SCIENCE CHINA
Information Sciences

• LETTER •

November 2026, Vol. 69, Iss. 11, 219104:1–219104:2
https://doi.org/10.1007/s11432-025-4961-3

RAGMIT: rapid APT group malware identification technology

Hao LIU[1,2,3†], Zhiyuan ZENG[1,2,3†], Hui LU[1,2,3], Yuan LIU[1,2,3] & Zhihong TIAN[1,2,3*]

[1]*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510555, China*
[2]*Huangpu Research School of Guangzhou University, Guangzhou 510530, China*
[3]*Guangdong Key Laboratory of Industrial Control System Security, Guangzhou 510555, China*

Received 25 July 2025/Revised 12 December 2025/Accepted 30 April 2026/Published online 19 October 2026

Citation Liu H, Zeng Z Y, Lu H, et al. RAGMIT: rapid APT group malware identification technology. Sci China Inf Sci, 2026, 69(11): 219104, https://doi.org/10.1007/s11432-025-4961-3

As technology advances, key sectors like finance and industry depend more on cyberspace, increasing the risk of malicious attacks with serious political and economic effects. This has led to the rise of advanced persistent threat (APT) groups [1] that threaten network security. As these attacks get more complex [2], quickly detecting, deceiving, and defending them becomes more important [3–5]. Current malware detection methods mainly focus on features, models, and accuracy, but they do not balance accuracy and speed well and need heavy computing power, which limits real-time use. To solve this, we introduce a new metric called time-average accuracy (TAcc) and propose a rapid APT group malware identification technology called RAGMIT. RAGMIT uses machine learning to identify APT groups by analyzing assembly code and API features. Tested on a dataset with 9 APT groups and 839 benign samples, RAGMIT achieved the best TAcc (0.041998%/s), thereby showing it can improve network security with high accuracy and low detection time.

Motivations. This section presents the core motivations behind RAGMIT, focusing on three key aspects: instruction analysis, API function analysis, and their integrated interpretation in binary malware analysis.

Operation instructions in disassembled binaries represent processor actions and are vital for understanding malicious behavior. API functions reveal a program's interaction with system resources and are essential for static and dynamic analysis. They are categorized in Figure 1(b).

Together, instruction and API analysis provide complementary views (low-level execution patterns and high-level system interactions) enabling a comprehensive understanding of binary behavior. Their joint examination forms the foundation of RAGMIT's approach to detecting and characterizing malware.

The requirement for efficiency in APT identification is two-fold. First, defense systems must block active threats in near real-time based on detection results; second, security analysts require rapid initial categorization to prioritize subsequent deep-dive forensics. Consequently, malware classification remains a time-critical process where high latency can lead to irrecoverable system compromise.

RAGMIT. As shown in Figure 1, building on the motivation analysis, this study proposes RAGMIT, a rapid identification method for APT organization malware. RAGMIT consists of three stages: assembly code extraction (Stage 1), feature extraction (Stage 2), and APT organization identification (Stage 3).

In Stage 1, IDA Pro[1)] disassembles the binary into LST format to preserve the complete structural information, including code, data, and metadata. The LST format is preferred over ASM to ensure full instruction coverage.

Stage 2 extracts features based on instruction and API types defined in *Motivations*. Specifically, seven instruction types (e.g., `inst_math`, `inst_bit`) and twelve API categories (e.g., `api_file`, `api_net`) are counted from the disassembled code. These raw counts are normalized by the total number of instructions to form a fixed-length feature vector for each sample.

In Stage 3, a random forest (RF) classifier is trained using the extracted features and known labels. Due to the relatively low dimensionality of the feature space, RF outperforms high-dimensional models in both efficiency and accuracy, as validated in *Experiments*. It effectively captures discriminative patterns in APT-related binaries, enabling accurate attribution to specific threat groups.

By integrating static disassembly with lightweight, semantically meaningful features, RAGMIT achieves fast and effective APT malware classification without relying on complex deep learning or large-scale data preprocessing.

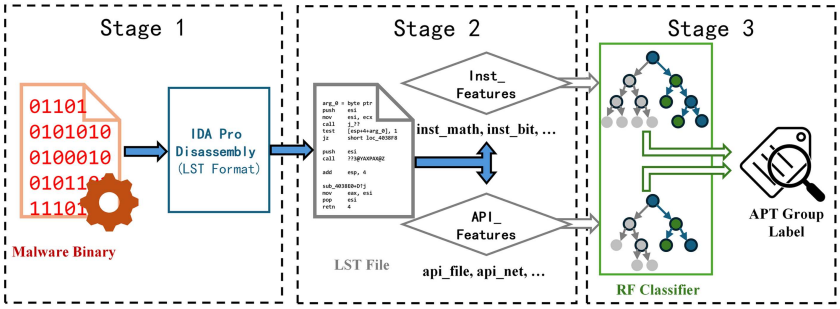
Experiments. The experiment was conducted on a system equipped with an Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60 GHz (32 cores, 64 threads), an NVIDIA GeForce RTX 4090 GPU with 24 GB memory, 503 GB RAM, and 21.9 TB storage, running Ubuntu 20.04.6 LTS; Python libraries include scikit-learn 1.1.3 and PyTorch 2.1.2; five-fold cross-validation is used for evaluation.

The dataset is sourced from open-source websites[2)], and this article collected local benign software. Only PE files were used for

* Corresponding author (email: tianzhihong@gzhu.edu.cn)
† These authors contributed equally to this work.

1) https://hex-rays.com/ida-pro.
2) https://github.com/cyber-research/APTAttribution.

© Science China Press 2026



(a)

Feature	Description
inst_math	Arithmetic operations (add, sub)
inst_bit	Bitwise operations (and, shl)
inst_logic	Logical comparisons (cmp, test)
inst_rep	String repetition (movs)
inst_port	I/O port access (in, out)
inst_reg	Register data transfer (mov, push)
inst_stack	Stack manipulation (call, ret, push, pop)
api_auth	Access control
api_dll	DLL management
api_file	File operations
api_net	Network operations
api_obj	Object handling
api_reg	Registry modifications
api_proc	Process control
api_res	Resource allocation
api_serv	Service manipulation
api_sync	Thread coordination
api_sys	System queries
api_time	Time operations

(b)

APT Group	Number
APT 10	224
APT 19	30
APT 21	86
APT 28	161
APT 29	262
Dark Hotel	264
Energetic Bear	132
Equation Group	395
Gorgon Group	214
Benign	839
Total	2607

(c)

Alg.	Train time	Test time	Acc	TA19	TA256	TA2000
LR	0.1331	0.0012	45.49	0.0204	0.0172	0.0028
SVM	0.1005	0.0395	67.36	0.0302	0.0303	0.0044
DT	0.0211	**0.0007**	86.23	0.0387	0.0297	0.0044
RF	**0.3571**	0.0075	**93.56**	**0.0420**	0.0335	0.0058
KNN	0.0026	0.0541	81.63	0.0366	0.0311	0.0052
NB	0.0103	0.0010	47.07	0.0211	0.0192	0.0028
MLP	5.7096	0.0032	73.92	0.0331	0.0322	0.0053
GB	6.3388	0.0071	90.10	0.0403	0.0318	0.0057
LGBM	0.8346	0.0034	92.48	0.0415	**0.0341**	**0.0059**
RNN	88.8424	0.0230	84.70	0.0366	0.0307	0.0054
LSTM	179.425	0.0281	89.64	0.0372	0.0307	0.0053
GRU	271.150	0.0322	89.72	0.0359	0.0298	0.0053
TF	387.726	0.0403	69.20	0.0265	0.0268	0.0055
TCN	110.944	0.0924	68.58	0.0293	0.0279	0.0021

(d)

Figure 1 (Color online) Framework, features, dataset, and comparison of RAGMIT. (a) Framework of RAGMIT; (b) features of RAGMIT; (c) dataset of RAGMIT; (d) comparison of RAGMIT. Bold indicates the optimal indicator value in the comparison algorithm.

the experiments, and the number of available samples is shown in Figure 1(c).

The indicators used in this experiment are common malware classification metrics such as accuracy,

$$\mathrm{Acc} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}. \tag{1}$$

In (1), true positive (TP) is correctly predicted positive cases. True negative (TN) is correctly predicted negative cases. False positive (FP) is incorrectly predicted positive cases. False negative (FN) is incorrectly predicted negative cases.

In addition, in order to measure the efficiency of the classification model in training, a new metric TAcc (2) is adopted in this study,

$$\mathrm{TAcc} = \frac{\mathrm{Acc}}{T}. \tag{2}$$

In (2), T refers to the total time, which consists of the feature extraction time for all samples, the average training time per fold, and the average testing time per fold.

To evaluate the effectiveness of RAGMIT, 13 distinct algorithms were implemented, including nine machine learning[3)] and four deep learning[4)] methods. Specifically: logistic regression (LR), support vector machine (SVM), decision tree (DT), random forest (RF), K-nearest neighbors (KNN), naive Bayes (NB), multi-layer perceptron (MLP), gradient boosting (GB), LightGBM (LGBM), recurrent neural network (RNN), long short-term memory (LSTM), gated recurrent unit (GRU), and Transformer (TF). Note that some researchers use time convolutional network (TCN) classifiers.

Notably, as a rapid identification method, RAGMIT prioritizes training accuracy (TAcc) over overall accuracy (Acc). Therefore, excessive preprocessing, representation learning, or generative methods that inflate Acc at the cost of efficiency are not aligned with its design principle. A compact feature vector is adopted to ensure fast and practical model inference.

To justify our feature engineering and provide a comparative study against standard methodologies, RAGMIT's 19-dimensional categorized features (TA 19) are benchmarked against two representative baselines: 256-dimensional byte frequency (TA 256), typical of data-level detection, and 2000+ unclassified raw instruction/API counts (TA 2000), typical of high-fidelity semantic modeling in Figure 1(d).

As shown, RAGMIT achieves the highest TAcc (0.0420%/s) and accuracy (93.56%). While deep learning models (RNN-Transformer) exhibit competitive accuracy, their significantly higher training overhead leads to lower TAcc, validating RAGMIT's efficiency in balancing detection performance with computational costs.

Limitations. While RAGMIT achieves high efficiency, its effectiveness against advanced metamorphic code and practical performance in large-scale, real-world cyber ranges remain to be further explored in future work.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. U2436208, 62372129), Guangdong S&T Program (Grant No. 2024B0101010002), and Project of Guangdong Key Laboratory of Industrial Control System Security (Grant No. 2024B1212020010).

References

1 Wang Z, Zhou Y, Liu H, et al. ThreatInsight: innovating early threat detection through threat-intelligence-driven analysis and attribution. IEEE Trans Knowl Data Eng, 2024, 36: 9388–9402
2 Liu H, Guo C, Cui Y, et al. 2-SPIFF: a 2-stage packer identification method based on function call graph and file attributes. Appl Intell, 2021, 51: 9038–9053
3 Liu H, Zhou Y, Fang B, et al. PHCG: PLC honeypoint communication generator for Industrial IoT. IEEE Trans Mobile Comput, 2025, 24: 198–209
4 Wang R, Yang C, Deng X, et al. Turn the tables: proactive deception defense decision-making based on Bayesian attack graphs and Stackelberg games. Neurocomputing, 2025, 638: 130139
5 Xu Y, Li M, Fang B, et al. Neural honeypoint: an active defense framework against model inversion attacks. IEEE Trans Neural Netw Learn Syst, 2025, 36: 16186–16197

3) https://scikit-learn.org/stable/.
4) https://docs.pytorch.org/docs/stable/index.html.