On the Use of Machine Learning for Modern IoT ELF Malware Detection

Cristian Souza and Daniel Batista

DFIR Specialist, Ph.D Student
(Kaspersky & IME-USP)

Agenda

IoT security

- Security flaws in Internet of Things (IoT) devices can be critical to users' privacy and personal safety.
- The heterogeneity of protocols requires the use of adaptive approaches and a holistic view of the infrastructure for effectively mitigating modern threats.
- The advent of the IoT paradigm was accompanied by an increase in the number of malicious programs designed for ARM and MIPS architectures.

IoT malware

- Mirai and other botnets.
- Ransomware continue to be among the main threats to industrial environments.
- Several advances have been and continue to be made by industry and academia in the areas of malicious artifact analysis and detection.

IoT malware detection

- Detection tools must be able to identify and contain unknown threats.
- Multiple analysis techniques must be employed to increase the reliability of solutions.
- Behavioral analysis stands out as the most effective at detecting zero-day malware.

Proposal

- Since many IoT devices are based on Linux and run ELF binaries, this work focuses on **detecting IoT ELF malware through combined static and dynamic analysis**.
- We use a dataset of **5,169 binaries**, compare seven classification algorithms based on Machine Learning (ML), and assess both detection accuracy and classification speed.

Related work

- **Tahir (2024):** A small-scale study using a Multi-Layer Perceptron, achieving 98.57% detection accuracy.
- **Tien (2020):** A framework combining static analysis on a 6,000-sample dataset, with an overall accuracy of 98%.
- **Ravi (2022):** An ELF-binary feature-extraction approach trained on 1,773 malicious files, reaching 99% accuracy.

Our approach

- Unlike other solutions, our work considers both **static and dynamic attributes** of malicious artifacts for classification.
- **We compare different algorithms implemented**, aiming to guide professionals in choosing the most appropriate models.
- **We make the dataset used public** in order to disseminate and encourage new research.

Data collection

- We developed a Python script[1] capable of collecting malicious samples from the MalwareBazaar[2] project.
- We specifically filtered for ELF files, which can be recognized by the signature `0x464c457f`. This resulted in 3,306 unique samples.

[1] https://gist.github.com/cristianzsh/2e88b3c33f58a9b83e268d0050eadbdb
[2] https://bazaar.abuse.ch

Malware architectures

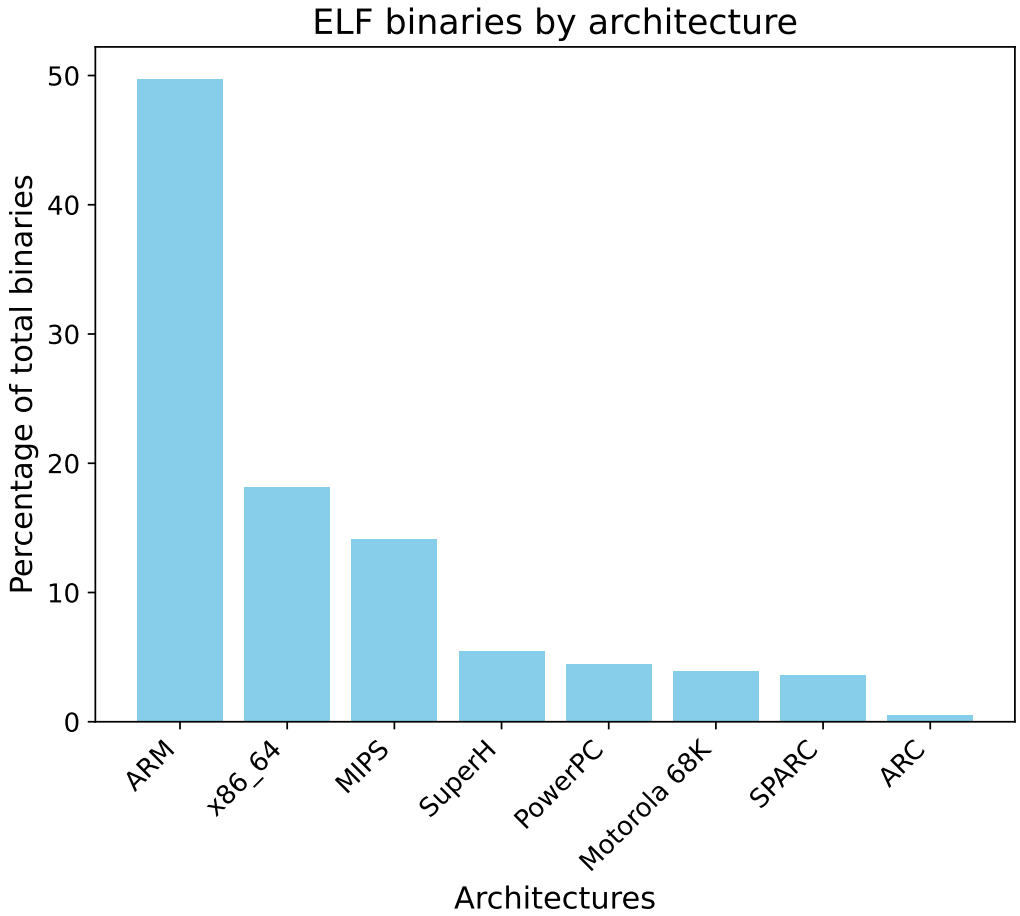

Figure: Malware architectures

Malware verdicts

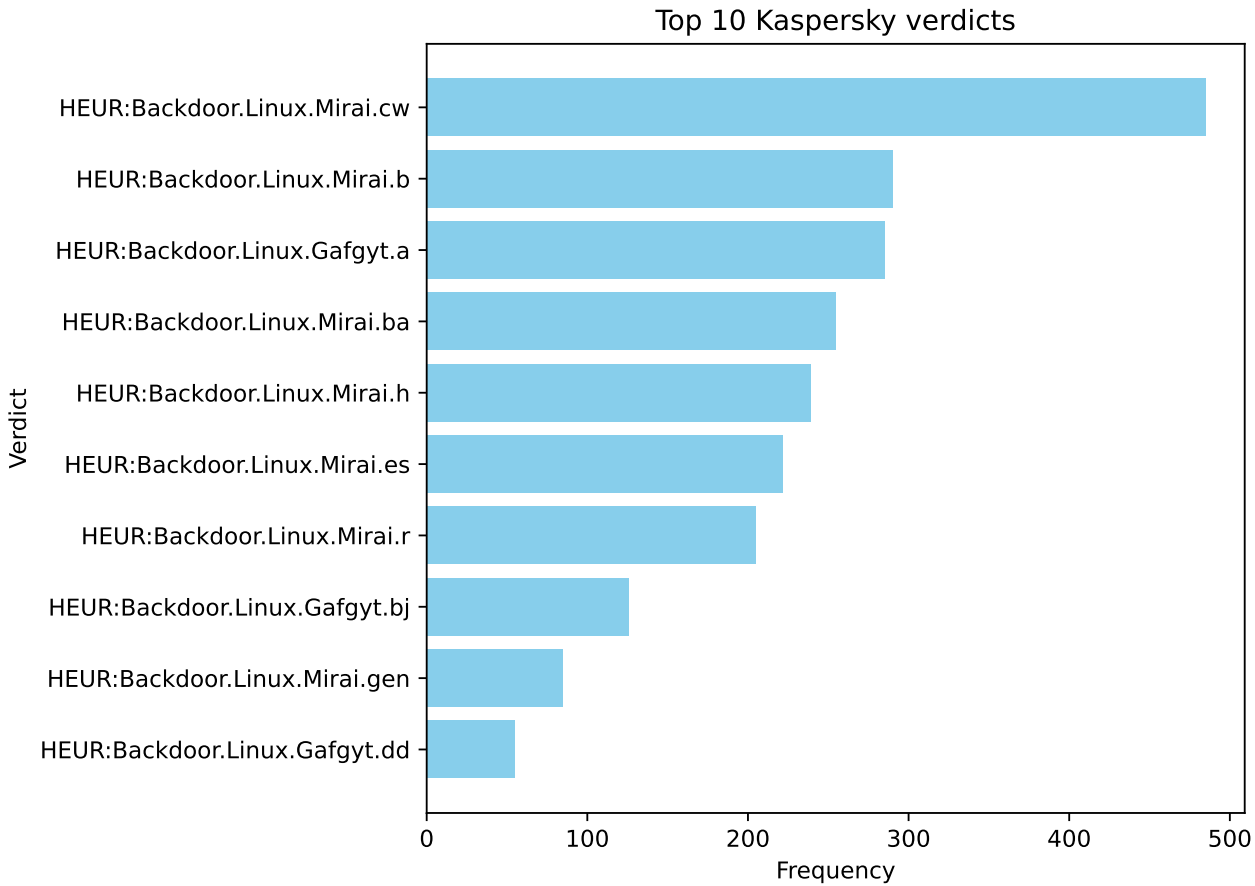

Figure: Malware verdicts according to Kaspersky's engine

Data collection

- For collecting the goodware samples, we relied on the artifacts present in pristine Linux systems.
- These binaries are located at `/bin`, `/sbin`, and `/usr/bin`; resulting in a total of **1,863 artifacts**.

Data collection

Table: Distribution of Samples

Type	Count	Percentage
Malware	3,306	64.0%
Goodware	1,863	36.0%
Total	5,169	100.0%

Feature extraction

- Feature extraction proceeds in two phases, gathering static and dynamic indicators.
- Static indicators are obtained by inspecting the ELF structure and performing string analysis.
- To approximate runtime behavior, we scan the binary's raw content for keywords suggesting process manipulation, system calls, and signals.

Implemented algorithms

- Our classification pipeline is developed entirely in Python, making extensive use of the `scikit-learn`[3] library for training and evaluation.
- We also employ the Python implementation of `XGBoost`[4] for gradient boosting.

[3] https://scikit-learn.org
[4] https://xgboost.readthedocs.io

Implemented algorithms

- Random Forest (RF)
- Support Vector Machine (SVM)
- Multi-Layer Perceptron (MLP)
- K-Nearest Neighbors (KNN)
- Logistic Regression (LR)
- Naive Bayes (NB)
- XGBoost (XGB)

Evaluation metrics

- **Accuracy:** $\frac{TP + TN}{TP + TN + FP + FN}$
- **Precision:** $\frac{TP}{TP + FP}$
- **Recall:** $\frac{TP}{TP + FN}$
- **F1-Score:** $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Results

Table: Performance metrics for classification algorithms

Algorithm	Accuracy	Precision	Recall	F1-Score
Random Forest	**99.33%**	0.97	**0.99**	**0.98**
SVM	95.00%	0.93	0.90	0.91
MLP	87.95%	0.88	0.86	0.87
KNN	97.33%	**0.98**	0.96	0.97
Logistic Regression	96.80%	0.96	0.96	0.96
Naive Bayes	92.83%	0.97	**0.99**	**0.98**
XGBoost	90.98%	0.91	0.90	0.90

Classification time

Table: Classification time (ms) for each algorithm

Algorithm	**Mean (ms)**	**Min (ms)**	**Max (ms)**
Random Forest	82.150	68.184	93.076
SVM	85.845	69.797	103.146
MLP	91.971	73.474	112.544
KNN	86.344	72.969	101.106
Logistic Regression	82.631	65.084	99.746
Naive Bayes	79.247	69.547	88.131
XGBoost	79.903	73.474	86.163

Conclusion

- Among the seven classifiers tested, Random Forest achieved the highest accuracy at 99.33%.
- Although it is not the fastest it still provides near real-time predictions suitable for many IoT scenarios.
- Future directions include refining the feature extraction process to capture more elaborate anti-analysis or obfuscation techniques.
- We hope to inspire further research and foster improvements in safeguarding IoT and ICS ecosystems against evolving malware threats.

GitHub repository

https://github.com/cristianzsh/malware-research

On the Use of Machine Learning for Modern IoT ELF Malware Detection

Cristian Souza and Daniel Batista

DFIR Specialist, Ph.D Student
(Kaspersky & IME-USP)