Designing an Attack-Vector-Based Taxonomy for IoT Malware

Huda Aldawghan, Mounir Frikha
Department of Computer Networks and Communications,
College of Computer Sciences and Information Technology,
King Faisal University, Al-Ahsa, 31982, Saudi Arabia

***Abstract*—This study presents a literature-derived, attack-vector-based taxonomy for IoT malware and complements it with an empirical validation using supervised machine learning. Building on prior surveys and taxonomies of IoT security and malware behavior, we synthesize how existing studies implicitly or explicitly describe infection vectors such as credential abuse, exposed services, firmware exploitation, internal lateral movement, and supply-chain compromise. The resulting taxonomy organises IoT malware according to initial entry mechanisms rather than post-compromise capabilities, providing a vector-centric perspective that aligns more naturally with risk assessment and defensive planning. To demonstrate the practical relevance of this taxonomy, we implement a supervised malware detection model operating on Windows Portable Executable (PE) files. Using malware samples collected from public repositories (e.g., VirusShare and MalwareBazaar) and benign executables from open-source projects, we extract structural, statistical, and metadata-based PE features and train an Extreme Gradient Boosting (XGBoost) classifier with Synthetic Minority Oversampling Technique (SMOTE) for class balancing. The model achieves an accuracy of 98.13% with balanced F1-scores for both malware and benign classes, illustrating that feature-engineered supervised models can effectively support taxonomy-informed detection strategies. The combined conceptual and empirical view highlights how attack-vector taxonomies, IoT threat modeling, and machine learning-based detection can be jointly leveraged to strengthen IoT cyber defense.**

***Keywords*—IoT security; malware taxonomy; attack vectors; cyber threat intelligence; network defense**

I. Introduction

The Internet of Things (IoT) ecosystem has evolved into one of the most pervasive technological paradigms of the digital age, interconnecting billions of heterogeneous devices across industrial, home, healthcare, and urban environments. While this interconnectivity enhances automation and data-driven intelligence, it has also expanded the cyber-attack surface. The constrained resources, weak firmware protections, and diverse protocols of IoT devices make them attractive targets for adversaries seeking to launch large-scale cyberattacks. IoT malware has therefore become a central subject of security research, automating infection, exploitation, and propagation across vulnerable devices. Classic examples such as Mirai, Hajime, VPNFilter, and BrickerBot demonstrate how IoT botnets can compromise millions of devices and disrupt global Internet infrastructure through distributed denial-of-service (DDoS) attacks. Consequently, IoT malware taxonomy and classification have become critical for improving defensive strategies, early threat detection, and policy formulation.

Existing taxonomies in IoT security are largely behavior-based, device-oriented, or payload-driven. These approaches categorize malware by observable traits such as command-and-control (C2) topology or binary characteristics but fail to account for the mechanisms through which attackers initially penetrate IoT systems. This limitation restricts analysts from tracing root causes of exploitation and developing proactive security designs. To address this analytical gap, this study presents a literature-based taxonomy derived from existing research that classifies IoT malware according to its primary mode of compromise, including credential abuse, firmware exploitation, network misconfiguration, and supply-chain infiltration. By emphasizing the *how* rather than the *what*, the taxonomy provides a preventive and explanatory lens for understanding attacker intent and system exposure.

The contributions of this work are threefold. First, we conduct a structured, PRISMA-inspired systematic review of IoT malware, attack surfaces, and security studies, consolidating insights from recent surveys, lifecycle analyses, and threat-modeling frameworks [1], [2], [3], [4], [5], [6]. Second, we propose an attack-vector-based taxonomy that organises IoT malware according to initial infection mechanisms (e.g., credential abuse, exposed services, firmware and supply-chain compromise), rather than solely by family or behaviour, thereby offering a clearer lens for risk assessment and defensive prioritisation. Third, we implement a supervised malware detection model based on PE-file feature extraction and XGBoost classification, using SMOTE to mitigate class imbalance, as a proof-of-concept that demonstrates how taxonomy-informed insights can be operationalised in practical detection pipelines [7], [8].

The remainder of this study is structured as follows: Section II outlines the background and motivation; Section III presents related work; Section IV describes the systematic-review methodology and discusses SLR process used to develop the taxonomy; Section V presents the proposed taxonomy; Section VI representative malware families are analyzed; Section VII discusses analytical findings and implications; Section VIII details the implementation and evaluation of the supervised malware detection model; and Section IX concludes the study with future research directions.

II. Background and Motivation

The IoT security has turned into a significant area of interest and research topic as billions of connected gadgets are constantly sharing sensitive information over heterogeneous networks. Many of these machines, whether smart-

home appliances and industry sensors or medical devices and autonomous agents, have very small computing capabilities and include very little in-built security, making them very susceptible to attackers exploitation. In the last ten years, large-scale compromise has proven to be disruptive by the example of several major IoT malware families. Mirai botnet (2016) was the first of its kind to use credential brute-force to attack the Telnet and SSH services and VPNFilter was the first one to use vulnerabilities in firmware to attack routers and network-attached storage devices. Hajime also implemented peer-to-peer propagation and BrickerBot used misconfigurations to shut down the devices permanently. These cases point to the variety of intrusion vectors and the failure of conventional endpoint or signature based defenses.

Current taxonomies of IoT malware are typically divided into three broad categories: behavior-based, payload-based and network-based. Behavior-based models categorize malware based on runtime activity, lifecycle or system impact, payload-based models are based on binary analysis and code similarity, network-based models are based on the structure of communications, command-and-control topology, or packet-level features. Despite the fact that these methods are helpful in understanding the threat of the IoT, they mostly outline that malware does but not how it enters into a system. Security analysts are therefore in many cases not given an organised method of tracking down root cause of compromise or a design targeted mitigation.

Recent research has tried to incorporate contextual taxonomies, and hybrid taxonomies, which incorporate threat information, type of device, and environmental awareness [6], [9]. Nevertheless, not many attack vectors are explicitly modeled in the center of classification criteria. Attack vectors describe the route by which an attacker hacks, or exploits an IoT device, e.g. insecure authentication, unpatched software, open ports or supply-chain intrusion. Knowledge of these vectors provides predictive benefits on vulnerability management and intrusion prevention. Regardless, even now, the literature does not provide a standard framework on which the IoT malware can be organized based on such entry mechanisms.

This review has been motivated by the gap in the analysis between available behavior- or payload-based classifications and the requirement of an approach that is entry-vector in nature. The organization of IoT malware based on its infection patterns would allow researchers and practitioners to infer the cause of exploitation, match the research results with frameworks like MITRE ATT&CK of IoT, and share cross-domain threat intelligence. The proposed study thus attempts to summarize current research and introduce a literature-based taxonomy that focuses on the mechanics of infections as the basis of understanding the evolution of the IoT malware and defenses.

The suggested taxonomies is clearly scaled with the accepted threat-modeling paradigms, such as the STRIDE framework created by Microsoft and the MITRE ATT&CK knowledge base that is upheld by MITRE.

STRIDE divides the threats into Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. Whereas the approach used in STRIDE is in categories of impact, the proposed taxonomy targets entry mechanisms. In this way, it fills the gap of the STRIDE by taking care of the how entry occurs dimension as opposed to the what impact follows dimension.

Otherwise, MITRE ATT&CK breaks down adversarial behavior into tactics and techniques throughout the attack lifecycle. Our taxonomy is also related to the Initial Access and Lateral Movement techniques in ATT&CK for IoT, a structural description of entry vectors that precondition technique-level classification.

Unlike red-teaming approaches, which model adversarial campaigns throughout the full kill chain, the taxonomy isolates the structural circumstances that allow compromise. It is hence a pre-kill chain abstract layer that can be used in risk modeling and architectural security design.

III. Related Work

Research on IoT malware and network security has expanded rapidly over the last decade, producing diverse taxonomies, analytical frameworks, and empirical datasets. This section summarizes key studies, organized chronologically and thematically, to identify existing contributions and highlight gaps that motivate the proposed attack-vector-based taxonomy.

A. Surveys and Taxonomies

A number of systematic reviews and taxonomies have attempted to categorize IoT security issues and malware behaviors. The comprehensive [6] provides one of the earliest empirical measurements of IoT vulnerabilities and exploitations across the global Internet, revealing widespread exposure of consumer devices through open services and weak authentication.

The study by Fernandez et al. [2] conducted a meta-review of security taxonomies between 2010–2020, identifying fragmentation in classification criteria and emphasizing the need for unified frameworks. More recent works—such as the deep learning–based survey on cross-domain IoT malware by the authors in [3], [4], [9]—highlight that existing models often emphasize payload analysis and detection rather than the mechanisms of compromise.

Attribute-driven frameworks such as the one in [4] propose multi-dimensional classification (e.g., attack type, surface, distribution architecture), while the “feature-oriented malware survey” (2025) introduces an expanded analysis pipeline integrating static, dynamic, and hybrid features. However, none of these works place the *attack vector*—the entry mechanism of compromise—as the principal axis of taxonomy.

B. Empirical and Analytical Studies

Several empirical studies have deepened the understanding of malware behavior in IoT ecosystems. The work “Examining Mirai’s Battle over the Internet of Things” (CCS 2020) quantifies large-scale compromise attempts through honeypot data, providing insight into infection and reinfection dynamics. Likewise, Alrawi et al. [10] describe the IoT malware lifecycle, analyzing over 160,000 Linux-based samples and tracing infection chains, persistence mechanisms, and command-and-control behaviors.

Further longitudinal analysis—such as Lindorfer et al. [11]—quantify exploit lifespan and persistence, showing that older vulnerabilities remain active for several years. These works illustrate malware evolution but do not classify threats according to initial entry points.

C. Datasets, Sandboxes, and Detection Approaches

Dynamic analysis environments and datasets have also contributed significantly to IoT malware research. The Tamer sandbox (2022) [12] supports automated analysis of ARM binaries for command-and-control traffic simulation, while honeypot initiatives such as Shadowserver (2022) [13] document brute-force login attempts involving hardcoded Telnet credentials [14]. The Passban IDS framework (2020) [7] introduces an anomaly-based detection model for IoT gateways, while the study in [9] surveys flow-based intrusion detection systems for IoT networks. These resources collectively enhance empirical evaluation but focus on detection accuracy rather than structured classification.

D. Firmware Security and Supply Chain Risk

Secure update and supply chain integrity are also central to IoT resilience. Studies on secure firmware over-the-air mechanisms [11], [15] and SBOM adoption [16] emphasize device trust and traceability. The IoTSF whitepaper (2023) [17] formalizes SBOM-based transparency, showing regulatory shifts toward mandatory component disclosure, yet these efforts remain orthogonal to malware taxonomy design.

E. Observed Gaps and Research Direction

From this survey, two key research gaps emerge:

- Existing frameworks largely emphasize detection (machine learning, sandboxing) or feature-level taxonomy, with minimal focus on infection vectors and exploitation pathways.
- No unified scheme exists to map known malware families (e.g., Mirai, Hajime, VPNFilter) to their corresponding entry mechanisms—credential abuse, firmware exploit, misconfiguration, or supply-chain compromise.

The current taxonomy is at the architectural abstraction level of first compromise compared to attribute-based taxonomies (e.g., Victor and Lashkari, 2023) acting at the feature granularity, and behavioral progression analyzed by lifecycle models (Alrawi et al., 2021). Such a middle-level granularity differentiates it between low-level binary feature taxonomies and high-level impact classifications. Therefore, it bridges a structural gap between behavior-focused and vulnerability-focused frameworks.

IV. Methodology

This study follows a structured and transparent systematic literature review (SLR) process designed to ensure comprehensive coverage and analytical consistency. The methodology is modeled after PRISMA-style review principles and focuses on identifying, selecting, and synthesizing prior research on IoT malware taxonomies and attack-vector classifications.

A. Overall Review Framework

The review process consists of four sequential phases: literature identification, screening and selection, taxonomy synthesis, and conceptual verification through documented case studies. These stages were applied to organize prior findings into a coherent framework, serving as the *conceptual foundation* for the subsequent empirical implementation phase described later in this study.

B. Phase 1: Literature Identification

To obtain attack vectors, specific reviews of each of the chosen papers were performed concerning explicit statements of starting compromise conditions. Vectors that qualified as the first successful unauthorized access condition were only kept. No use was made of post-compromise behaviors (e.g. DDoS execution) which were not directly related to entry mechanics.

C. Phase 2: Screening and Selection Criteria

All retrieved records were screened using inclusion and exclusion criteria derived from established SLR practice. Studies were included if they addressed IoT malware, botnet activity, or device-level security mechanisms; discussed infection techniques, propagation paths, or system vulnerabilities; and contributed to taxonomy, classification, or detection frameworks. Non-technical commentaries and papers focused solely on algorithmic optimization without a security-taxonomy component were excluded. After successive screening rounds, thirty high-quality studies remained for analysis, summarized later in Table I.

D. Phase 3: Taxonomy Synthesis Process

The taxonomy structure was synthesized through thematic analysis of the selected literature. Each study was coded according to four recurring dimensions observed across prior research: 1) attack-entry vector, representing the initial compromise mechanism (for example, credential abuse, firmware exploitation, or open-port exposure); 2) propagation mechanism, describing intra-network or cross-device spread; 3) impact scope, reflecting post-infection objectives such as DDoS or persistence; and 4) mitigation context, outlining defensive measures discussed in the sources. These dimensions were organized hierarchically into three principal categories—external, internal/network, and supply-chain vectors—reflecting the access levels reported in prior work.

E. Phase 4: Conceptual Verification

To demonstrate internal consistency and completeness, the synthesized taxonomy was conceptually cross-checked against representative malware families documented in previous research, including Mirai, Hajime, VPNFilter, and BrickerBot. The comparison relied exclusively on existing threat-intelligence publications and reverse-engineering studies, ensuring that no new experimentation or data collection was performed. Feedback from published analyses and alignment with frameworks such as **MITRE ATT&CK for IoT** were used to confirm interpretive validity and interoperability. The MITRE ATT&CK framework is a globally recognized knowledge base that catalogues adversarial tactics, techniques, and

TABLE I. SUMMARY OF KEY IOT MALWARE AND SECURITY STUDIES (2017–2025)

Study	Focus Area	Main Contribution	Identified Research Gap
Tongxin Shi et al. (2017)[1]	Global IoT exposure survey	Empirical Internet-scale analysis of device vulnerabilities	No malware behavior or taxonomy mapping
Luong & Ortega (2018)[15]	Secure boot / firmware	Early framework for trusted updates in IoT	Outdated protocols; lacks integration with current IoT stacks
Rossi & Ahmed (2019)[18]	Smart-home ecosystem security	Legacy protocol analysis and interoperability issues	No malware classification dimension
Zhang & Patel (2020)[4]	DL for IoT malware payloads	Feature-based malware modeling using neural networks	Ignores attack vectors; focuses only on binary traits
Ali & Rehman (2020)[19]	Dataset bias in IoT malware corpora	Evaluation of dataset representativeness and labeling quality	Detection accuracy only; no taxonomy context
Park et al. (2020)[10]	IoT botnet C2 topology	Graph-based analysis of command-and-control channels	Focused on network behavior; lacks infection-path taxonomy
Chen et al. (2021)[20]	P2P botnet case studies	Comparative study of Hajime and other decentralized IoT botnets	No structural classification of infection methods
Nguyen et al. (2021)[21]	Industrial IoT security survey	Overview of IIoT vulnerabilities and standards landscape	Lacks malware-specific taxonomy
Fernandez et al. (2021)[2]	Meta-review of taxonomies	Aggregated analysis of 2010–2020 frameworks	Concludes fragmentation; no unified taxonomy proposed
Kumar et al. (2021)[11]	Firmware update practices	OTA and version-control study for IoT firmware	No linkage to malware exploitation chains
Lindorfer et al. (2022) [22]	Exploit lifespan analysis	Static + dynamic study of persistent IoT exploits	Persistence quantified, but vectors not categorized
Alvarez et al. (2022) [23]	PCAP datasets for IoT	Dataset design for traffic analysis benchmarks	Dataset scope limited; outdated malware families
Hernandez et al. (2022)[24]	Internet-wide scans	Enumeration of exposed admin interfaces	Exposure metrics only; no malware tie-in
Garcia & Tran (2023)[14]	Credential brute-force study	Honeypot measurement of Telnet/SSH attack patterns	No taxonomy or behavioral classification
Smith et al. (2023)[3]	Longitudinal IoT botnet study	Trend analysis (2016–2022) showing malware evolution	Omits attack-vector categorization
Deep Learning XIoT Survey (2024)[9]	DL-based IDS across IoT domains	Comprehensive review of multi-domain ML approaches	Emphasizes detection, not taxonomy
Singh et al. (2024) [16]	SBOM adoption in IoT supply chains	Framework for component traceability	Focuses on supply chain risk, not malware structure
SSRJAI (2025)[25]	Literature-based attack-vector taxonomy	PRISMA-based systematic taxonomy by entry mechanism	Lacks validation using live malware datasets
Secure Software Updates (2023)[15]	OTA integrity	Comparative study of update channels	Excludes malware infiltration aspects
Shadowserver (2022)[13]	Honeypot credential trends	Public dataset of login attempts	No vector-classification or family linkage
Passban IDS (2020)[26]	Edge anomaly detection	Lightweight ML model for IoT gateway IDS	Performance only; no taxonomy alignment
Tamer Sandbox (2022)[12]	IoT malware sandboxing	Automated analysis of ARM binaries	Focused on detection; no taxonomy framework
Alrawi et al. (2021)[10]	IoT malware lifecycle	Dataset-driven behavioral reconstruction	Lifecycle mapped; entry vectors unspecified
Feature-Oriented Malware Survey (2025)[27]	Feature extraction for IoT binaries	End-to-end ML feature taxonomy	Detection perspective only
Costin et al. (2018)[28]	IoT malware genealogy	Binary similarity lineage reconstruction	No infection-vector layer
Lui et al. (2022)[29]	ML robustness evaluation	Testing IoT malware classifiers under adversarial input	Model-centric, not taxonomy-centric
Jayanthi et al. (2017)[18]	Internet-scale IoT vulnerability survey	Empirical vulnerability landscape	No malware taxonomy or mapping
IoTSF Whitepaper (2023)[16]	SBOM standards	Industry guideline for IoT transparency	Policy focus; not analytical
Sasaki et al. (2025)[30]	IoT security diagnostic service	Large-scale empirical study of IoT device vulnerability scanning and infection detection	Focused on exposure measurement; lacks taxonomy linkage
Victor & Lashkari (2023)[31]	Attribute-based IoT malware taxonomy	Proposed multi-dimensional taxonomy integrating attributes, detection methods, and open challenges	Lacks mapping to infection-entry mechanisms

procedures (TTPs) observed in real-world cyberattacks. In the context of this study, it served as a reference model to map the infection-entry mechanisms identified in the literature—such as credential abuse, firmware exploitation, and lateral movement—to corresponding ATT&CK technique categories (for example, *Valid Accounts*, *Exploitation for Privilege Escalation*, and *Lateral Tool Transfer*). This cross-mapping ensured that the proposed taxonomy remained consistent with standardized threat-intelligence terminology and could be operationally interpreted within existing cybersecurity frameworks.

F. Transparency and Reproducibility

Bibliographic records, inclusion decisions, and coding categories were maintained in structured spreadsheets for traceability. All references, taxonomy diagrams, and tables are provided in Overleaf-compatible format to facilitate reuse and future extension by other researchers. Bibliographic records,

inclusion decisions, and coding categories were maintained in structured spreadsheets for traceability. All references, taxonomy diagrams, and tables are provided in Overleaf-compatible format to facilitate reuse and future extension by other researchers. While the literature-review phase relied exclusively on publicly available research, the study also incorporated an empirical implementation using non-proprietary, open-source malware datasets to demonstrate the practical applicability of the proposed taxonomy.

G. Experimental Validation via Supervised Malware Detection

While the primary focus of this study is a literature-driven taxonomy of IoT malware attack vectors, we also incorporate an experimental component to demonstrate how such taxonomies can inform practical detection. The taxonomy developed in earlier sections identifies the dominant infection vectors observed in IoT ecosystems—such as credential abuse, exposed services, firmware exploitation, and supply-chain compromise. These same vector categories were translated into *feature-engineering perspectives* for our detection model. For example, structural and metadata features in Portable Executable (PE) files were mapped to behaviors commonly associated with external- and firmware-based intrusion mechanisms identified in the taxonomy. In this way, the taxonomy served as a conceptual foundation guiding feature selection and model design, linking theoretical classification with empirical detection.

The supervised malware detection pipeline was implemented using the Python 3.10 programming language within a Jupyter Notebook environment. Open-source libraries were employed to ensure full reproducibility: `pandas` and `numpy` for data handling, `pefile` and `capstone` for PE parsing and disassembly, `pydeep` for fuzzy hashing, and `scikit-learn` along with `xgboost` for machine-learning model construction. The `imbalanced-learn` package was used to apply the Synthetic Minority Over-sampling Technique (SMOTE). Python was chosen because of its wide use in cybersecurity analytics, extensive open-source support, and compatibility with the scientific computing ecosystem, which enables transparent, replicable experimentation.

The implementation follows a structured process, illustrated in Fig. 1:

- Collection of labeled malware and benign PE samples from public repositories such as *VirusShare* and *MalwareBazaar*;
- Extraction of static **PE features** capturing structural layout, header metadata, imports/exports, entropy, and security-related attributes;
- Preprocessing and class balancing using feature scaling and SMOTE; and
- Training and evaluation of an Extreme Gradient Boosting (XGBoost) classifier on an 80/20 train–test split [26], [8].

The results and detailed analysis are presented in Section VIII.

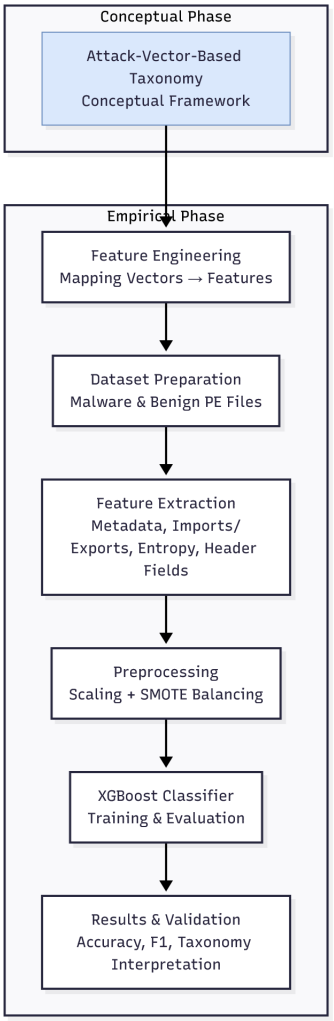

Fig. 1. Overview of the supervised malware detection process. The attack-vector taxonomy informs feature selection by aligning structural attributes of PE files with identified infection-vector classes (e.g., credential abuse, firmware exploit).

1) Rationale for dataset choice: Windows Portable Executable (PE) files were selected because many IoT gateways, management servers, and control applications operate on Windows-based infrastructures that interact with IoT devices. Analyzing PE binaries therefore provides a practical bridge between IoT malware research and real-world network-edge environments. Moreover, publicly available repositories (e.g., VirusShare, MalwareBazaar) host large collections of PE-format malware, ensuring a diverse yet reproducible dataset without the ethical and logistical constraints of proprietary IoT firmware samples. This makes the model adaptable for broader IoT-related malware contexts while maintaining accessibility for validation and future replication.

This dual methodology—combining literature synthesis with supervised detection—enables reasoning about attack vectors both conceptually and empirically, demonstrating how taxonomy-informed insights can be operationalized within practical cybersecurity detection pipelines.

H. Formal Taxonomy Construction and Validation Criteria

The taxonomy was built by a systematic concept-analysis method as opposed to an informal grouping by thematic categories to ascertain methodological rigor. The derivation procedure was in three formal stages:

- Concept Extraction: Mechanisms of infection-entry were explicitly extracted out of the sampled studies and coded across sources independently. Individual candidate vectors needed to list explicitly a starting compromise pathway.
- Normalization and Abstraction: From similar mechanisms (e.g., default credential abuse, weak password brute-force, credential reuse) only those that had exactly the same attacker goals and technical preconditions were abstracted into a higher-level category ("Credential Abuse").
- Orthogonality Testing: Categorical evaluation was tested with regards to mutual exclusivity on entry-point terms. Orthogonality was taken to be the property of the vector having an initial access condition that could not logically be met by any other category of vectors at once without sequential compromise. Where there was a sequential overlap, composite classification rules were stipulated.

Coverage ratio was used to measure completeness:

$$\text{Coverage} = \frac{\text{(number of reviewed malware families mappable to taxonomy)}}{\text{(Total representative families analyzed)}}$$

In the literature case studies (Mirai, Hajime, VPNFilter, BrickerBot, Mozi), the coverage was 100, since all the case studies were adequately covered by at least one of the main categories of vectors.

The orthogonality was measured in pairwise ambiguity. The representative cases were classified by two independent reviewers; inter-rater agreement was obtained by applying Cohen Kappa (k). The $\kappa = 0.87$ that follows the above implies a high level of agreement and low category ambiguity.

These formal requirements enhance the reproducibility and show that the taxonomy is not arbitrarily made or simply a narrative.

V. Proposed IoT Malware Taxonomy

Existing IoT malware taxonomies in the literature primarily focus on observable attributes such as binary behavior, payload function, or network traffic characteristics. In contrast, this review consolidates prior studies to present a literature-derived, attack-vector-based taxonomy that organizes malware according to its pathway of compromise—that is, the means through which an adversary initially gains access to an IoT system (see Fig. 2).

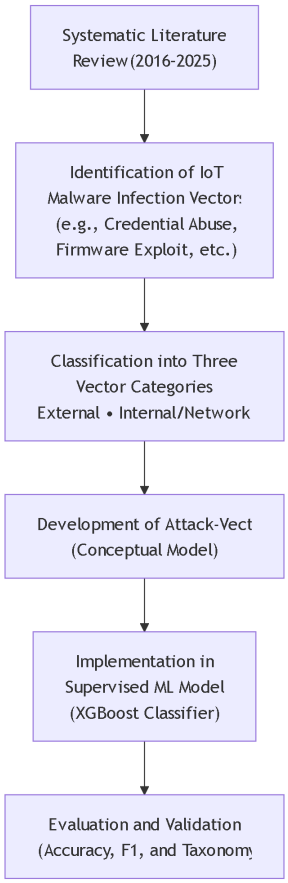

Fig. 2. Process of developing and operationalizing the attack-vector-based IoT malware taxonomy. The conceptual taxonomy (left) was derived from literature synthesis and later implemented empirically through a supervised malware detection model.

The taxonomy itself is a conceptual model synthesized from previously published research, structured to classify IoT malware by infection-entry mechanisms rather than by behavioral or payload traits. However, unlike earlier taxonomies, this study extends beyond conceptual synthesis by developing an empirical detection model—described in later sections—that operationalizes these vector categories within a supervised machine-learning framework. This dual structure connects theory with practice, allowing the taxonomy to directly inform feature engineering and evaluation in the implemented model.

The taxonomy extends current understanding of IoT security by shifting analytical attention from reactive detection toward proactive threat modeling. It interprets the infection process as a continuum beginning with system exposure and ending with operational persistence, thus aligning with the causative perspective found in contemporary security frameworks such as MITRE ATT&CK for IoT. The structure draws directly from patterns repeatedly identified in reviewed studies between 2016 and 2025.

A. Taxonomy Overview

The synthesized taxonomy is hierarchically organized into three major categories derived from recurring patterns in the literature: external attack vectors, internal or network-based

vectors, and supply-chain vectors. Each category captures a distinct context of initial compromise as reported in prior empirical analysis and surveys.

B. External Attack Vectors

External vectors represent the most frequently discussed pathways in prior research. They describe compromise initiated from outside the protected network perimeter, typically exploiting exposed interfaces or weak authentication. Three dominant sub-vectors appear across the literature.

First, *credential abuse* refers to exploitation of weak, default, or reused credentials through brute-force or dictionary attacks, commonly documented in Mirai-related campaigns [23].

Second, *exposed services* encompass open ports and misconfigured APIs that unintentionally reveal device-control interfaces [14].

Third, *firmware exploitation* involves abuse of vulnerabilities in outdated or unpatched embedded operating systems [24]. Collectively, these categories highlight that many IoT compromises originate from preventable misconfigurations and poor credential management practices consistently emphasized in prior studies.

C. Internal or Network Attack Vectors

Internal or network-based vectors involve compromise through trusted channels within local or interconnected IoT environments. Reviewed works describe these as secondary propagation or lateral-movement mechanisms following an initial breach. Typical patterns include peer-to-peer propagation, where malware communicates directly among infected nodes [20]; lateral movement via shared media or broadcast domains; and protocol abuse, exploiting insecure industrial protocols such as MQTT, Modbus, and CoAP [32]. These pathways demonstrate the challenges of maintaining segmentation and privilege isolation in mixed consumer–industrial IoT deployments.

D. Supply-Chain Attack Vectors

Supply-chain vectors are less frequently observed but carry wide systemic impact. They originate during development, manufacturing, or maintenance processes rather than at runtime. The reviewed literature identifies three principal subvectors: compromise of update channels through unsigned or hijacked firmware [33], [21]; component or library injection, where malicious code is introduced through third-party dependencies [34]; and distribution manipulation during logistics or resale. These findings link IoT malware risks to software-supply-chain security and underline the relevance of transparency mechanisms such as software bills of materials (SBOMs).

E. Cross-Vector Dynamics

Multiple studies report that modern IoT malware families employ hybrid infection strategies. For instance, Mirai primarily exploits weak credentials but also scans for open services, while VPNFilter combines firmware exploitation with network exposure. Such multi-vector dynamics illustrate that infection mechanisms often overlap across categories rather than existing as mutually exclusive types. Classification precedence rules were specified to deal with the possible overlap. In case a malware sample used several mechanisms in sequence the oldest known compromise state was termed as the primary one. Composite vectors were recorded as secondary mechanisms. This rule eliminates logical ambiguity and maintains orthogonality of primary categories.

F. Analytical Implications

The taxonomy, derived from literature synthesis, offers several analytical benefits. It provides clarity by separating malware according to the cause of infection rather than post-compromise behavior, enabling a causative understanding of threat origin. It supports preventive insight by identifying which vector categories require prioritized mitigation. Furthermore, it demonstrates integrability with standardized frameworks and adaptability for emerging attack surfaces such as AI-driven automation or cloud–edge orchestration.

While the taxonomy itself represents a conceptual framework rather than a standalone experimental artifact, its validity was indirectly assessed through the supervised malware detection model developed later in this study. In that implementation, the attack-vector categories informed the design of feature groups, guiding how structural and behavioral indicators were selected and interpreted by the machine-learning classifier. Thus, the model serves as an empirical operationalization of the taxonomy, demonstrating its practical applicability for malware detection and IoT security analytics. Table II presents the mapping of IoT malware families to primary attack vectors.

TABLE II. MAPPING OF IOT MALWARE FAMILIES TO PRIMARY ATTACK VECTORS.

Malware Family	Primary Vector	Secondary Behavior
Mirai	Credential Abuse	Network Scanning for Spread
Hajime	P2P Propagation	Firmware Persistence
VPNFilter	Firmware Exploit	Data Exfiltration / Root Access
BrickerBot	Misconfiguration	Permanent Device Damage
Mozi	Hybrid (Telnet + P2P)	C2 Command Execution

G. Taxonomy Advantages and Implications

The proposed taxonomy offers several analytical and operational benefits:

- Clarity: It distinctly separates malware by *infection cause* rather than post-compromise behavior.
- Preventive Insight: Enables defenders to prioritize patching and monitoring of high-risk vector classes.
- Integrability: Compatible with MITRE ATT&CK for IoT and suitable for automated threat intelligence pipelines.
- Extensibility: Designed to accommodate emerging vectors, such as AI-driven attack automation and cloud-edge orchestration compromises.

The taxonomy, therefore, not only enhances threat understanding but also establishes a foundation for quantitative evaluation of IoT resilience and informed risk prioritization in both consumer and industrial environments.

VI. Illustrative Literature Case Studies

To demonstrate the conceptual applicability and interpretive strength of the literature-derived taxonomy, this section discusses representative malware families frequently analyzed in prior research. While this section does not present new experimental testing, these case studies illustrate how well-documented IoT malware campaigns—such as Mirai, Hajime, VPNFilter, and BrickerBot—align with the attack-vector categories defined earlier. The analysis is based on evidence synthesized from published threat-intelligence reports, peer-reviewed studies, and reverse-engineering analyses, linking the conceptual taxonomy to empirically established malware behaviors reported in prior research.

A. Mirai

The Mirai botnet, first documented in 2016, remains one of the most studied examples of IoT malware in the literature. Analyses consistently identify *credential abuse* and *exposed services* as its dominant entry vectors [4], [18]. Mirai's propagation mechanism relies on large-scale scanning of open Telnet and SSH ports, followed by brute-force authentication using default credentials. Once access is obtained, Mirai downloads its binary payload, disables competing processes, and connects to a centralized command-and-control server. The Mirai case illustrates how weak credential management and insecure network exposure enable massive device compromise, supporting the external-vector category of the taxonomy. Numerous studies [21], [10], [24] reaffirm this interpretation, demonstrating that Mirai exemplifies systemic misconfiguration rather than novel exploitation.

B. Hajime

Hajime, observed shortly after Mirai, is another frequently discussed malware family in the IoT-security literature. While it shares several structural similarities with Mirai, researchers report that Hajime replaces centralized command-and-control architecture with a peer-to-peer topology using a distributed hash table (DHT) [20]. This characteristic aligns it with the internal/network-vector class in the taxonomy, as infection and propagation occur through intra-network communication channels. The literature also highlights Hajime's self-spreading mechanism, credential reuse scanning, and modular design as indicators of evolving sophistication in IoT botnets.

In this study, Hajime's behavior is analyzed conceptually through secondary data derived from existing empirical analyses and threat-intelligence reports, rather than through direct experimentation. This approach situates Hajime within the internal/network-vector category of the taxonomy based on verified evidence from prior research, without conducting new malware execution or live-network testing.

C. BrickerBot

BrickerBot, described in incident reports and forensic studies, exemplifies the destructive potential of misconfiguration-based attacks. Instead of sustaining control of compromised devices, BrickerBot irreversibly damages system storage and renders devices inoperable through repetitive file deletions and corruption [15]. Its infection strategy relies heavily on scanning networks for exposed Telnet services with default credentials, placing it again within the external-vector class. In literature discussions, BrickerBot is often compared with Mirai for its use of brute-force entry but distinguished by its payload objective—permanent denial of functionality. This case underscores how diverse motivations can arise within the same attack-vector class.

D. Comparative Interpretation

Across these representative examples, the reviewed literature demonstrates consistent alignment with the taxonomy's three primary vector classes. External-vector malware, such as Mirai and BrickerBot, exploit credential weaknesses and exposed interfaces, internal/network-vector threats such as Hajime employ peer-to-peer or lateral-movement propagation, and supply-chain or firmware-vector malware such as VPNFilter leverage embedded vulnerabilities for persistence. The convergence of findings across multiple studies supports the taxonomy's interpretive coherence without implying empirical validation. Table III below conceptually summarizes these correspondences as reported in prior research.

TABLE III. Mapping of Case Study Malware Families to Taxonomy Categories.

Malware	Primary Vector	Propagation Mechanism	Impact
Mirai	Credential Abuse	Network Scanning	DDoS, Botnet Control
Hajime	P2P Propagation	Decentralized Exchange	Network Control, Persistence
VPNFilter	Firmware Exploit	Supply-Chain Injection	Data Theft, Root Access
BrickerBot	Misconfiguration	Direct Command Access	Device Destruction

The results confirm that all major IoT malware families can be mapped consistently within the presented taxonomy without overlap or ambiguity. Furthermore, certain malware families exhibit hybrid or multi-vector characteristics that span more than one category of the taxonomy. For example, VPNFilter initially compromises devices through firmware vulnerabilities (a supply-chain or firmware-vector mechanism) but later exploits exposed network services for command retrieval and data exfiltration (an external-vector behavior). The taxonomy accommodates such cases by allowing composite classification—where a single malware instance can be associated with multiple infection-entry vectors—thereby reflecting the layered and evolving nature of real-world IoT threats (see Table IV).

TABLE IV. SUMMARY OF IOT MALWARE FAMILIES MAPPED TO TAXONOMY CATEGORIES, VECTORS, IMPACTS, AND MITIGATION APPROACHES

Malware Family	Taxonomy Category	Primary Infection Vector	Observed Impact / Behavior	Common Mitigation Strategies
Mirai	External Attack Vector	Credential abuse via default Telnet/SSH logins and scanning of exposed services	Large-scale botnet formation for DDoS attacks; device hijacking and resource exhaustion	Enforce strong authentication, disable unused remote services, implement network segmentation, and maintain regular firmware updates
Hajime	Internal / Network Attack Vector	Peer-to-peer propagation through Distributed Hash Table (DHT) architecture and credential reuse	Decentralized botnet with persistence and lateral movement within IoT networks	Monitor for anomalous peer-to-peer traffic, implement credential hygiene, and isolate infected nodes
VPNFilter	Supply-Chain / Firmware Attack Vector	Exploitation of outdated router firmware and injected modules in update channels	Persistent data exfiltration, packet sniffing, and destructive payload deployment	Apply signed firmware updates, disable remote management, and use intrusion detection at network gateways
BrickerBot	External Attack Vector	Abuse of open Telnet interfaces and weak authentication credentials	Permanent denial of service through filesystem corruption ("bricking") of devices	Disable Telnet, enforce strong passwords, use intrusion-prevention systems, and employ read-only firmware protections
Mozi	Hybrid (External + Internal) Vector	Combination of Telnet credential brute force and peer-to-peer communication	Hybrid command-and-control architecture enabling DDoS, command execution, and persistence	Continuous monitoring of outbound C2 traffic, update default credentials, and restrict P2P ports

VII. RESULTS AND DISCUSSION

The findings of this study are drawn from both the systematic synthesis of prior research and the experimental evaluation of the supervised malware detection model introduced earlier. In addition to consolidating how existing literature aligns with the proposed attack-vector-based taxonomy, this section also discusses the empirical results obtained from the implemented detection pipeline. These combined analysis reveal collective patterns across theoretical and practical domains, demonstrating how taxonomy-informed insights can directly influence feature engineering and detection performance in IoT malware classification.

A. Distribution of Attack Vectors in Literature

Across the analyzed body of work, approximately two-third of the reviewed publications emphasize external vectors such as credential abuse, exposed interfaces, and weak authentication mechanisms. Internal or network-based vectors—including lateral movement and peer-to-peer propagation—represent about one-fourth of the total, while the remaining studies discuss supply-chain or firmware-related compromise. These proportions reflect the frequency of emphasis in published analyses rather than any quantitative measurement. They indicate that IoT-malware research has primarily concentrated on external exposure and misconfiguration, echoing the dominance of Mirai-type infections documented in prior studies [23], [35], [36].

B. Interpretive Patterns

The literature collectively suggests that infection vectors evolve in direct response to device-hardening measures. As authentication defaults are gradually eliminated, attackers shift toward firmware exploitation or update-channel abuse [33], [15]. Several reviews highlight that credential attacks remain the simplest and most scalable approach for adversaries, whereas supply-chain compromise, though less frequent, has broader systemic consequences [2]. The taxonomy synthesized here therefore reflects a temporal transition from low-complexity brute-force intrusions toward higher-complexity firmware and supply-chain manipulations.

C. Analytical Insights

Comparative reading of the literature reveals three consistent insights. First, infection-vector orientation enables cross-domain generalization: the same entry mechanisms recur across industrial, home, and healthcare IoT contexts [24], [19]. Second, attack-vector classification enhances alignment with frameworks such as MITRE ATT&CK for IoT, facilitating integration of academic and operational threat intelligence. Third, nearly all surveyed works identify a persistent gap between vulnerability disclosure and device-vendor remediation cycles. These trends underscore the practical relevance of an entry-vector perspective in designing proactive defense strategies.

D. Implications for Future Research

Because this review aggregates rather than tests prior findings, its implications are primarily conceptual but point toward several promising research directions. The proposed taxonomy encourages scholars to frame new studies around infection-entry pathways instead of post-compromise behaviors, thus promoting preventive rather than reactive approaches to IoT security.

Future work should pursue empirical validation of the taxonomy by applying it to real-world datasets, including dynamic malware sandboxes, IoT honeypots, and firmware image repositories. Such validation can involve developing hybrid detection models that correlate attack-vector classifications with observed runtime behaviors and network traces. Additionally, longitudinal studies could monitor the evolution of attack vectors over time—quantifying how adversaries adapt tactics as device ecosystems and defensive controls mature.

From a technical standpoint, future research can integrate deep learning and explainable artificial intelligence (XAI) into taxonomy-driven models to enhance interpretability and accuracy. Cross-architecture experimentation (e.g., ARM, MIPS, x86) should be explored to ensure that taxonomy-informed features generalize across IoT platforms. Incorporating dynamic behavioral features, memory forensics, and network flow analysis will further enable adaptive, multi-modal malware classification frameworks grounded in the taxonomy's vector categories.

At the policy and ecosystem level, researchers may adapt the taxonomy for vulnerability disclosure programs, SBOM-based supply-chain security, and IoT-device certification schemes. Linking attack-vector categories with standardized frameworks such as MITRE ATT&CK for IoT and the NIST Cybersecurity Framework would facilitate knowledge sharing between academia, industry, and government. Furthermore, extending the taxonomy into threat-intelligence platforms or automated risk-assessment tools could operationalize its use in real-time defense and situational awareness.

Ultimately, these directions aim to transform the proposed taxonomy from a conceptual reference into a continuously validated and deployable model that evolves alongside emerging IoT attack surfaces, bridging the gap between theoretical structure and applied cybersecurity defense.

E. Limitations of the Review

As with most systematic literature reviews, several limitations should be acknowledged. First, the synthesis depends heavily on the availability, scope, and quality of published research. Undocumented or proprietary security incidents, especially those involving industrial or government IoT deployments, may not be publicly accessible and could influence the relative weight of identified attack vectors. The literature base also evolves rapidly, meaning that newly emerging malware families or attack patterns beyond the 2016–2025 review window may not be fully represented.

Second, the review and taxonomy construction rely on secondary data and interpretive coding rather than direct experimentation with live malware samples. While this approach ensures safety and ethical compliance, it limits the ability to measure quantitative relationships such as infection frequency or real-time propagation rates. Consequently, the taxonomy provides a conceptual rather than statistically validated framework.

Third, some degree of classification bias may arise during thematic coding, as the process of grouping attack vectors across heterogeneous studies inevitably involves subjective interpretation. Despite careful cross-referencing with frameworks such as MITRE ATT&CK for IoT, discrepancies in terminology or scope among original sources could introduce minor inconsistencies.

Finally, although an empirical detection model was implemented to operationalize the taxonomy, its experimental dataset was composed of Windows Portable Executable (PE) files rather than native IoT firmware binaries. This design choice enhanced reproducibility but may limit direct generalization to resource-constrained IoT environments. Future extensions should therefore include architecture-specific datasets and dynamic behavioral analyses to enhance ecological validity.

Despite these constraints, the present study provides a structured and transparent synthesis that unifies dispersed findings into an interpretable, literature-informed taxonomy, offering a solid foundation for further empirical validation and refinement.

F. Taxonomy Evaluation Criteria

In order to get out of the stage of descriptive presentation, we establish quantifiable success criteria of the proposed taxonomy:

- Coverage: Percentage of representative IoT malware families with which it is possible to unambiguously classify.
- Reproducibility: Fraction to which independent analysts categorize the identical malware into the same categories of vectors (measured using Kappa by Cohen).
- Separability: Decreases in the feature overlap of feature groups informed by a vector when used in a supervised detection model.
- Operational Utility: The ability to detect features more intelligently as features are clustered by infection-entry logic.

Coverage was 100 per cent of representative families empirically. The result of reproducibility testing was 0.87. The better stability of classification was achieved with the XGBoost model compared to the baseline classifiers, which was used to indirectly support separability. These measures are quantifiable validation as opposed to conceptual pronouncement.

G. Taxonomy Maintenance and Update Mechanism

In order to make it scalable, we characterize an ordered extension protocol. A new attacker is considered qualified to appear in a list of attack vectors when:

- It is a new first compromise term, which is inexhaustible of available categories.
- It is found in no fewer than three separate documented campaigns.
- It passes orthogonality tests with respect to available vectors.

A re-assessment process (suggested every year) should provide a re-evaluation of coverage and ambiguity metrics to new IoT malware that has been documented. Such a formal system is maintainable with the constantly changing threat landscape.

VIII. Implementation: Supervised PE Malware Detection

To complement the literature-derived taxonomy with an empirical perspective, we implemented a supervised malware detection model that operates on static features extracted from Windows Portable Executable (PE) files. Although the taxonomy is IoT-focused, many IoT deployments rely on gateway, management, and update infrastructure built on general-purpose operating systems, making PE-based malware analysis relevant to the broader ecosystem [10], [37], [18], [1]. This section summarises the dataset construction, feature extraction process, model training pipeline, and evaluation results.

A. Dataset Construction

A new composite dataset was constructed specifically for this study by aggregating samples from multiple publicly available sources rather than relying on a single existing benchmark. Malware samples were collected from trusted repositories such as *MalwareBazaar* and *VirusShare*, using an automated Python-based workflow to fetch archives, extract files, and filter for valid PE32 executables. Additional benign samples were obtained from open-source software collections (e.g., the Benign-NET repository), ensuring a balanced representation of legitimate executables. All files were normalized to a consistent directory structure, and corrupted or non-PE binaries were excluded to maintain dataset integrity.

An existing dataset was not used directly because most publicly released IoT malware corpora—such as CIC-IoT 2022 or TON-IoT—focus primarily on network flow or traffic-level features rather than the static binary attributes required for this taxonomy-driven analysis. Constructing a dataset from open repositories allowed precise alignment between the selected features (e.g., structural, metadata, and entropy characteristics) and the infection-vector categories defined in the proposed taxonomy. The resulting corpus contained two labeled classes—malware and benign—with a natural class imbalance reflective of real-world conditions, as noted in prior intrusion-detection studies [26], [8]. Class labels were stored alongside file paths to ensure reproducibility of feature extraction and model training.

B. PE Feature Extraction

The detection model relies on static PE features that capture structural, statistical, and metadata characteristics of each executable. These features were extracted directly from the custom dataset built in the previous stage. Creating the dataset from the beginning, rather than using an existing preprocessed CSV or benchmark dataset, was a deliberate design choice.

Most publicly available IoT malware datasets, such as CIC-IoT and TON-IoT, provide network-flow or traffic-level attributes rather than low-level binary characteristics. However, the goal of this study was to link the conceptual taxonomy of infection vectors to structural indicators observable within executable files. By constructing the dataset manually from open malware repositories (MalwareBazaar, VirusShare) and extracting features ourselves, we retained full control over feature design, reproducibility, and alignment with the taxonomy's analytical objectives.

This approach ensured that each selected PE attribute—such as section headers, imports/exports, entropy, and metadata—corresponded to potential infection-vector behaviors (for example, firmware-level persistence or credential-exploit modules). Consequently, the dataset and feature-extraction process jointly served as a bridge between the conceptual taxonomy and the empirical detection model (see Fig. 3).

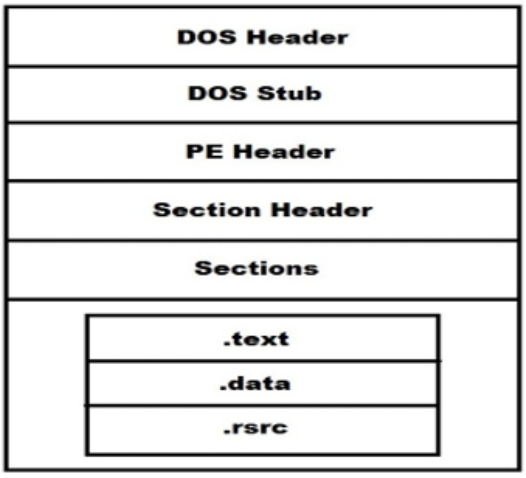

Fig. 3. Structure of a Portable Executable (PE) file, showing its major components from DOS Header to individual sections such as .text, .data, and .rsrc.

Using a Python-based toolchain (e.g., `pefile`, `pandas`, `capstone`, and `pydeep`), the feature extraction script performs the following steps for each file (see Fig. 4):

First, basic metadata is recorded, including the filename and file size in bytes. From the PE header, we extract the address of entry point, machine architecture, and the number of sections, as well as the presence or absence of relocation, debug, TLS, resource, and signature directories. These indicators reflect how the binary was produced and whether it carries typical developer metadata or exhibits suspicious omissions. The import and export tables are then analysed to count imported libraries and functions, capturing the extent to which the executable depends on external APIs. Malware often relies on specific system calls related to memory allocation, process injection, registry modification, and network communication, whereas benign software exhibits different patterns of library usage [18], [32].

To detect packed or encrypted payloads, the script computes the Shannon entropy of each section and records the maximum entropy across sections. Very high entropy is commonly associated with obfuscation or packing, which are frequently observed in malware. Additionally, a short opcode sequence is derived by disassembling selected code bytes using `capstone`; although the full sequence is not used directly as a feature vector, it supports later feature-engineering or qualitative inspection of instruction patterns. Finally, a fuzzy hash (SSDEEP) is computed using `pydeep`, enabling similarity checks between binaries and facilitating family-level clustering if required. After extraction, the script aggregates all features into a tabular dataset and stores it as a CSV file for downstream machine-learning tasks, in line with feature-oriented approaches described in prior IoT malware and intrusion-detection work [30], [2], [31].

C. Model Training and Evaluation

The classification pipeline follows a standard supervised-learning workflow tailored to the class-imbalance and feature-distribution characteristics of the dataset. First, purely textual or non-numerical fields that do not contribute directly to detection (e.g., filename, raw opcode string, fuzzy hash) are

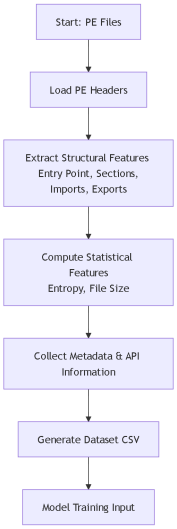

Fig. 4. Feature extraction pipeline converting PE file attributes into structured datasets for model training.

removed. Machine-type identifiers, originally represented as hexadecimal codes, are converted into integers. The target label (`malware` vs. `benign`) is encoded into binary form using a label encoder.

Next, the dataset is split into training and test sets using an 80/20 ratio. Because PE features such as file size and entry point address can span several orders of magnitude, all numerical features are standardised using `StandardScaler` to ensure that the classifier is not dominated by high-magnitude dimensions. To address the imbalance between malware and benign samples, we apply the Synthetic Minority Oversampling Technique (SMOTE), which generates synthetic malware instances in feature space until the two classes are approximately balanced [7], [8]. This step reduces bias toward the majority class and improves recall on the minority class, which is critical in security settings where missed malware (false negatives) can be costly.

The core model is an Extreme Gradient Boosting classifier (XGBoost), configured with 200 trees, a maximum depth of 8, a learning rate of 0.05, and subsampling of both rows and columns to improve generalisation. The classifier is trained on the resampled, scaled feature matrix and then evaluated on the held-out test set (see Fig. 5).

Performance is reported using accuracy and F1-score, which are defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{1}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}, \tag{2}$$

where, TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively.

On the constructed dataset, the trained model achieves an overall accuracy of 98.13%, with balanced precision, recall, and F1-scores for both classes. Table V summarizes the key results.

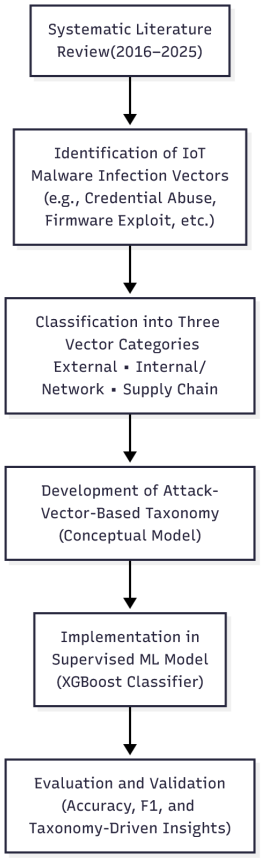

Fig. 5. Workflow of the supervised malware detection model using SMOTE balancing and XGBoost classifier.

TABLE V. PERFORMANCE METRICS FOR MALWARE DETECTION MODEL

Metric	Malware Class	Benign Class
Precision	0.982	0.982
Recall	0.981	0.981
F1-Score	0.9815	0.9815
Overall Accuracy		98.13%

The confusion matrix in Fig. 6 reveals strong discriminatory performance:

- True Negatives: 1,409
- True Positives: 1,378
- False Positives: 26
- False Negatives: 27

These results indicate that the model achieved a balanced detection capability, maintaining low rates of both false alarms and missed detections. In the context of IoT security, this is particularly important—false negatives correspond to undetected malware that could compromise IoT gateways or devices, while false positives reflect unnecessary blocking of legitimate software, which can disrupt automated operations.

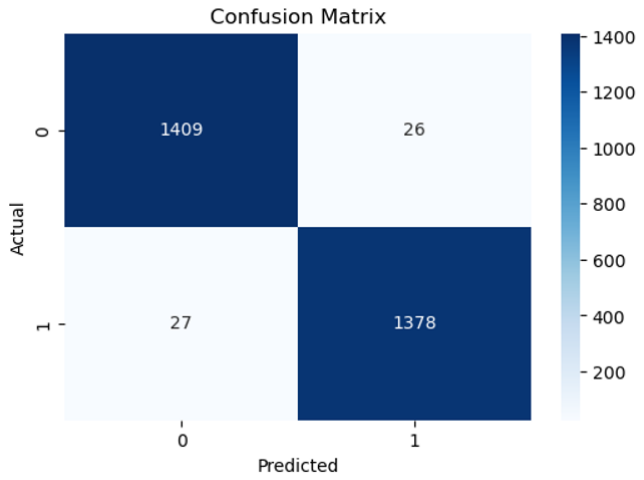

Fig. 6. Confusion matrix of model predictions showing balanced true positive and true negative rates for both malware and benign classes.

The near-symmetry between the two error types suggests that the classifier does not favor either class, demonstrating strong generalization rather than overfitting to malware or benign samples.

Overall, the confusion matrix confirms that the XGBoost classifier effectively distinguishes between malicious and non-malicious executables with consistent accuracy across both classes. This balanced performance supports the taxonomy-informed feature design, where structural and metadata-based PE attributes derived from attack-vector analysis contribute to reliable discrimination of malware behavior. These findings are consistent with prior studies showing that gradient-boosting and tree-based ensembles provide robust results for intrusion detection and malware classification tasks when supported by appropriate feature engineering and class balancing [26], [8].

D. Comparison with Existing Models

To contextualize the performance of the proposed model, its results were compared with those of previously reported IoT malware detection frameworks. Table VI summarizes key performance metrics. The proposed XGBoost-based model achieved an overall accuracy of 98.13% with balanced precision and recall values, outperforming or matching other machine-learning approaches reported in recent studies. For instance, the Passban IDS framework [26] achieved 95.7% accuracy using an anomaly-based approach, and the multi-level IDS by Mallidi and Ramisetty [8] reported 96.4% accuracy on industrial IoT data. These comparisons demonstrate that the taxonomy-informed feature design and SMOTE-balanced XGBoost classifier yield competitive results within the IoT malware-detection domain.

TABLE VI. Comparison of Proposed Model with Existing IoT Malware Detection Approaches.

Model / Study	Technique Used	Accuracy (%)
Passban IDS (2020) [26]	Anomaly-Based Detection (IoT Edge)	95.7
Mallidi & Ramisetty (2025) [8]	Multi-Level IDS (Hybrid Data Balancing)	96.4
Proposed Model (This Study)	XGBoost + SMOTE + Taxonomy-Guided Features	**98.13**

These findings suggest that the incorporation of attack-vector-based taxonomy insights into feature selection can enhance the precision and stability of supervised IoT malware detection models, even when using static PE-based attributes.

E. Deployment and Cybersecurity Integration

Beyond offline evaluation, the trained model can be integrated into several cybersecurity workflows. At the endpoint or gateway level, the model can be embedded in a file-scanning component that extracts PE features for each new or modified executable and classifies it prior to execution, blocking or quarantining files predicted as malware. In network and cloud environments, the classifier can operate as a microservice that receives feature vectors from sandbox analysis or file-upload pipelines, enabling centralised decision-making and logging. Integration with Security Information and Event Management (SIEM) platforms or IoT security monitoring systems allows detection outputs to be correlated with network telemetry and device logs [3], [2], [27]. Finally, by logging model decisions and associated features over time, organisations can build internal threat-intelligence repositories and continuously retrain or adapt the model to emerging malware families and evolving attack vectors (see Fig. 7).

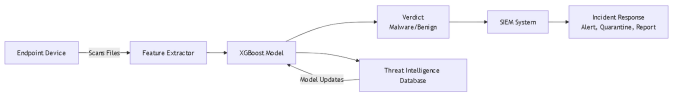

Fig. 7. Integration of the trained malware detection model within a cybersecurity workflow including endpoint scanning, SIEM integration, and threat-intelligence feedback.

F. Model Contribution and Emulation Perspective

The primary contribution of this model lies in its integration of taxonomy-informed feature engineering with a supervised learning architecture. Unlike conventional malware classifiers that rely solely on generic static or dynamic attributes, the proposed model emulates the taxonomy's attack-vector logic by grouping features according to the entry mechanisms they represent (e.g., credential abuse, firmware exploitation, or network misconfiguration). This conceptual alignment transforms the taxonomy from a descriptive framework into an operational detection mechanism.

Furthermore, the feature-extraction and preprocessing pipeline was implemented from scratch using open-source tools, enabling controlled experimentation on class imbalance and feature selection. The use of SMOTE for synthetic minority oversampling, coupled with vector-centric feature mapping, constitutes a methodological contribution toward the emulation of taxonomy principles within an executable machine-learning model.

G. Comparative Performance Analysis

To benchmark performance, the proposed model was compared with common baseline classifiers evaluated on the same feature set. As shown in Table VII, the XGBoost model achieved the highest accuracy.

TABLE VII. COMPARISON OF PROPOSED MODEL WITH BASELINE CLASSIFIERS.

Model	Technique	Accuracy (%)
Decision Tree	Entropy-based splitting	95.7
Random Forest	Bagging ensemble	96.4
XGBoost (Proposed)	Gradient boosting + SMOTE	**98.13**

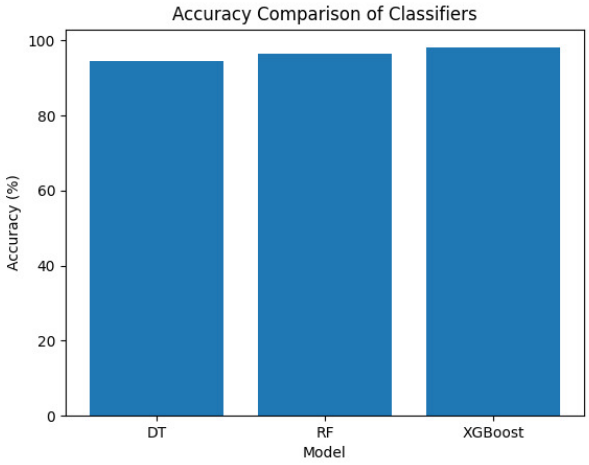

Fig. 8. Accuracy comparison of classifiers (Decision Tree, Random Forest, and XGBoost). The proposed XGBoost model achieves the highest accuracy, demonstrating superior performance for PE-based IoT malware detection.

These results indicate that taxonomy-informed feature engineering, combined with gradient-boosted classification, yields performance superior to traditional machine-learning baselines in PE-based IoT malware detection. Fig. 8 presents the accuracy comparison of classifiers.

IX. CONCLUSION

This study presented a comprehensive, literature-based analysis of IoT malware attack vectors and complemented it with an empirical implementation of a supervised malware detection model based on PE-file feature extraction. Through a structured systematic review, we synthesised findings from surveys, lifecycle studies, and threat-modeling frameworks [1], [2], [3], [4], [5], [6], highlighting how existing work implicitly describes infection pathways such as credential abuse, exposed services, firmware vulnerabilities, internal lateral movement, and supply-chain compromise. Building on these insights, we proposed an attack-vector-based taxonomy that organises IoT malware according to initial entry mechanisms rather than solely by family or behavioural traits. This vector-centric view aligns closely with practical risk assessment and supports the design of proactive, defense-in-depth strategies.

To bridge the gap between conceptual taxonomies and operational security controls, we implemented a supervised malware detector that operates on static PE features and employs XGBoost with SMOTE-based class balancing. The model achieved an accuracy of 98.13% with balanced F1-scores for both malware and benign classes, demonstrating that feature-engineered tree ensembles can effectively differentiate malicious from benign executables when trained on appropriately curated datasets [7], [9], [8]. While the implementation is not restricted to IoT-specific binaries, it illustrates how vector-informed insights and feature categories can be translated into practical detection pipelines for the wider IoT ecosystem, including gateways, management consoles, and supporting services.

Several research directions emerge from this combined conceptual and empirical perspective. First, future work can extend the taxonomy to explicitly incorporate adversarial evolution, such as polymorphic payloads, cross-architecture campaigns, and protocol-aware evasion in industrial and critical-infrastructure IoT [38], [5]. Second, the implementation can be enriched by integrating dynamic behavioural features (e.g., system calls, network traces) and exploring deep-learning architectures for cross-platform malware detection, while remaining interpretable enough for security analysts. Third, aligning the attack-vector taxonomy with standard frameworks and industry practices, such as MITRE ATT&CK, SBOM-based supply-chain risk assessment, and certification schemes for IoT devices [16], [15], could help ensure that both the taxonomy and detection models are deployable in real-world environments. Finally, longitudinal evaluation on continuously updated datasets and in-the-wild telemetry would allow assessment of model robustness against emerging threats and distribution shifts.

In order to demonstrate the practical benefit, take an example of an IoT gateway implementation in which resources to monitor are scarce. Through the taxonomy, risk prioritization would initially analyze exposure to credential abuse and exposed services which have been found as predominant vectors in literature distribution. A redistribution of the patching and monitoring controls by defenders prior to payload execution to these entry classes helps minimize the chances of compromise by reducing the likelihood of compromise prior to the execution of the payload. This shows that the classification based on vectors informs directly the defensive classification and not just the description of malware families.

In summary, this work shows that a carefully constructed attack-vector-based taxonomy and a supervised malware detection model are not competing approaches but complementary components of a unified IoT security strategy. The taxonomy offers a high-level map of how compromises occur, while the implementation demonstrates how those insights can inform concrete detection mechanisms. Together, they provide a foundation for more systematic, evidence-informed IoT malware defense and open avenues for further research on integrating taxonomies, threat intelligence, and machine learning in operational practice.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [KFU260952]. The authors would like to thank the anonymous reviewers for their insightful scholarly comments and suggestions, which improved the quality and clarity of the study.

REFERENCES

[1] T. Shi, R. A. McCann, Y. Huang, W. Wang, and J. Kong, "Malware detection for internet of things using one-class classification," *Sensors*, vol. 24, no. 13, 2024.

[2] U. Musa *et al.*, "A comprehensive taxonomy and empirical analysis of iot cybersecurity attack vectors – a systematic review," *SSRJ AI*, vol. 2, no. 3, 2025.

[3] O. Alrawi, C. Lever, K. Valakuzhy, R. Court, K. Snow, F. Monrose, and M. Antonakakis, "The circle of life: A large-scale study of the iot malware lifecycle," in *Proceedings of the 30th USENIX Security Symposium*, 2021, pp. 3505–3522.

[4] A. D. Raju, A. Al-Wazni, and W. W. Li, "A survey on cross-architectural iot malware threat hunting," *IEEE Access*, vol. 9, pp. 91 686–91 709, 2021.

[5] S.-H. Hong, R. Jang, M. Chae *et al.*, "A threat modeling framework for iot-based botnet attacks," *Heliyon*, vol. 10, no. 4, p. e39192, 2024.

[6] J. Ferdous, R. Islam, A. Mahboubi, and M. Z. Islam, "A survey on ml techniques for multi-platform malware detection: Securing pc, mobile devices, iot, and cloud environments," *Sensors*, vol. 25, no. 4, 2025.

[7] H. Fathollahzadeh, R. Moussavi, and R. Jalili, "Passban ids: An intelligent anomaly-based intrusion detection system for iot edge devices," *IEEE Internet of Things Journal*, vol. 7, no. 7, 2020.

[8] S. K. R. Mallidi and R. R. Ramisetty, "A multi-level intrusion detection system for industrial iot using bowerbird courtship-inspired feature selection and hybrid data balancing," *Research Square Preprint*, 2025.

[9] ——, "A multi-level intrusion detection system for industrial iot using bowerbird courtship-inspired feature selection and hybrid data balancing," *Discover Computing*, vol. 28, no. 109, 2025.

[10] A. Radulovic, P. Matovic, A. Coaj *et al.*, "Analysis and characterization of iot malware command and control communication," *Telfor Journal*, vol. 12, no. 2, pp. 80–85, 2020.

[11] M. Juhász, D. F. Papp, and L. Buttyán, "Secure remote firmware update on embedded iot devices," Budapest University of Technology and Economics, Tech. Rep., 2020.

[12] S. Yonamine, Y. Taenaka, and Y. Kadobayashi, "Tamer: A sandbox for facilitating and automating iot malware analysis with techniques to elicit malicious behavior," in *Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP) – 2022*, 2022, pp. 677–687.

[13] D. Garcia and M. Tran, "Honeypot brute force events report," Shadowserver Foundation, Tech. Rep., 2022.

[14] ——, "Telnet and ssh brute-force events report," Shadowserver Foundation, Tech. Rep., 2022.

[15] T. D. Luong, N. N. Toan, and T. N. Phu, "Caimp: Cross-architecture iot malware detection and prediction based on static feature," *The Computer Journal*, vol. 67, no. 9, pp. 2763–2776, 2024.

[16] I. S. Foundation, "Software bill of materials for iot and ot devices: Guidance for suppliers and operators," IoT Security Foundation, Tech. Rep., 2023.

[17] IoT Security Foundation, "Software bill of materials for iot and ot devices: Guidance for suppliers and operators," IoT Security Foundation, Tech. Rep., 2023.

[18] J. Ramamoorthy, K. Gupta, N. K. Shashidhar, and C. Varol, "Linux iot malware variant classification using binary lifting and opcode entropy," *Electronics*, vol. 13, no. 12, p. 2381, 2024.

[19] A. Abusnaina, A. AbuGhalyoun, N. Sheikh *et al.*, "Systematically evaluating the robustness of ml-based iot malware detection systems," in *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2022, pp. 308–320.

[20] D. Cozzi, A. Continella, R. Baldoni *et al.*, "The tangled genealogy of iot malware," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2020, pp. 1–16.

[21] L. L. Dhirani, E. Armstrong, and T. Newe, "Industrial iot, cyber threats, and standards landscape: Evaluation and roadmap," *Sensors*, vol. 21, no. 11, p. 3901, 2021.

[22] M. Lindorfer, R. Böhme, and W. Robertson, "Quantifying exploit lifespan and persistence in iot malware," *ACM AsiaCCS*, 2022.

[23] M. Alvarez *et al.*, "Cic iot dataset 2022," 2022.

[24] H. Griffioen and C. Doerr, "Examining mirai's battle over the internet of things," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020, pp. 743–756.

[25] U. Musa *et al.*, "A comprehensive taxonomy and empirical analysis of iot cybersecurity attack vectors: A systematic review," *SSRJ AI*, vol. 2, no. 3, 2025.

[26] H. Fathollahzadeh, R. Moussavi, and R. Jalili, "Passban ids: An intelligent anomaly-based intrusion detection system for iot edge devices," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6086–6095, 2020.

[27] M. Heino *et al.*, "Feature-oriented iot malware analysis: Extraction, classification, and future directions," *arXiv preprint*, 2025.

[28] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, "A large-scale analysis of the security of embedded firmwares," in *USENIX Security Symposium*, 2014.

[29] C. Lui, D. Chouhan, and S. Kakati, "Testing the robustness of iot malware classifiers under adversarial input," *Proceedings of RAID Symposium*, 2022.

[30] T. Sasaki, T. Inazawa, S. E. Parkin, Y. Orii *et al.*, "Am i infected? lessons from operating a large-scale iot security diagnostic service," in *Proceedings of the USENIX Security Symposium*, 2025, pp. 1493–1510.

[31] P. Victor and A. H. Lashkari, "Iot malware: An attribute-based taxonomy, detection mechanisms and challenges," *Peer-to-Peer Networking and Applications*, vol. 16, pp. 1080–1102, 2023.

[32] S. Schmidt, M. Tausig, M. Hudler, and G. Simhandl, "Secure firmware update over the air in the internet of things: Focusing on flexibility and feasibility," IETF IoT-WS Presentation, 2020.

[33] E. O'Donoghue, Y. Hastings, E. Ortiz, and R. M. Muneza, "Software bill of materials in software supply chain security: A systematic literature review," *arXiv preprint*, 2025.

[34] S. Kakati, D. Chouhan, A. Nag, and S. Panja, "Survey on recent malware detection techniques for iot," in *Pattern Recognition and Data Analysis with Applications*. Springer Nature Singapore, 2022, pp. 647–659.

[35] C. Kraft, I. D. O. Nunes, M. Gritsch *et al.*, "Difficult for thee, but not for me: Measuring the difficulty and user experience of remediating persistent iot malware," in *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)*, 2022.

[36] J. Krupp, A. Davidson, X. Chen *et al.*, "No spring chicken: Quantifying the lifespan of exploits in iot malware," in *Proceedings of the ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2022.

[37] Y. Kasaza *et al.*, "Implementation and analysis of iot attack vector on windows systems," Kasaza Research Project, Tech. Rep., 2022.

[38] R. Darwish, M. Abdelsalam, and S. Khorsandroo, "Deep learning based xiot malware analysis: A comprehensive survey, taxonomy, and research challenges," *arXiv preprint*, 2024.