Vulnerability Analysis of Network Traffic Based Mirai Detection Systems

Rajdeep Kumar Dutta*, Bishal Chhetry*, Rakesh Matam*, Ferdous Ahmed Barbhuiya*, and Ashok Singh Sairam†
*Dept. of Computer Science and Engineering, Indian Institute of Information Technology Guwahati, India
Email: {rajdeep.dutta, bishal.chhetry, rakesh, ferdous}@iiitg.ac.in
†Department of Mathematics, Indian Institute of Technology Guwahati, India
Email: ashok@iitg.ac.in

***Abstract*—Mirai remains one of the most pervasive IoT botnets, whose evolving variants exploit weakly secured devices to launch large-scale DDoS attacks. Network Intrusion Detection Systems (NIDS) are one component of the multilayered defense deployed to safeguard a network. Existing NIDS achieve high accuracy but remain opaque and vulnerable to subtle, functionality-preserving code modifications that alter traffic characteristics without changing malicious intent. This work addresses these gaps by performing lifecycle-aware analysis of Mirai reconnaissance, exploitation, infection, and attack using an enriched dataset combining IoT-23 and CICIoT2023. We correlate source-code behaviors with discriminative traffic features through SHAP-based explainability and feature-correlation analysis. Using absolute SHAP contributions, we theoretically and empirically estimate model degradation bounds without re-training, showing that early-phase obfuscations reduce classifier logits by $\Delta z \in [-0.24, -0.05]$ ($\approx$ 2–6% confidence loss), while DDoS traffic remains inherently non-evasive. The framework exposes how explainable models can quantify robustness, guiding the development of resilient, interpretable IoT malware detectors.**

***Index Terms*—Internet of Things, IoT Malware life-cycle, Mirai botnet, Network Traffic Analysis, Anomaly Detection, Explainable-AI**

I. Introduction

Among IoT malware families, *Mirai* is among the most studied due to its ability to exploit weak device configurations and launch large-scale Distributed-Denial-of-Service (DDoS) attacks that disrupt critical services [1]. Despite extensive research, most detection efforts emphasize the attack phase, overlooking earlier stages such as reconnaissance, exploitation, and infection, where proactive defense is possible.

Existing network-based intrusion detection techniques, whether signature-based or learning-based, achieve high reported accuracies but remain fragile against evolving variants. Signature-based systems rely on fixed traffic signatures that fail under minor packet or payload alterations [2], [3], while anomaly-based models generalize better yet behave as opaque black boxes, limiting interpretability and robustness in heterogeneous IoT environments [4]. Consequently, subtle functionality-preserving source-code modifications to Mirai can suppress behavioral indicators, leading to undetected compromises.

Effective detection therefore requires associating Mirai's lifecycle behaviors with explainable and verifiable network features rather than relying solely on statistical correlations. Explainable AI (XAI) methods such as SHAP provide feature-level transparency [5], but existing datasets incompletely represent Mirai's lifecycle: IoT-23 lacks exploitation traffic, while CICIoT2023 includes complementary brute-force samples. Moreover, Mirai's open-source code, which was used in this analysis [6], can be trivially altered e.g., replacing TCP scans with UDP or randomizing credential tables to obscure tell-tale features. These gaps underscore the need for lifecycle-aware and code-resilient detection frameworks.

This study addresses these challenges through five contributions: (i) stage-wise mapping of Mirai's lifecycle to discriminative traffic features, (ii) construction of an enriched IoT-23 + CICIoT2023 dataset covering all phases, (iii) feature validation using correlation, Random Forest, Logistic Regression, and SHAP, (iv) correlation of features with source-code routines to model realistic obfuscations, and (v) theoretical quantification of detection degradation via SHAP, showing a measurable confidence loss when key features are neutralized. The remainder of the paper reviews related work (Section II), describes the methodology (Section III), presents source-code analysis and evasion design (Section IV), discusses results and degradation estimation (Section V), and concludes with practical implications (Section VI).

II. Related Work

Network-Traffic-Based Detection: IoT botnet detection, particularly for *Mirai*, has relied heavily on network traffic analysis through either rule-based or learning-based systems. Signature-driven IDSs such as Snort and Suricata [7], [8] detect known threats via static byte or header patterns but fail against encrypted, polymorphic, or zero-day traffic [2]. Studies show that minor behavioral changes allow Mirai variants to evade such static filters [9], [10]. Learning-based anomaly detectors including LSTM [11] and CNN models like ExPose [12] capture temporal and spatial dependencies for phase-wise identification. However, they suffer from high false-positive rates in heterogeneous environments [13] and rely on incomplete datasets: IoT-23 lacks exploitation traffic, limiting full lifecycle validation [14].

Explainable AI (XAI) for IDS: Explainability frameworks such as SHAP and LIME [4], [15] improve interpretability by linking features to model outputs [16], but most remain post-hoc tools with poor generalizability across IoT datasets [5]. They also neglect adversarial manipulations at the source-code level, leaving systems vulnerable when Mirai modifies protocol usage or credential logic [10].

Source-Code Analysis: Mirai's architecture and command-and-control (C2) design have been extensively characterized [1], [10], [17], [18], yet few studies quantify how code changes manifest in traffic features. Sinanović *et al.* [19] highlighted this link conceptually but did not evaluate its effect on model accuracy.

Summary of Gaps: Existing works lack a unified, explainable framework connecting Mirai's source code, lifecycle-specific traffic behaviors, and their measurable influence on detection confidence. This study addresses that gap by correlating source-level modifications with SHAP-validated network features and theoretically estimating the resulting model degradation.

III. Methodology

This study investigates Mirai's lifecycle through combined network-traffic and source-code analysis to evaluate how functionality-preserving code changes affect detectability. To achieve full lifecycle coverage, an enriched dataset was created by merging IoT-23 with brute-force traces from CICIoT2023, capturing reconnaissance, exploitation, infection, persistence, propagation, and attack phases. Table I gives a brief description of the features extracted by Zeek.

TABLE I: Key Fields in Zeek's `conn.log` and Their Descriptions

Feature	**Description**
`protocol`	Transport-layer protocol (e.g., TCP, UDP).
`service`	Application service detected (e.g., HTTP, DNS); "–" if unknown.
`duration`	Connection lifetime in seconds.
`orig_bytes`	Bytes sent from originator to responder.
`resp_bytes`	Bytes sent from responder to originator.
`conn_state`	Final connection outcome: `S0` (no reply), `SF` (normal close), `RSTR` (reset), `OTH` (other). Indicates connection completeness or anomalies.
`local_orig`	True if the originator host is local.
`local_resp`	True if the responder host is local.
`missed_bytes`	Bytes lost or not captured in trace.
`history`	Encodes packet sequence: `S` (SYN), `H` (handshake), `R` (reset), `D` (data), `F` (FIN). Summarizes state transitions.
`orig_pkts`	Packets sent by the originator.
`orig_ip_bytes`	Total bytes from originator IP.
`resp_pkts`	Packets sent by the responder.
`resp_ip_bytes`	Total bytes from responder IP.
`tunnel_parents`	Parent tunnels (e.g., VPNs) containing the connection.

A. Lifecycle Characterization

Each Mirai stage exhibits distinct network artifacts. *Reconnaissance* produces rapid TCP SYN probes to Telnet/SSH ports, yielding incomplete handshakes (`S0`) and high packet counts. During *exploitation*, brute-force logins using default credentials form short-lived `S3` connections with small payloads. The *infection* phase downloads binaries from command-and-control (C2) servers, visible as large responder-byte volumes (`resp_ip_bytes`). *Persistence* maintains long-lived connections with minimal state transitions, and the *attack* phase generates heavy outbound traffic (`OTH`) characteristic of TCP/UDP floods. Finally, *propagation* resumes scanning patterns similar to reconnaissance.

B. Feature Validation Pipeline

Four complementary analyzes identified the most discriminative features for each stage: (i) Pearson correlation for linear associations, (ii) Random-Forest importance for non-linear relevance, (iii) Logistic-Regression weights for interpretability, and (iv) SHAP analysis for per-stage explainability of the Random-Forest model. This ensemble ensured that dominant features were validated across both statistical and model-based perspectives.

C. Dataset and Explainability Results

IoT-23 provided labelled traces for reconnaissance, infection, attack, and propagation, while CICIoT2023 contributed exploitation samples, yielding phase complete coverage. Core Zeek connection fields such as `conn_state_S0`, `history`, `orig_pkts`, `resp_ip_bytes`, and `protocol_tcp` were retained after encoding and normalization. SHAP analysis revealed consistent feature dominance across phases: `conn_state_S0` and `orig_pkts` mark reconnaissance; `local_orig` and small packet volumes characterize exploitation; `resp_ip_bytes` denotes infection via C2 downloads; and `history` with `conn_state_OTH` distinguishes attack-phase DDoS floods. These validated features form the empirical basis for modelling source-level obfuscation and theoretical degradation estimation.

IV. Source Code Analysis and Evasion Strategies

We analyze Mirai's core routines (Table III) and propose compact, functionality-preserving evasion techniques that target the high-impact network features identified by our explainability pipeline (Table I, Fig. 1, Figs. 8–11). The aim is not to redesign Mirai but to show small, realistic code-level tweaks that (i) retain the original control flow (scan $\rightarrow$ exploit $\rightarrow$ download $\rightarrow$ attack) and (ii) collectively reduce classifier reliance on concentrated SHAP attributions.

a) Design principles: Apply *joint perturbations* across correlated features (Fig. 1), tune perturbation magnitudes to align malicious statistics toward benign percentiles (KS/Wasserstein guidance), and avoid protocol changes unsupported by the C2 to prevent functional breakage.

b) Functional preservation: The proposed edits keep the semantic steps required for Mirai to succeed: reconnaissance still discovers open ports, exploitation still tries credentials, infection still retrieves and runs payloads, and DDoS retains volumetric impact. Hence, these changes primarily alter timing, ordering, and payload framing rather than core logic.

TABLE II: Phase-wise attribution (top features) and theoretical degradation bounds. Features listed in decreasing attribution; Δz from SHAP-based bounds; Δp approximated at baseline $p \approx 0.8$.

Phase	**Top attribution features (ordered)**	**$\sum$ Top–K |SHAP|**	**Realistic Δz**	**Approx. Δp (%)**
Reconnaissance / Propagation	`conn_state_S0`, `history`, `orig_pkts`, `protocol_tcp`	0.20	[−0.14, −0.06]	2.0–3.5
Exploitation	`local_orig`, `conn_state_S3`, `orig_bytes`, `orig_pkts`	0.35	[−0.24, −0.10]	3.0–5.5
Infection	`resp_ip_bytes`, `service_http`, `history`, `protocol_tcp`	0.17	[−0.12, −0.05]	1.5–3.0
Attack (DDoS)	`history`, `orig_pkts`, `conn_state_OTH`	0.24	[−0.07, −0.02]	<2.0

Notes: (i) Top features derive from per-class absolute SHAP summaries; features are ordered by decreasing attribution. (ii) $\sum$Top–K |SHAP| gives the phase-level upper bound; realistic Δz assumes 30–70% suppressible contribution under functionality-preserving evasions and observed feature correlations. (iii) $\Delta p \approx \sigma(f)(1-\sigma(f))\,\Delta z$; reported as percentage-point confidence reduction for typical logits ($f \approx 1$–1.5).

TABLE III: Concise evasion strategies (functionality-preserving)

Phase	**Code target / change**	**Primary expected feature effect**
Reconnaissance	`scanner.c`: inter-packet jitter, random source ports, occasional UDP probes (same dest ports).	Smear `conn_state_S0`, reduce tight `orig_pkts` bursts, diversify `history`.
Exploitation	`attempt_login()`: permute credential order, random delays, segment credentials.	Broaden `orig_bytes` / `orig_pkts` distributions; weaken repeated `conn_state_S3` signatures.
Infection	`loader.c`: vary URIs/-headers, chunked or variable transfers (TLS if supported).	Reduce single-valued `resp_ip_bytes` patterns; increase benign-like HTTP variability.
Persistence	`process.c`: mimic legitimate process names; stagger keep-alives.	Make `duration` / `history` resemble benign long-lived sessions.
Attack (DDoS)	`attack.c`: short-interval vector rotation (TCP/UDP/HTTP) while preserving volume.	Minimal realistic reduction in volumetric indicators without degrading attack efficacy.

c) Limits: Volumetric attack traces (high `orig_pkts`, `OTH`) are inherently hard to mask without impairing attack goals, thus DDoS remains largely observable. Early phases (reconnaissance, exploitation, infection) are more amenable to soft obfuscations that measurably reduce classifier logit contributions.

d) Validation checklist: For any implemented evasion report: (i) targeted features and pre/post stats (mean, 5/95 pctiles); (ii) distance to benign distributions (KS/Wasserstein); and (iii) aggregated per-sample Δz and Δp. These confirm distributional grounding and functional integrity.

V. Results and Discussion

This section summarizes the empirical findings from SHAP explainability, feature correlation, and theoretical degradation analysis. Figures 9–12 show phase-wise SHAP distributions, while Fig. 1 illustrates key inter-feature dependencies that inform the evasion strategies in Table III.

A. Feature Attribution and Correlation

SHAP and correlation analyzes confirm that each Mirai lifecycle phase has distinct traffic signatures. During *reconnaissance/propagation*, `conn_state_S0`, `history`, and `orig_pkts` dominate, indicating rapid SYN probes and incomplete TCP handshakes. *Exploitation* is characterized by `local_orig`, `conn_state_S3`, and small payload sizes, representing brute-force login attempts. *Infection* exhibits strong attribution for `resp_ip_bytes` and `service_http`, reflecting binary downloads from command-and-control servers. Finally, *attack*-phase traffic shows elevated `conn_state_OTH` and `orig_pkts`, consistent with DDoS floods. Correlation analysis further reveals strong coupling among these fields (e.g., `S0` with `orig_pkts`, `resp_ip_bytes` with `service_http`), implying that effective obfuscation must perturb feature clusters rather than single metrics.

B. Impact of Source-Level Evasions

Mapping source-level changes to SHAP evidence shows that early phases are most vulnerable to obfuscation. Introducing packet jitter and randomized ports in reconnaissance reduces the model's logit confidence by $\Delta z \approx -0.06$ to -0.14. Credential permutation during exploitation yields $\Delta z \approx -0.10$ to -0.24, the highest degradation among all stages. Modifying HTTP header or URI patterns during infection causes smaller shifts ($\Delta z \approx -0.05$ to -0.12), while DDoS floods remain largely immutable ($\Delta z > -0.07$) due to unavoidable volumetric traces. Thus, functionality-preserving code edits can blur stage boundaries in feature space but cannot hide high-intensity attacks.

C. Theoretical Model Degradation Estimation

Model degradation is formalized using the SHAP additivity principle:

$$f(x) = \phi_0 + \sum_i \phi_i(x), \tag{1}$$

where $f(x)$ is the classifier logit and $\phi_i(x)$ denotes feature i's contribution. For an obfuscation transformation T that neutralizes a subset S_{crit} of dominant features,

$$\Delta z = f(T(x)) - f(x) \approx -\sum_{i \in S_{crit}} \phi_i(x). \tag{2}$$

Aggregating top-K mean(|SHAP|) values per phase yields realistic Δz bounds, converted to probability changes by $\Delta p \approx \sigma(f)(1-\sigma(f))\,\Delta z$. For logits near 1–1.5, every -0.1 reduction in Δz lowers detection confidence by roughly 2–3%.

D. Interpretation and Implications

The estimated Δz bounds quantify explainable confidence loss from feasible obfuscations. While exploitation-phase changes can flip near-threshold predictions, high-confidence detections remain stable. Overall, SHAP and source-code correlation confirm that early-stage modifications degrade network-only models measurably, whereas volumetric DDoS traffic remains detectable. These results highlight the limitation of purely network-based IDS and the necessity for hybrid frameworks that integrate host or behavioral telemetry for resilient IoT malware detection.

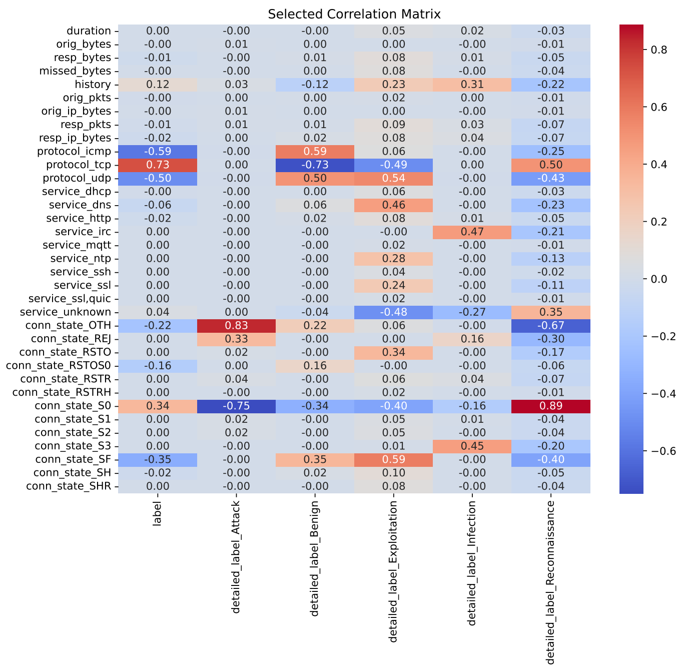

Fig. 1: Correlation Matrix for each stage in Mirai's Lifecycle.

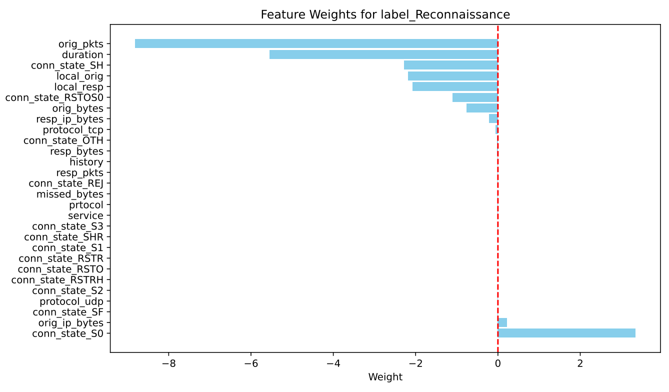

Fig. 2: Feature weights for the Reconnaissance label.

VI. Conclusion

This study provided an explainable, lifecycle-aware analysis of the Mirai botnet, linking source-code routines to network-level behaviors and quantifying detection resilience through SHAP-based feature attribution. By correlating critical traffic features with Mirai's operational stages, we showed that early phases of reconnaissance, exploitation, and infection can be partially concealed using lightweight, functionality-preserving source-code perturbations, whereas volumetric DDoS traffic remains inherently visible. Theoretical logit-level estimates indicated bounded model degradation of $\Delta z \in [-0.24, -0.05]$, corresponding to only a 2–6 % reduction in classifier confidence. Thus, while near-threshold samples may evade detection, high-confidence classifications remain robust. These

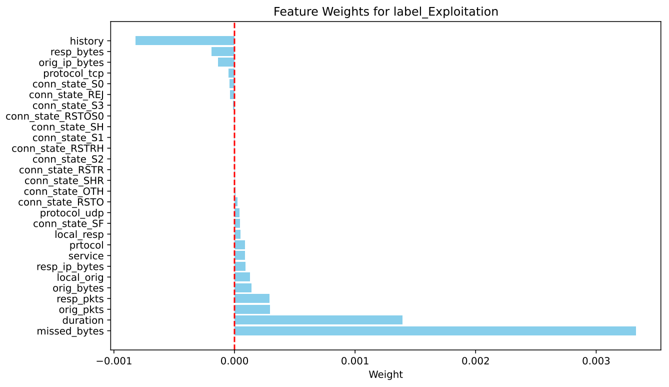

Fig. 3: Feature weights for the Exploitation label.

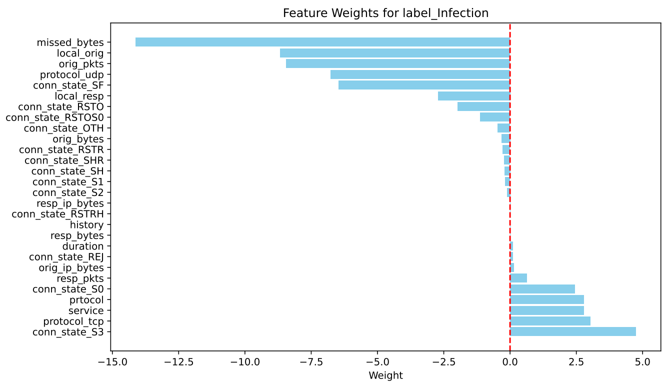

Fig. 4: Feature weights for the Infection label.

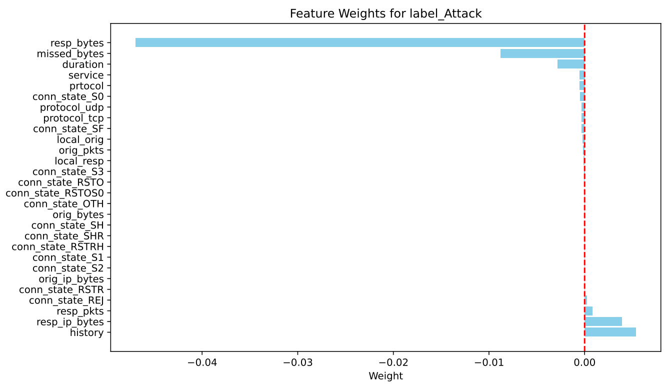

Fig. 5: Feature weights for the Attack label.

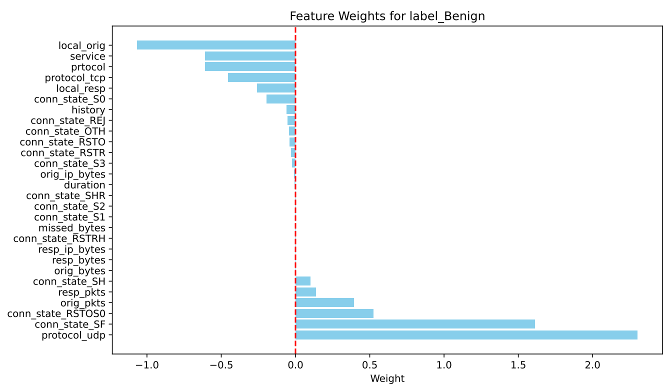

Fig. 6: Feature weights for the Benign label.

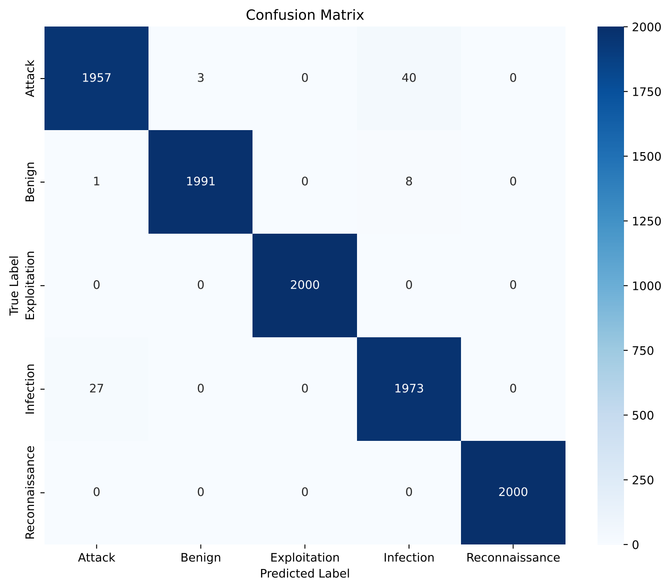

Fig. 7: Confusion Matrix for Random Forest Classifier.

results highlight that modern network-only NIDSs are explainably interpretable yet vulnerable to targeted feature obfuscation, motivating hybrid approaches that fuse network, host, and behavioral telemetry.

Practical Implications: The proposed explainability-driven framework allows security analysts to pinpoint fragile feature dependencies and prioritize retraining or sensor fusion where models exhibit high SHAP concentration. Such insight transforms explainable AI from a post-hoc diagnostic tool into a proactive defense mechanism, guiding the design of resilient, trustworthy IoT intrusion-detection systems deployable in real-world heterogeneous networks.

Acknowledgement: The authors acknowledge the support received from the Ministry of Electronics and Information Technology (MeitY), Government of India, through the project "LogIT: A Federated Learning and Fog-Computing based Framework for IoT Security" (Grant No. AAA-22/11/2022-CSD). This work has also been partially supported by ANRF, Government of India under the Grant ANRF/-PAIR/2025/000010/PAIR.

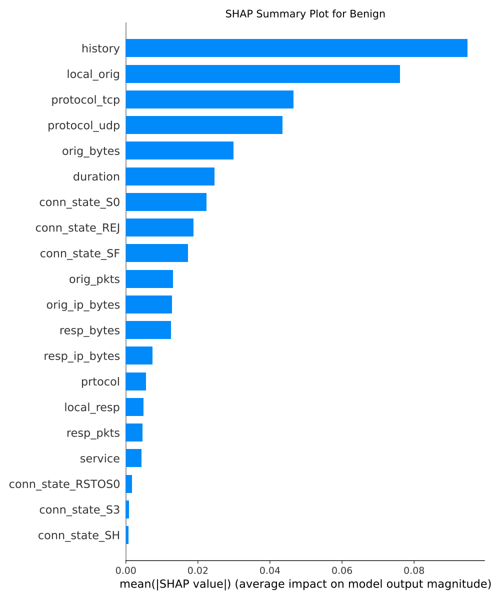

Fig. 8: SHAP summary plot for the Benign label.

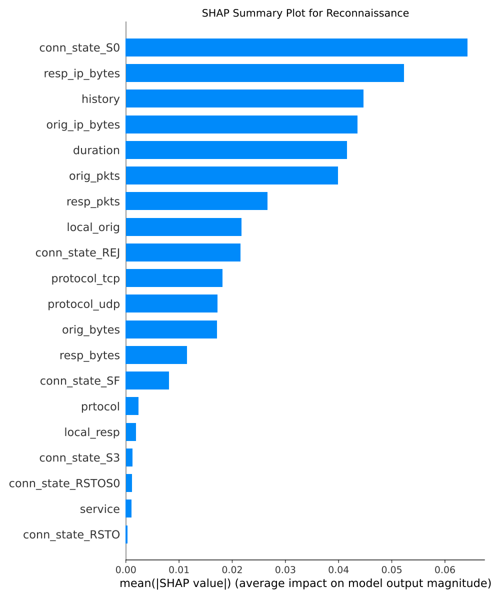

Fig. 9: SHAP summary plot for the Reconnaissance Phase.

References

[1] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai botnet," in *Proc. 26th USENIX Security Symposium (USENIX Security '17)*, (Vancouver, BC, Canada), pp. 1093–1110, 2017.

[2] L. Zhang, S. Yu, D. Wu, and P. Watters, "A survey on latest botnet attack and defense," in *Proc. 2011 IEEE 10th Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 53–60, 2011.

[3] N. B. Said, F. Biondi, V. Bontchev, O. Decourbe, T. Given-Wilson, A. Legay, and J. Quilbeuf, "Detection of Mirai by syntactic and

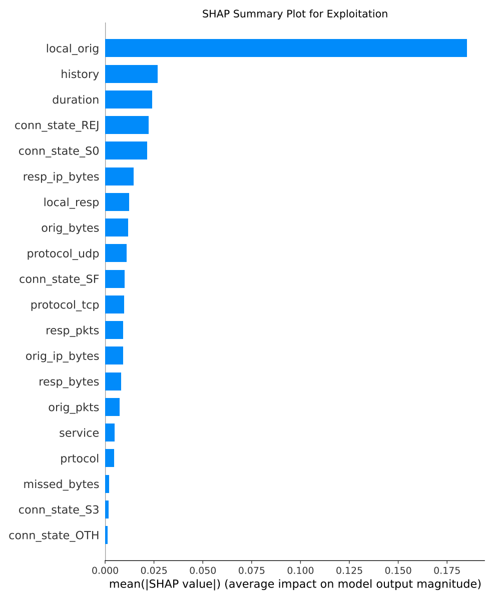

Fig. 10: SHAP summary plot for the Exploitation Phase.

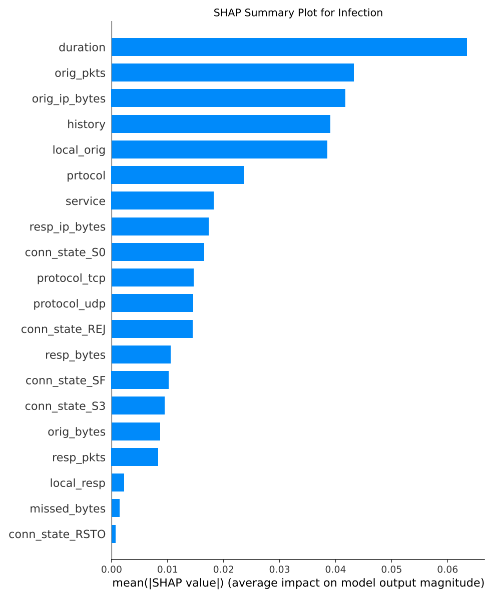

Fig. 11: SHAP summary plot for the Infection Phase.

behavioral analysis," in *Proc. 2018 IEEE 29th Int. Symp. on Software Reliability Engineering (ISSRE)*, pp. 224–235, 2018.

[4] D. Uhříček, K. Hynek, T. Čejka, and D. Kolář, "Bota: Explainable IoT malware detection in large networks," *IEEE Internet of Things Journal*, vol. 10, no. 10, pp. 8416–8431, 2023.

[5] R. Kalakoti, H. Bahsi, and S. Nõmm, "Improving IoT security with explainable AI: Quantitative evaluation of explainability for IoT botnet detection," *IEEE Internet of Things Journal*, vol. 11, no. 10, pp. 18237–18254, 2024.

[6] J. Gamblin, "Mirai source code." Online, 2016. Available: https://github.com/jgamblin/Mirai-Source-Code/tree/master/, Accessed: Aug. 10, 2025.

[7] The Snort Project, "Snort: The open source network intrusion detection system." Online, 2025. Available: https://www.snort.org, Accessed: Jan. 9, 2025.

[8] The Open Information Security Foundation (OISF), "Suricata: The

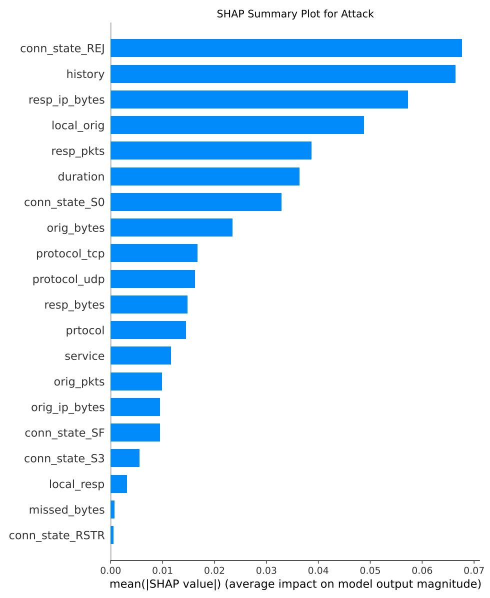

Fig. 12: SHAP summary plot for the Attack Phase.

next generation open source ids/ips/nsm." Online, 2025. Available: https://suricata.io, Accessed: Jan. 9, 2025.

[9] S. Pokhrel, R. Abbas, and B. Aryal, "IoT security: Botnet detection in IoT using machine learning." arXiv preprint arXiv:2104.02231, 2021.

[10] H. Griffioen and C. Doerr, "Examining Mirai's battle over the internet of things," in *Proc. 2020 ACM SIGSAC Conf. on Computer and Communications Security (CCS '20)*, (Virtual Event, USA), pp. 743–756, 2020.

[11] F. Ding, H. Li, F. Luo, H. Hu, L. Cheng, H. Xiao, and R. Ge, "Deeppower: Non-intrusive and deep learning-based detection of IoT malware using power side channels," in *Proc. 15th ACM Asia Conf. on Computer and Communications Security (ASIA CCS '20)*, pp. 33–46, 2020.

[12] J. P. Chapman, "SAD THUG: Structural anomaly detection for transmissions of high-value information using graphics," in *Proc. 27th USENIX Security Symposium (USENIX Security '18)*, (Baltimore, MD, USA), pp. 1147–1164, 2018.

[13] A. Sharma and H. Babbar, "IoT-pot: Machine learning-based detection of Mirai botnet attacks in IoT," in *Proc. 2024 First Int. Conf. on Innovations in Communications, Electrical and Computer Engineering (ICICEC)*, pp. 1–6, 2024.

[14] A. Tang, S. Sethumadhavan, and S. J. Stolfo, "Unsupervised anomaly-based malware detection using hardware features," in *Research in Attacks, Intrusions and Defenses* (A. Stavrou, H. Bos, and G. Portokalidis, eds.), pp. 109–129, Cham, Switzerland: Springer, 2014.

[15] A. Khediri, H. Slimi, A. Yahiaoui, M. Derdour, H. Bendjenna, and C. E. Ghenai, "Enhancing machine learning model interpretability in intrusion detection systems through SHAP explanations and LLM-generated descriptions," in *Proc. 2024 6th Int. Conf. on Pattern Analysis and Intelligent Systems (PAIS)*, pp. 1–6, 2024.

[16] F. Yang, J. Xu, C. Xiong, Z. Li, and K. Zhang, "PROGRAPHER: An anomaly detection system based on provenance graph embedding," in *Proc. 32nd USENIX Security Symposium (USENIX Security '23)*, (Anaheim, CA, USA), pp. 4355–4372, 2023.

[17] X. Zhang, O. Upton, N. L. Beebe, and K. R. Choo, "IoT botnet forensics: A comprehensive digital forensic case study on Mirai botnet servers," *Forensic Science International: Digital Investigation*, vol. 32, p. 300926, 2020.

[18] W. Zaki, R. Abdullah, W. Mohamed, S. R. Selamat, M. Rosli, and S. Yahya, "Constructing IoT botnet detection model based on degree centrality and path analysis," *Journal of Advances in Information Technology*, vol. 15, no. 3, pp. 330–339, 2024.

[19] H. Sinanović and S. Mrdovic, "Analysis of Mirai malicious software," in *Proc. 2017 25th Int. Conf. on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 1–5, 2017.