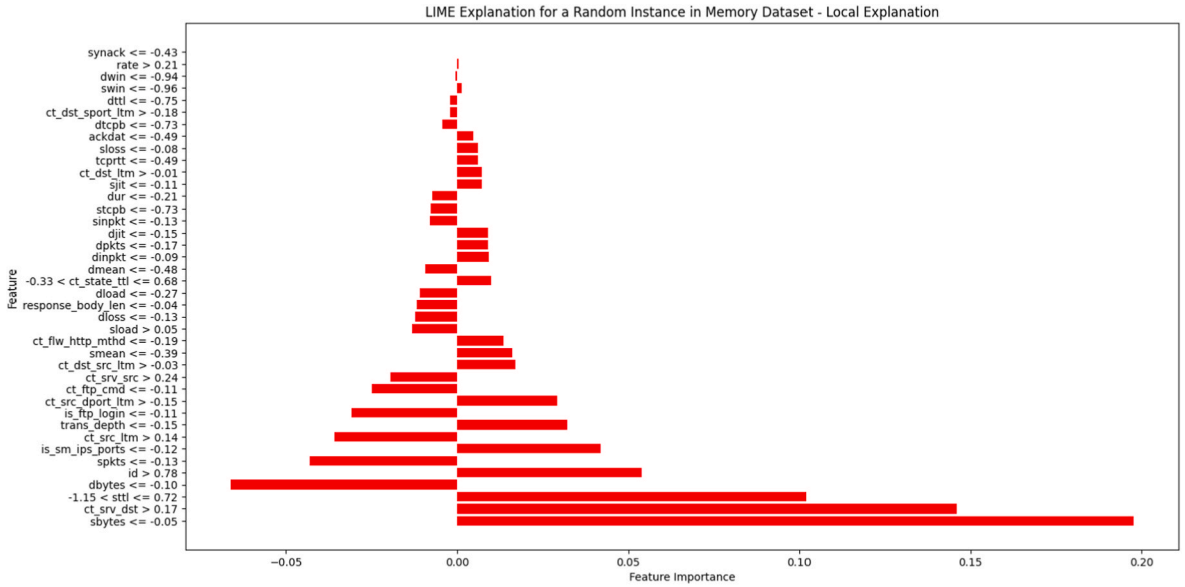

Fig. 8. LIME local explanation—features' importance for a random instance from dataset.

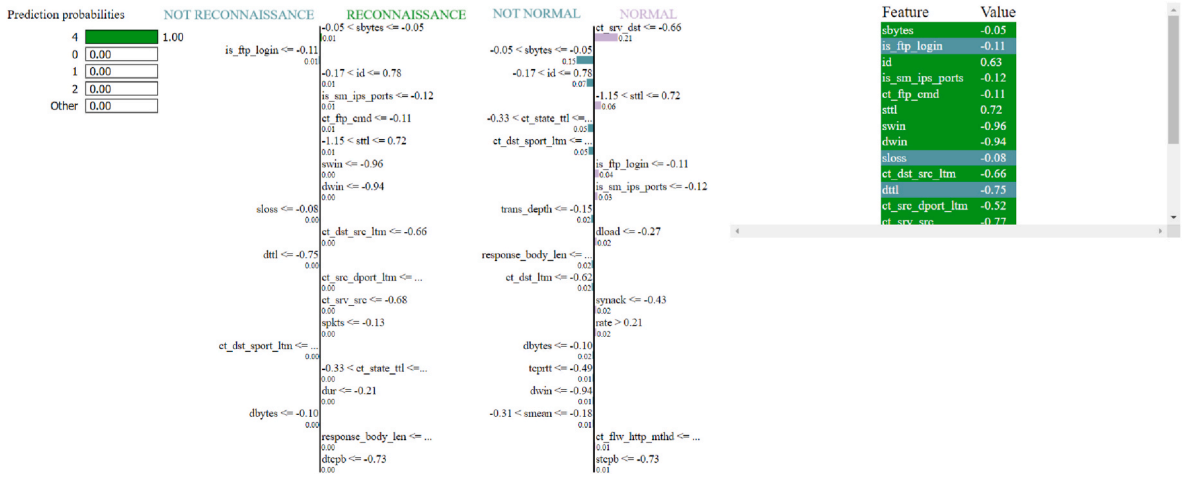

Fig. 9. LIME local explanation—features' importance with respect to specific attack categories.

6. Conclusion and future work

Explainable Artificial Intelligence (XAI) addresses the challenge of opaque AI systems in Digital Forensics and related fields by providing easily understandable explanations for AI model predictions. An exhaustive XAI-DF framework is proposed to standardize the workflow of investigations utilizing AI. The implementation of the framework is demonstrated in memory and network forensics investigative scenarios.

The XAI-DF project is introduced with an initial contribution of a memory forensics database that may be utilized not only for XAI-specific DF research but generally for other DF domains as well. Some memory features including process, network, injected code, API hooks, and process privilege features are extracted from the memory database in its current form followed by classification results' explanations for interpretability.

For future work, we aim to expand the memory database (by adding further memory dumps, including OSs of various vendors (macOS, Linux, etc.) and their versions). More records in the database will improve the efficiency of XAI models' training and testing capabilities. In addition, we are working to incorporate multi-class labels of malware activity in the memory database. It is also pertinent to note that memory dumps from actual host machines (in addition to VMs) with bigger RAM sizes also need to be incorporated to reflect modern-day sizes.

Acknowledgement

This study is supported by Research Incentive Funds (activity code: R23064), Research Office, Zayed University, Dubai, United Arab Emirates.

References

Alam, S., Altiparmak, Z., 2024. XAI-CF – Examining the Role of Explainable Artificial Intelligence in Cyber Forensics [WWW Document]. arXiv.org. https://doi.org/10.48550/arXiv.2402.02452.

Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F., 2020. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion 58, 82–115.

Baggili, I., Behzadan, V., 2019. Founding the Domain of AI Forensics [WWW Document]. arXiv.org. https://doi.org/10.48550/arXiv.1912.06497.

Docterman, E., 2015. Robot Kills Man at Volkswagen Plant [WWW Document]. Time. https://time.com/3944181/robot-kills-man-volkswagen-plant/.

Dunsin, D., Ghanem, M.C., Ouazzane, K., 2022. The use of artificial intelligence in digital forensics and incident response in a constrained environment. International Journal of Information and Communication Engineering 16, 280–285.

Hall, S.W., Sakzad, A., Choo, K.R., 2021. Explainable artificial intelligence for digital forensics. WIREs Forensic Science 4. https://doi.org/10.1002/wfs2.1434.

Hall, S.W., Amin, Sakzad, Minagar, Sepehr, 2022. A proof of concept implementation of explainable artificial intelligence (XAI) in digital forensics. Lect. Notes Comput. Sci. 66–85. https://doi.org/10.1007/978-3-031-23020-2_4.

Kalutharage, C.S., Liu, X., Chrysoulas, C., Pitropakis, N., Papadopoulos, P., 2023. Explainable AI-based DDOS attack identification method for IoT networks. Computers 12, 32. https://doi.org/10.3390/computers12020032.

Levin, S., Woolf, N., 2017. Tesla Driver Killed while Using Autopilot Was Watching Harry Potter, Witness Says [WWW Document]. the Guardian. https://www.theguardian.com/technology/2016/jul/01/tesla-driver-killed-autopilot-self-driving-car-harry-potter.

Moustafa, N., Slay, J., 2015a. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set) [WWW Document]. IEEE Xplore. https://doi.org/10.1109/MilCIS.2015.7348942.

Moustafa, N., Slay, J., 2015b. The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems. In: 2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS). https://doi.org/10.1109/badgers.2015.014.

Ribeiro, M.T., Singh, S., Guestrin, C., 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier [WWW Document]. arXiv.org. https://arxiv.org/abs/1602.04938.

Solanke, A.A., 2022. Explainable digital forensics AI: towards mitigating distrust in AI-based digital forensics analysis using interpretable models. Forensic Sci. Int.: Digit. Invest. 42, 301403 https://doi.org/10.1016/j.fsidi.2022.301403.

Volatility, 2024. The Volatility Foundation - Open Source Memory Forensics [WWW Document]. volatilityfoundation. https://volatilityfoundation.org/.

Yampolskiy, R.V., S, S.M., 2016. Artificial Intelligence Safety and Cybersecurity: a Timeline of AI Failures [WWW Document]. arXiv.org. https://arxiv.org/abs/1610.07997.

Yudkowsky, E., 2008. Artificial intelligence as a positive and negative factor in global risk. https://intelligence.org/files/AIPosNegFactor.pdf.

Zeltser, L., 2021. Free Malware Sample Sources for Researchers [WWW Document]. zeltser.com. https://zeltser.com/malware-sample-sources/.