Diagnosing Unknown Attacks in Smart Homes Using Abductive Reasoning

Kushal Ramkumar, Wanling Cai, John McCarthy, Gavin Doherty, Bashar Nuseibeh, *Member, IEEE*, and Liliana Pasquale

***Abstract*—Security attacks are rising, as evidenced by the number of reported vulnerabilities. Among them, unknown attacks, including new variants of existing attacks, technical blind spots or previously undiscovered attacks, challenge enduring security. This is due to the limited number of techniques that diagnose these attacks and enable the selection of adequate security controls. In this paper, we propose an automated technique that detects and diagnoses unknown attacks by identifying the class of attack and the violated security requirements, enabling the selection of adequate security controls. Our technique combines anomaly detection to detect unknown attacks with abductive reasoning to diagnose them. We first model the behaviour of the smart home and its requirements as a logic program in Answer Set Programming (ASP). We then apply Z-Score thresholding to the anomaly scores of an Isolation Forest trained using unlabeled data to simulate unknown attack scenarios. Finally, we encode the network anomaly in the logic program and perform abduction by refutation to identify the class of attack and the security requirements that this anomaly may violate. We demonstrate our technique using a smart home scenario, where we detect and diagnose anomalies in network traffic. We evaluate the precision, recall and F1-score of the anomaly detector and the diagnosis technique against 18 attacks from the ground truth labels provided by two datasets, CICIoT2023 and IoT-23. Our experiments show that the anomaly detector effectively identifies anomalies when the network traces are strong indicators of an attack. When provided with sufficient contextual data, the diagnosis logic effectively identifies true anomalies, and reduces the number of false positives reported by anomaly detectors. Finally, we discuss how our technique can support the selection of adequate security controls.**

***Index Terms*—Adaptive security, smart home security, abductive reasoning, anomaly detection.**

Received 6 December 2024; revised 1 September 2025; accepted 10 September 2025. Date of publication 18 September 2025; date of current version 14 November 2025. This work was supported in part by the Science Foundation Ireland under Grant 13/RC/2094 P2 with additional support from UKRI. Recommended for acceptance by P. Pelliccione. *(Corresponding author: Kushal Ramkumar.)*

Kushal Ramkumar and Liliana Pasquale are with Lero—the Science Foundation Ireland Research Centre for Software, University College Dublin, D04 V1W8 Dublin, Ireland (e-mail: kushal.ramkumar@ucdconnect.ie; liliana.pasquale@ucd.ie).

Wanling Cai and Gavin Doherty are with Lero—the Science Foundation Ireland Research Centre for Software, Trinity College Dublin, D02 YY50 Dublin, Ireland, (e-mail: wanling.cai@tcd.ie; gavin.doherty@tcd.ie).

John McCarthy is with Lero—the Science Foundation Ireland Research Centre for Software, University College Cork, T12 CY82 Cork, Ireland, (e-mail: john.mccarthy@ucc.ie).

Bashar Nuseibeh is with The Open University, MK7 6AA Milton Keynes, U.K. (e-mail: bashar.nuseibeh@open.ac.uk).

Digital Object Identifier 10.1109/TSE.2025.3610540

I. Introduction

The number of cyber attacks and vulnerabilities reported is on the rise [1], [2], [3]. Among the different types of attacks, we refer to unknown attacks collectively as novel variants of known attacks, technical blind spots [4], and previously undiscovered attacks (commonly referred to as zero-day attacks). These attacks pose challenges to providing security with a long-term protective outlook (i.e., sustainable security [5]) due to the limited number of automated techniques that diagnose them and identify adequate security controls. Although anomaly detection techniques are effective for detecting unknown attacks [6], they cannot typically reason about them [7]. A recent survey in cybersecurity incident response highlights a gap in the communication and management of attacks, caused by the lack of structured techniques that can reason about and then mitigate the detected attack [8]. Faced with an evolving threat landscape, current threat intelligence solutions also advocate for moving away from using known attack patterns toward identifying actionable indicators of attack behaviours and intent [9]. Thus, security and software engineers who develop security solutions must address the challenge of diagnosing unknown attacks once detected and selecting appropriate security controls to mitigate them.

In this paper, we argue that one of the first steps in providing enduring security requires us to go beyond detecting zero-day attacks by diagnosing these attacks and then identifying appropriate security controls. In our work, diagnosis entails identifying the violated security requirement and the class of attack that the anomaly represents. We provide a novel automated technique combining anomaly detection to detect unknown attacks at runtime with abductive reasoning to generate hypothetical diagnoses of these attacks. Our technique aims to bridge the gap between runtime anomalies and the threat model of the system. The intuition is that even when the attacks are unknown, we can identify security threats and requirements specifying what we would like to protect in the system. Thus, we could determine whether unknown attacks, revealed by anomalous network traffic, violate the security requirements. This, in turn, would enable us to identify security controls that preserve the satisfaction of these requirements.

We choose abductive reasoning for diagnosis due to its ability to map effects to causes. This technique has long been used in diagnosis and explanation generation, mainly when provided with background information about the system under consideration [10]. Among various abductive reasoning techniques, logic-based abduction works well with partial system representations commonly used to represent modern software systems [11].

We first model the system in terms of the permitted actions and its security requirements as a logic program using Answer Set Programming (ASP) [12]. We chose ASP for its expressive language and for its non-monotonic reasoning capabilities that enable abductive reasoning even with partial system models for smart homes that undergo frequent evolution. We then detect anomalies in the network traces captured from the system using an Isolation Forest (iForest) based anomaly detector and encode them in the logic program. Finally, we perform abduction by refutation [11] to identify the violated security requirement, i.e., identify which security requirements prevent a contradiction (anomaly) from existing. We use the Clingo ASP tool to model the system because it has good performance [13] and provides an expressive language to diagnose anomalies. For the anomaly detection, we chose the iForest algorithm since it exhibited the highest F1 score across most attacks within the datasets used for evaluation when compared with other widely recognised anomaly detectors such as One Class SVM and Local Outlier Factor (LOF) [14], [15].

We demonstrate our technique using a smart home scenario, where we aim to detect and diagnose anomalies in network traffic. We chose a smart home due to its dynamic nature, where changes in the devices connected to the network can potentially expand the attack surface. Cyber attacks in the home could pose risks to physical assets and users. For instance, a Denial of Service (DoS) attack against a smart lock would prevent users from locking/unlocking their doors, compromising their physical space [16].

We developed and evaluated our technique using two datasets, CICIoT2023 [17] and IoT-23 [18]. We trained the anomaly detector with unlabelled data to simulate unknown attack scenarios and used the labels only as ground truth for evaluation. We evaluated the anomaly detector's precision, recall and F1-score and the diagnosis technique against 18 attacks from the ground truth labels (i.e., true class labels) provided by two datasets. Our experiments show that the anomaly detector effectively identifies anomalies when the network traces are strong indicators of an attack. It also outperforms other techniques, such as Random Forest, commonly used to detect unknown attacks. When provided with sufficient contextual data, the diagnosis logic effectively identifies true anomalies and reduces the number of false positives reported by anomaly detectors. Combining the modelling of smart homes and their security requirements with the detection and diagnosis of unknown attacks allows us to support security/software engineers in selecting security controls that could mitigate the occurrence of unknown attacks. We also benchmarked our method against prevalent Explainable AI (XAI) approaches which are widely utilised for explaining network intrusions, and malware, phishing, and botnet attacks [19].

Our main contributions in this paper are:

- A novel end-to-end technique that detects unknown attacks in a smart home and diagnoses which security requirements they violate using abductive reasoning, effectively bridging the gap between runtime traces with the threat model of the system.
- An evaluation of the technique using the CICIoT2023 and IoT-23 datasets which together contain a comprehensive collection of smart home attacks against real and simulated devices. We made the ASP system model and a representation of all anomalies generated during our experiments publicly available to support future research.
- A benchmark comparison against XAI techniques showcasing that traditional XAI approaches explain the inferences made by the anomaly detector, but do not identify the security requirements that would be violated as a result of an anomaly like our technique does.

This paper is organised as follows. In Section II, we survey existing literature and relevant works. Section III provides a detailed description of the proposed technique, while Section VII evaluates its performance. Section VIII discusses our findings and the limitations of our work. Section IX concludes the paper.

II. Related Work

Unknown attacks can emerge from newly discovered vulnerabilities [14], referred to as zero-day attacks. They can also manifest through unforeseen vulnerabilities, often called blind spots [4]. Blind spots may be known in the security community and have fixes, but occur due to insufficient security knowledge [20], unforeseen changes in configuration [21], dynamic operating environments [22], delayed updates [23] and end-of-life products [24]. This section discusses approaches for detecting and diagnosing unknown attacks within cyber-physical systems (CPS), encompassing smart homes, IoT security, and computer network security.

Attack Detection: Unknown attack detection techniques can be broadly categorised into reasoning-based and machine learning-based [6] approaches.

iPSTL (inference parametric signal temporal logic) formulae have been used to model a train brake system and detect unknown attacks by automatically identifying the parameters that distinguish an anomaly from a normal value [25]. Other techniques such as zone partitioning have also been used to identify atypical causal relationships [26] in sensor data for industrial control systems. However, such reasoning-based techniques focus on single-variable data, limiting their applicability in contexts with multidimensional data, such as larger CPS like smart homes.

Supervised anomaly detection techniques rely on labelled training data, which limits their ability to detect previously unseen attacks that deviate from the network traffic in the training set. These methods include SARIMA and LSTM models for learning network behaviour bounds [27], LSTM-RNNs for detecting anomalous sensor values [28], and GANs combined with LSTM-RNNs to generate attack detection scores [29]. Other approaches incorporate privacy-preserving techniques by modelling network features as Gaussian Mixture Models and applying Kalman filtering to identify anomalies [30]. Federated deep learning has also been applied to classify attacks in IoT networks, though it does not address the detection of unknown attacks [31]. However, these approaches aim to detect known attacks in the datasets used for training and are not evaluated against previously unseen attacks. To address the limitations of supervised learning, some studies adopt few-shot and transfer learning techniques to detect unknown variants of known attacks. Few-shot learning using Siamese Convolutional Neural Networks has been applied to anomaly detection in cyber-physical systems to mitigate the challenge of limited labelled data [32]. Feature-based transfer learning has also been proposed, where clustering is used to estimate the relevance between source and target domains under the assumption that most network attacks belong to variants of known network attack families and share common traits in network features [33]. However, these methods have not been evaluated against attacks outside the datasets used, and their effectiveness in generalising to novel attacks remains unclear.

Supervised learning techniques have also been applied to malware detection in previously unknown attacks—for example, by identifying malicious command-and-control (C&C) communications [34], using deep learning to analyze Windows API calls for malware classification [35], or employing few-shot learning methods to detect malware based on opcode frequency analysis [36]. Another technique detects zero-day web attacks by training an RNN on benign data admitted by a Web Application Firewall (WAF). It then uses neural machine translation to to differentiate suspicious HTTP traces from benign, but does not diagnose unknown attacks or discuss effectiveness against encrypted traffic [37]. Moreover, these techniques are not directly applicable to CPSs due to varying processor architectures and platform-specific binaries, which may not be readily available with proprietary firmware.

A more effective approach to detect unknown attacks is to use unsupervised learning techniques [7], [38], [39]. Some works use clustering to identify new anomalies in low dimensional spaces with sparse data [7] or large amounts of network data collected from an ISP [39]. Other techniques train deep learning ensembles without labels to simulate unknowns [38]. Although they effectively detect unknown attacks, these techniques do not reason about the anomalies detected. A recent survey on unknown attack detection identified One-Class SVM (One-SVM), Local Outlier Factor, and Isolation Forest as effective machine learning algorithms [14]. In our study, we re-implemented and compared these methods, as the surveyed works did not provide their source code. Deep neural networks have also been used to model benign contextual behaviour of smart home devices and detect unknown contextual attacks by analysing control-plane data from an open-source home automation platform [40]. While these methods are effective in identifying unknown attacks, they do not offer explanations or reason about the detected anomalies.

One work on detecting zero-day attacks in malware binaries defines a zero-day attack as having a CVE but no disclosure by a vendor [41]. This work creates ground truth of virus data, identifies malicious binaries in hosts, and then analyses their presence on the Internet using Symantec antivirus software. The authors acknowledge that the technique is better suited to detecting host-based attacks with observable malware binaries. However, checking for disclosed CVEs can support the diagnosis of zero-day attacks.

Attack Diagnosis: Diagnosis in cybersecurity generally refers to the post-attack analysis of available data to inform and support remediation efforts [42], [43]. However, statistical and machine learning-based techniques often use diagnosis interchangeably with detection and typically refer to attack classification rather than reasoning about them.

A statistical S-estimator-based technique has been proposed to diagnose data injection attacks in phasor measurement unit (PMU) state estimation by enhancing the robustness of extended Kalman filters [44]. An ensemble of deep learning models combining wavelet transforms and singular value decomposition has been used to detect false data injection attacks in DC microgrids [45]. A diagnosis filter based on robust optimisation has also been developed to detect stealthy multivariate data injection attacks that introduce random errors into alternating current state estimation [46]. While these techniques are effective in identifying the presence and type of attack, they do not reason about which security requirements have been violated. Anomaly detection techniques have also been used to learn the normal behaviour of smart home devices using an autoencoder (AE), identify anomalies in run time traffic, and then use k-means clustering to analyse only the infrequent flows - therby classifying rare benign traces from unknown (malicious) ones [47]. Another technique self-adapts by applying reinforcement learning to tune the hyperparameters of an unsupervised classifier, using a reward function derived from clustering its own outputs [48]. Although these papers do not mention the word diagnosis, the second level analysis that they are doing after detecting the unknown attack is consistent with the definition of diagnosis [49].

Explainable AI (XAI) techniques are widely applied in the security literature to explain detection of network intrusions, malware, phishing, and botnet attacks [19]. Among the commonly used XAI methods, Shapley values (SHAP), Local Interpretable Model-agnostic Explanations (LIME), permutation importance, and feature importance are frequently employed to explain machine learning decisions in cybersecurity research. In the intrusion detection literature (IDS), SHAP has been used to provide local and partial explanations for detected anomalies [50], and to support causal reasoning that enhances human interpretability of machine learning outputs [51]. SHAP has also been used to enhance model transparency of botnet detection models [52], and to explain the identification of unknown malware [53]. The LIME technique has also been used to generate local explanations of individual anomalies detected by IDS [54]. It has also been used in conjunction with permutation importance (PI) in botnet detection algorithms treated as black-boxes to identify feature contributions for individual predictions and highlight the most influential features across the dataset, respectively [55]. For more interpretable outputs, LIME has been applied to explain the predictions of a convolutional neural network (CNN) trained on app descriptions from the Google Play Store, identifying specific words that triggered the detection of dangerous permissions[56]. In this paper, we

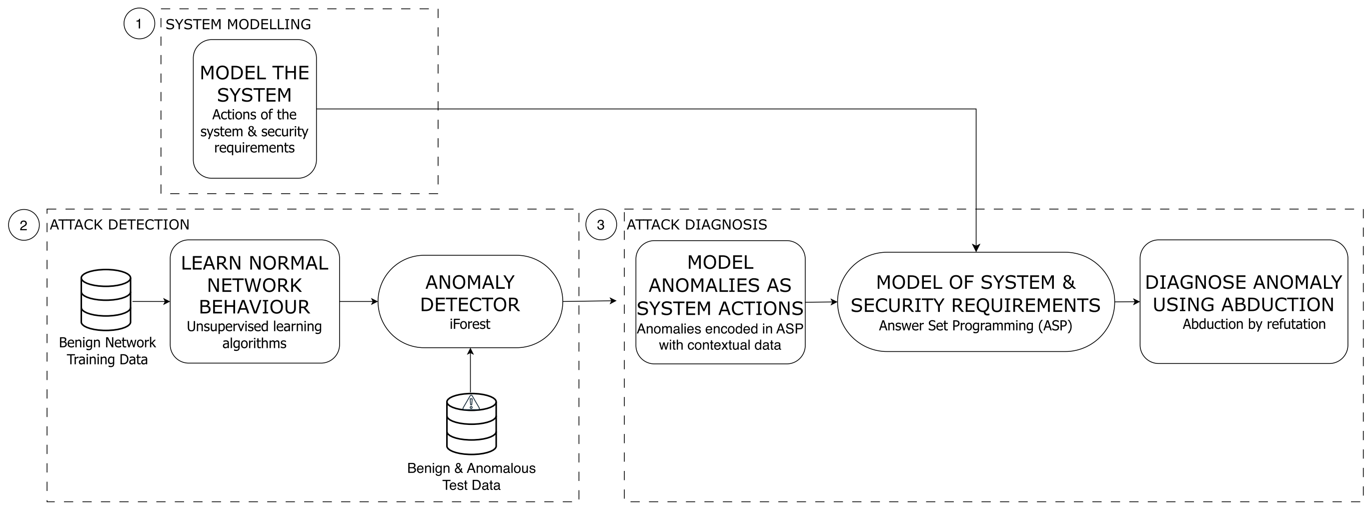
1
SYSTEM MODELLING
MODEL THE SYSTEM
Actions of the system & security requirements
2
ATTACK DETECTION
Benign Network Training Data
LEARN NORMAL NETWORK BEHAVIOUR
Unsupervised learning algorithms
ANOMALY DETECTOR
iForest
Benign & Anomalous Test Data
3
ATTACK DIAGNOSIS
MODEL ANOMALIES AS SYSTEM ACTIONS
Anomalies encoded in ASP with contextual data
MODEL OF SYSTEM & SECURITY REQUIREMENTS
Answer Set Programming (ASP)
DIAGNOSE ANOMALY USING ABDUCTION
Abduction by refutation

Fig. 1. Overview of the unknown attack detection and diagnosis technique.

compare our technique with traditional XAI approaches used for diagnosis to explain the inferences made by the anomaly detector.

To our knowledge, the literature that reasons about unknown attacks is limited, and abductive reasoning techniques, known for their effectiveness in reasoning about behavioural anomalies [10], hold promise. Assumption-Based Truth Maintenance Systems [13] incorporate beliefs that include abduction when propositional clauses with hard true or false classifications are insufficient. HORN clauses [13] have been used to determine the most likely explanation represented in penalty logic. However, both these techniques require modelling the world using probabilistic methods that are not directly suitable for our smart home scenario and can be reserved for future work. ASP-based representations that support partial models and counterexample generation emerge as a promising avenue for diagnosing anomalies in cyber-physical systems [57] due to their tractability and expressive language. While abductive reasoning has different application domains, e.g. generating specifications for forensic-ready systems [58] and identifying violating safety properties [11], there is a research opportunity to use it to diagnose unknown attacks.

Our approach is different from root cause analysis. NIST guidelines on managing information security risk [49] and industry resources on cyber incident response [59] describe root cause analysis as the process of identifying the underlying causes or vulnerabilities responsible for a given attack. In contrast, our approach focuses on bridging the gap between network anomalies detected at run time and the security requirements defined during the system's threat modelling process when there is no access to device status [60]. We aim to reason about the symptoms of an attack, i.e., how they manifest in the home network. We argue that identifying the underlying vulnerabilities for the network anomalies detected requires access to firmware or backend software, which we would like to explore in future work.

III. Overview of Our Technique

The novelty of this work lies in its end-to-end approach that integrates smart home modelling, runtime network anomaly detection, and diagnosis of anomalies[1]. While existing research has addressed system modelling and attack detection, we are not aware of any prior work that diagnoses attacks in terms of violated security requirements. Moreover, to the best of our knowledge, no existing technique sequentially combines all three actions to detect and diagnose unknown attacks. Our technique comprises three key activities, illustrated in Fig. 1 in execution order. (1) *System Modelling*: We first create a smart home model representing possible legitimate actions and its security requirements using Answer Set Programming (ASP) (Section IV). (2) *Attack Detection*: We then explore and implement unsupervised machine learning techniques to model benign device behaviour and detect network anomalies in the smart home (Section V). The significance of the anomaly detection algorithm lies in its critical role in identifying abnormal network behaviour, potentially indicative of unknown attacks. (3) *Attack Diagnosis*: We represent the detected anomalies in ASP and diagnose them using abductive reasoning in refutation mode (Section VI). These three activities allow us to detect and diagnose unknown attacks and select adequate security controls to mitigate them.

We require datasets that represent real devices to design the smart home model and train the anomaly detector on the normal behaviour of the devices in the smart home. However, only a few smart home datasets contain real network data and/or sufficient features that can be used to reason about detected anomalies [14]. Among those available, we have selected the CICIoT2023 [17] and the IoT-23 dataset [18]. CICIoT2023 [17] includes both benign and attack data from real-world devices, providing a realistic representation of the behaviour of the smart home

[1]The source code of the experiments and system models are available at https://github.com/kushalramkumar/more-than-zero.

network. The IoT-23 dataset [18] complements the CICIoT2023 dataset by offering simulated attacks using IoT devices. Both datasets employ a flat network topology in which devices are directly connected to the router or through a network switch.

IV. System Modelling

This section describes how we model the smart home and its security requirements. To model the system's actions and the security requirements that govern them, we require a language and tooling that supports both expressive rule-based specification and automated reasoning over potentially incomplete system models. This includes evaluating which security requirements are violated by any anomalies detected in the network traffic. Answer Set Programming (ASP) is a declarative programming paradigm in which the system is described through logic programs rather than imperative control flow [57]. Compared to propositional logic [61], which models system behavior using logical connectives (e.g., AND, OR) applied to true or false propositions, and its extension, temporal logic [25], which incorporates time-based variations, ASP offers non-monotonic reasoning—making it better suited for dynamic, partially modeled systems such as smart homes.

We have chosen the Clingo ASP tool [12] for our implementation due to its expressivity and tractability [13]. We manually created the model for this work, but in future work, we intend to explore the usage of inductive learning and neuro-symbolic learning to infer the model at run time from the network traces of the system [15].

A. Smart Home

We aim to represent a typical smart home network characterised by a router connected to a number of devices. An example of this topology is provided in Fig. 2 in the paper that describes the CICIoT2023 dataset [17]. The smart home devices are the assets to be protected and are defined as a *type* in Clingo. The type is then used to define the domain of objects, such as the individual devices (e.g., *router, alexaechodot*) and relationships between them, as shown in Listing 1. Note that lines starting with '%' represent comments.

```
% Define Types
type(device).

% Devices within a smart home
device(router; alexaechodot; amazonplug;
    amcrestcamera; dlinkcamera;
    philipshuebridge; techkinlightstrip;
    irobotroomba; rpi).
```

Listing 1. System Assets

We model network interactions using a predicate, as shown in Listing 2. A predicate is a function or relation that takes one or more arguments and evaluates to true or false depending on whether the arguments satisfy certain conditions.

```
{ communicate(S,D,T,P,F) : endpoints(S),
    endpoints(D), protocols(P), S != D,
    packet_rate(F) } = 1 :- T = 0..23.
```

Listing 2. Communicate Predicate

The arguments of the predicate include the source of the network trace (*S*), the destination (*D*), the time at which it was sent (*T*), the protocol used to communicate (*P*) and a descriptor of whether the packet rate was within the learned normal limit for the device (*F*). We can model the communication in the smart home without listing every instantiation of the predicate ourselves using a choice construct ('{}'). This syntactic element enumerates the list of possibilities of a predicate. Clingo expands this construct to all possible combinations of the communicate predicate given the constraints on its variables. Clingo does not natively support temporal logic, so we discretise the time (*T*) between 0 and 23 hours. While a more granular approach can be taken to represent the time of the network flows, this representation was sufficient for our example.

We did not represent the internal behaviour of the smart home devices since we assumed that our security solution could only view the network traffic in the smart home. This is a reasonable assumption considering that it is not easy to monitor the internals of the IoT devices, which are based on proprietary firmware [60].

We model the domain assumptions of the system, e.g., *"the home's router is secure"*, as grounded atoms, which are predicates where the variables are replaced with constants. The assumption that a router is secure if the password is over eight characters long and uses an encrypted protocol (*wpa2*) is formalised, as shown in Listing 3.

```
% Domain assumption
password(router, 8).
encrypted(router, wpa2).
#const l = 8.

protected(router) :- password(router, L),
   L >= l, encrypted(router, wpa2).
```

Listing 3. Domain Assumptions

The model of the system is not meant to exhaustively cover all possible actions within the home that are not essential to demonstrate the effectiveness of our technique. The complete model used in our study is provided in the replication package.

B. Threats and Security Goals

We begin by performing a threat model of the smart home using a combination of the OWASP Threat Modelling Process [62] and an asset-centric approach [63] that assesses a home's assets, their security goals & requirements, and the threats against the system. We first identify the system's assets, which are the home's devices. We began with a simple example of a smart home but expanded it to include the list of devices in the datasets (Table I) to enable validation. We assume that only the network traffic is observable, not the internals of the devices, which may be using proprietary firmware. We also

TABLE I
LIST OF DEVICES AND ATTACKS

Device	Attack
Philips Hue Bridge	DDoS HTTP Flood
iRobotRoomba	DNS Spoofing
AmcrestCamera	DoS HTTP Flood
DlinkCamera	DoS HTTP Flood
AlexaEchoDot	Mirai UDP Plain
AmazonPlug	Recon Port Scan
TechkinLightStrip	Recon Port Scan
Raspberry Pi	Upload Attack
Smart Speaker	Ultrasonic Voice Command Attack
Raspberry Pi	Mirai Botnet
Raspberry Pi	Torii Botnet
Raspberry Pi	Trojan Botnet
Raspberry Pi	Gagfyt Botnet
Raspberry Pi	Kenjiro Botnet
Raspberry Pi	Okiru Botnet
Raspberry Pi	Hakai Botnet
Raspberry Pi	IRCBot Botnet
Raspberry Pi	Muhstik Botnet
Raspberry Pi	Hide&Seek Botnet

assume that the router is secure and that the user does not intentionally communicate with malicious actors. The attack vector considered is the smart home network, and we also include one example to demonstrate physical attacks (ultrasonic voice command attack).

We then defined the system's security goals in terms of confidentiality, integrity, and availability of the identified system assets [64]. To do so, we refer to a recent taxonomy of cyber-physical threats against a smart home [21] that classifies 24 types of attacks based on violated security goals, attack vectors, and impact on the system and domestic life. These goals are listed in Table II. The confidentiality goals of the system aim to safeguarde the secrecy of data transmitted to/from a device, and protect sensitive device data that attackers can probe to identify vulnerabilities (rows *CCOM1* & *CDEV1*). The integrity goals require trustworthiness of communication and commands, authorised inter-device communication, and the integrity of device software and firmware (*ICOM1, ICOM2, & IDEV1*). Lastly, the availability goals ensure that the devices remain accessible and operate as intended (*ADEV1* & *ADEV2*).

C. Security Requirements

Security requirements establish constraints on the system that operationalise one or more security goals [65]. Unlike security goals, requirements are verifiable and outline specific actions to be prevented, constraints on functional requirements, and assumptions regarding the system. The security requirements that operationalise the security goals discussed in Section IV-B are listed in Table II. The list of security requirements is not meant to be exhaustive but is detailed enough to demonstrate the technique's effectiveness.

The *CCOM1* requirement prevents devices from communicating using unencrypted protocols, violation of which could indicate a security misconfiguration or compromised device. *CDEV1* prevents the traffic rate of the same type from exceeding a learned limit. The attacks in the datasets that violate this requirement are Port Scan, Brute Force, and Botnet attacks (which typically begin by probing for device vulnerabilities). Brute force attacks and Port Scan attacks (i.e., reconnaissance) [17] probe a system for weaknesses and are characterised by abnormal traffic volumes to a device, similar to a DoS attack. Brute Force attacks [17] often consist of repeated submissions of requests to gain unauthorised access to confidential data of a system, such as a case with dictionary attacks that attempt to guess a password.

The *ICOM1* requirement prevents devices from communicating with malicious endpoints. The DNS Spoofing, Uploading Attack, and the C&C Communication of Botnet attacks violate this security requirement. An uploading attack [17] is one where an attacker attempts to upload a malicious file onto a device, which may violate multiple security goals and requirements. If the malicious file is a known malware binary, its signature may be available in malware databases. Without observing the malware binaries or changes to the device's internal state, these attacks are difficult to diagnose from network traces alone. Further, deep packet inspection can be difficult if the attacker uses HTTPS or some encrypted communication protocol. Given our limitation of only being able to observe the network traffic, we can still diagnose uploading attacks by their communication with a malicious endpoint. In a DNS Spoofing attack, the attacker corrupts the local DNS cache to redirect requests to malicious sites [17]. However, while it might not be possible to detect the corruption of a DNS cache from the network traffic alone, this man-in-the-middle (MitM) can be diagnosed when the traffic from a device is redirected to a malicious endpoint, similar to uploading attacks.

The *ICOM2* requirement prevents devices from communicating with each other unless authorised to do so. A stealthy Recon attack might violate this requirement. While there are valid situations in which devices may communicate with each other (e.g., a smart speaker controlling devices in a home), such communication may be an indicator of a malicious/compromised device. The datasets that we considered did not include this type of Recon attack.

The *IDEV1* requirement prevents devices from communicating outside the permitted operating hours. It is violated by cyber-physical attacks such as ultrasonic voice command attacks where the attack is carried out through a physical medium and maybe undetectable by a network monitor. The impact of this attack can be observed through unauthorised actuation of another device within the home. This attack is not present in the datasets used for our study, but we have included it to illustrate how a user-specifiable requirement can be used to diagnose an attack performed using a physical medium that is not monitored.

Requirements *ADEV1* and *ADEV2* prevent the rate of traffic to/from a device from exceeding a learned threshold. The former is violated by DoS attack whereas the latter is violated by DDoS/Botnet attacks. A denial-of-service (DoS) attack is a flood of network traffic that overwhelms the target so that it cannot receive or transmit data, i.e., renders it unavailable [21]. A distributed denial-of-service (DDoS) attack is similar to a DoS attack except that the perpetrators are typically more than one [21].

TABLE II
LIST OF SECURITY REQUIREMENTS

Index	Security Goal	Security Requirement	Attack	Diagnosis
CCOM1	Confidentiality - Communication	Devices do not communicate using unencrypted protocols	Security Misconfiguration / Vulnerability	Vulnerability/Malware
CDEV1	Confidentiality - Device Data	Rate of requests of the same type do not exceed learned limit	Port Scan, Brute Force, Botnets (Mirai, Torii, Trojan, Gagfyt, Kenjiro, Okiru, Hakai, IRCBot, Muhstik, Hide&Seek)	Recon/BruteForce
ICOM1	Integrity - Communication	Devices do not communicate with malicious endpoints	DNS Spoofing, Uploading Attack, Botnets (Mirai, Torii, Trojan, Gagfyt, Kenjiro, Okiru, Hakai, IRCBot, Muhstik, Hide&Seek)	MiTM/Malware
ICOM2	Integrity - Communication	Devices do not communicate with each other unless authorised	Port Scan / Reconnaissance Attack	Recon
IDEV1	Integrity - Device Firmware	Devices do not communicate outside permitted hours	Ultrasonic Voice Command Attack	Vulnerability/Malware
ADEV1	Availability - Device	Rate of traffic to/from a single source does not exceed the learned threshold	DoS HTTP Flood	DoS
ADEV2	Availability - Device	Rate of traffic to/from multiple sources does not exceed the learned threshold	DDoS HTTP Flood, Botnets (Mirai, Torii, Trojan, Gagfyt, Kenjiro, Okiru, Hakai, IRCBot, Muhstik, Hide&Seek)	DDoS

Botnet attacks can be executed in multiple stages and may violate different security requirements at each stage. For instance, communication with a C&C server can violate requirement *ICOM1*, the DDoS violates requirement *ADEV2*, and reconnaissance activities can violate requirement *CDEV1*.

The security requirements are encoded using integrity constraints, i.e., rules or conditions that must be satisfied for a model to be considered valid. If an anomaly violates an integrity constraint, it is found to violate that security requirement. For instance, Man-in-the-Middle and some Malware may force a device to communicate with malicious endpoints. A security requirement that prevents such unsafe communication is encoded, as shown below.

```
:- communicate(_,X,_,_,_), malicious_endpoints
    (X).
```

Listing 4. Security Requirement as an Integrity Constraint

The '_' represents variables that are unused or irrelevant to the integrity constraint, and can take any value the predicate permits. In this case, the integrity constraint prevents any communication where the destination is a malicious endpoint.

V. Detecting Anomalies in Smart Homes

To select the anomaly detection algorithm for our approach, we considered three algorithms that have been shown to be effective [14], [15] - One Class SVM (One-SVM), Local Outlier Factor and Isolation Forest. We excluded deep learning algorithms such as autoencoders [47], since they require a large amount of training data, and in our experiments, the Recon attack against the Amazon Plug had as few as 1,000 benign samples. Further, the creators of the CICIoT2023 dataset used in this paper found decision tree-based approaches such as Random Forest to outperform a deep neural network [17]. We used the Scikit-learn library [66] for our implementation due to its ease of use and our familiarity with it. We initialised the One-SVM with an RBF kernel since the network traffic in the dataset is not linearly separable [67]. For the Local Outlier Factor, we maintained the k_neighbours parameter to its default value (20) because any value above 10 could remove statistical interference [68]. We retained the default parameters of the Isolation Forest since, during our experiments, they allowed the algorithm to achieve the highest F1 score overall across all the attacks considered in the study (Tables IV and VI).

The default predict functions in Scikit-learn use static thresholds set in the libraries, which performed poorly at identifying anomalies in our experiments. To mitigate this issue, we adopted a two-stage thresholding approach [69], in which the anomaly scores output by the algorithms are subjected to a univariate analysis technique [70], [71] that identifies thresholds for anomalous packets. We evaluated well-known univariate techniques [72] such as IQR/Tukey Fences, 95th and 99th Percentile, and Z-Score thresholds, as shown in Table V, and selected the Z-score thresholding method since it had the best performance. A detailed discussion of the results is described in Section VII.

We assume that the smart home is secure during the training phase. Thus, we only use benign data to learn the system's normal behaviour, but we use the labelled data during the test phases. This simulates a situation where the attack traces are injected into the system, since they are only seen by the model for the first time during testing. However, we note that since the datasets chosen in our study do not indicate when an attack has failed, every attack trace is assumed to be successful. This can be remedied in future work by using a test bed that includes successful and failed attacks. We created anomaly detection models per device trained on benign data and tested them using anomalous data. This was because previous studies have shown the difficulties in identifying thresholds and creating a single model for a heterogeneous collection of devices [6], [60], which could also be intermittently connected to the network. Our findings also corroborate the need to create anomaly detection models for each device.

Since the abductive reasoning logic uses ASP, the network anomalies reported by the machine learning algorithm must be represented in this language. To do that, we programmatically convert the network trace flagged as anomalous to the communicate predicate (Listing 2). The anomaly detector provides a binary classification output by considering all the features in the dataset as an input. Since the communicate predicate does not require all of these features, we process the network trace in a pandas dataframe to identify the source, destination, and whether the traffic exceeds normal thresholds for this device and represent it in the format shown in Listings 5, 6, 7, and 8. The anomalous network traces are represented as atoms in the model, and we classify them as security anomalies if they are found to violate the model, e.g., a smart bulb that communicates with a Command & Control (C&C) server violates the security requirement *ICOM1*.

```
% Anomaly Trace: Device communicates with
% malware site
communicate(bulb,c2c_server1,_,https,_).
```

Listing 5. Anomaly Trace

We also introduce other contextual data (e.g., the availability of devices and the reputation of the initiator of communication) that may help diagnose the anomaly. We use the impacts on the system (e.g. malicious network traffic) and domestic life (e.g. device availability) [21] to represent contextual factors that a network monitor or smart home user can provide about an attack that can help with diagnosis. We augment the anomalies detected with three types of contextual data: (1) the computed variation from normal network traffic rate, (2) the number and reputation of the source(s) of the traffic, and (3) the availability of the device after the attack. Many types of attacks exhibit anomalous packet rates, and we use Tukey Fences, to identify the bounds of normal traffic and then identify abnormal rates [73]. If an anomalous network trace exhibits an abnormal packet rate, the *packet_rate* field of the communicate predicate is changed to "exceeds_limit". The trace below was generated for a Port Scan attack against the Amazon Plug.

```
available(amazonplug).communicate(rpi,
    amazonplug,10,https,exceeds_limit). %Recon
```

Listing 6. Contextual Factor - Rate Limit

Known malicious endpoints can be identified using IP blacklists and IP reputation checkers [74], which can be useful in identifying botnet attacks and multiple types of malware. While such checkers might miss new malware endpoints or local ones (as with the datasets we used in which the attackers were present in the LAN), a more sensitive diagnosis logic might choose to flag a locally originated packet or one with an unknown reputation as malicious. This could increase the false positive rate but lead to fewer missed anomalies. We change the *source* variable of the communicate predicate to "malicious_endpoint" when communication is done with insecure endpoints. The following trace was generated when the Raspberry Pi communicated with a C&C server during the Kenjiro botnet attack.

```
available(rpi-17-1).communicate(
    malicious_endpoint,rpi-17-1,10,https,
    within_limit). %MitM/Malware
```

Listing 7. Contextual Factor - Endpoint Reputation

To distinguish between DoS and DDoS, we modify the *source* of the communicate predicate to "single_endpoint" or "multiple_endpoints". We verified this by inspecting the network packet captures provided with the CICIoT2023 dataset. Knowing whether a device is offline after detecting an anomaly can also be useful in diagnosis. This is observable through network monitors or user input and may not be available in existing datasets. The following trace was generated after detecting the DoS HTTP Flood attack against the Dlink Camera. Note the addition of the device's availability at the beginning of the trace and the modified source.

```
not available(dlinkcamera).communicate(
    single_endpoint,dlinkcamera,10,https,
    exceeds_limit). %DoS
```

Listing 8. Contextual Factor - Availability & Number of Sources

In practice, contextual facts are relatively easy to monitor at run time since multiple tools and techniques exist to identify them. However, since we are working with a dataset and not a testbed, we have programmatically added the contextual factors ourselves while encoding each anomalous network trace in ASP, with knowledge obtained from inspection of the network traces of each anomaly and the labels of each attack.

VI. Diagnosis Using Abductive Reasoning

In cybersecurity, diagnosis typically involves analysing available data post-attack to support remediation [42], [43]. Since we first identify anomalous network behaviours and then diagnose them, our objective is to determine a plausible explanation (cause) for the observed network anomalies (effect). Specifically, given a partial model of the system and anomalous observation(s) that may signify unknown attacks, we need a technique that can identify which security requirement is violated by the attack.

Abductive reasoning is a technique that identifies a plausible explanation for an event based on a given system description [10], and proves to be an apt choice for this task. We opted for it due to its efficacy in generating diagnoses in the desired format and its demonstrated tractability using Clingo ASP.

Our approach draws inspiration from prior work employing **abduction by refutation** to identify safety property violations in a system with a partial model [11]. The similarities with our use case are that we aim to generate explanations for anomalies (safety violations in their case), we also have a partially modelled system, and we require a solution that always terminates to

Algorithm 1 Abduction by refutation for diagnosing network anomalies

```
Input: Model of a smart home, security requirements and network anomalies encoded in Clingo ASP
Output: Class of an attack and violated security requirement
  for each anomaly do
      check satisfiability of the model
      if not satisfied then
          for requirement in list of security requirements do
              exclude requirement
              check satisfiability of the model
              if satisfied then
                  requirement is the diagnosis
              end if
          end for
      else
          not a security anomaly
      end if
  end for
```

be able to respond to security attacks. Abduction by refutation is a *reasoning method in which a hypothesis is generated by eliminating alternative explanations that contradict the system invariants [11]*.

In our technique, given a network anomaly (counterexample) that violates the system invariants (security requirements), it means identifying which security requirement must be excluded for the anomaly to satisfy the system model (contradiction). By making the anomaly satisfy the model, we recognise the condition in which that contradiction can exit, thus identifying the violated security requirement. Our technique is described in Algorithm 1. Our model defines the security requirements for the different asset types (Listing 1) rather than for each device individually. When provided with a network anomaly, Clingo grounds every predicate for each type of device to identify which security requirement has been violated. For example, the step that identifies the counterexample of the Mirai UDP Plain botnet's DDoS phase evaluates 226,290 rules, 33 choice constructs, 47,234 atoms, and 44,944 bodies. In the absence of ASP, this would be a cumbersome and error-prone process that involves defining the predicates of normal operation and the security requirements for each device.

While the effectiveness of our technique against the attacks in the datasets considered is discussed in Section VII, it is important to note that it applies to various cyber-physical attacks where the attack vector or impact extends beyond the cyber space [21]. Although we do not formally apply frameworks that identify advanced persistent threats such as the Cyber Kill Chain [75], the IoT-23 dataset contains attack traces from the different phases of each botnet attack, from Reconnaissance, to Command&Control, and finally Action on Objectives (i.e., DDoS exploit). We are able to diagnose the different stages of the botnet attacks, but since the dataset discretises each of these phases into different network flows and lacks temporal data for the different phases, we cannot string the diagnoses together. In this section, we demonstrate the diagnosis technique using two examples from the list of attacks evaluated. In Section VII, we discuss the diagnosis of all the attacks chosen in this study, which are among the most common attacks against smart homes [21].

The first example describes the diagnosis of the Ultrasonic Voice Command attack against a smart speaker. Devices such as smart speakers may be controlled by physical media such as voice that introduce new attack vectors. For example, some smart speakers are vulnerable to an ultrasonic voice command attack [76], [77] that allows unauthorised actuation and, in turn, control of other devices within the home through inaudible commands. Traditional security solutions that monitor only the network may miss such attacks but can potentially detect their symptoms, such as the anomalous actions initiated by the attack. Examples of such actions would be commands sent from the smart speaker to other devices outside of regular operating hours or if the user observes anomalous smart speaker activity. Although the latter requires user intervention, the former may be observable if the smart speaker sends actuation commands within the Local Area Network (LAN) [78], [79]. As an example, we included an integrity constraint to the model that identifies any commands sent by the smart speaker outside of the normal operating hours set by the smart homeowner as a violation of the integrity of the smart speaker firmware potentially due to a compromise as detailed in Listing 9.

```
% User Generated Requirement: The Smart
   Speaker must not be operated between
   23:00-04:00 hours
permitted_operating_time(T) :- T > 4, T <= 22,
     T = 0..23.
:- communicate(X,_,T,_,_), X = smart_speaker,
   not permitted_operating_time(T).
```

Listing 9. Security Requirement - Communication Outside Permitted Hours (IDEV1)

Using a smart home outside the selected operating hours violates the above security requirement and results in an unsatisfiable model. This anomalous usage is represented below.

```
communicate(smart_speaker,trusted_app_server
   ,23,https,within_limit).
```

Listing 10. Anomaly Trace - Unauthorised Actuation

The abductive reasoning logic inserts this anomaly as an atom (where values replace the variables of the predicate) into the system, checks model satisfiability, and, if not satisfied, identifies the violated security requirement using Algorithm 1. The diagnosis is then output in the format shown below:

```
Violated Security Requirement:  User Generated
     Requirement: The Smart Speaker must not
   be operated between 23:00-04:00 hours
Diagnosis: Vulnerability/Malware
```

Listing 11. Diagnosis - Ultrasonic Voice Command Attack

This diagnosis can inform the smart home owner that the smart speaker is being operated by an unauthorised user (row

IDEV1 in Table IX). While there are only a few mitigations to such attacks, this diagnosis enables smart home owners and software engineers to implement an adequate security control. One of them would be to enable voice matching on the device [76]. In this case, such mitigation would require adequate explanations to the homeowner and mechanisms for user intervention, both of which we reserve for future work. However, in this example, we wish to highlight the benefit of providing a diagnosis for the anomalies observed.

The second example demonstrates the diagnosis of the DDoS HTTP Flood attack against the Philips Hue Bridge. The security requirement that prevents a DDoS attack is indicated in row *ADEV2* in Table II, and is encoded as the integrity constraint shown below.

```
% Availability Security Requirement : Volume
    of traffic from multiple sources does not
    exceed learned threshold
:- communicate(X,Y,_,P,F), X =
    multiple_endpoints, device(Y), protocols
    (P), F = exceeds_limit, not available(X)
    . % Diagnosis: DDoS/Botnet
```

Listing 12. Security Requirement - Excess Rate of Traffic from Multiple Sources (ADEV2)

The anomaly trace generated by our technique is represented below.

```
not available(philipshuebridge).communicate(
    multiple_endpoints,philipshuebridge,10,
    https,exceeds_limit). %DDoS/Botnet
```

Listing 13. Anomaly Trace - DDoS Attack

As described in the previous example, the abductive reasoning logic inserts this anomaly into the system, and Algorithm 1 generates the following diagnosis.

```
Violated Security Requirement:  Availability
Security Requirement : Volume of traffic
    from multiple sources does not exceed
    learned threshold
Diagnosis: DDoS/Botnet
```

Listing 14. Diagnosis - DDoS

DDoS/Botnet attacks are typically performed in multiple stages and this diagnosis helps us identify that the anomaly is caused by the botnet actuation phase which sends a large volume of traffic targeted at a device. This informs our choice of a suitable security control, which is to rate limit the traffic that is sent to a device (row ADEV2 in Table IX).

A list of the possible security controls for each of the attacks included in our study is provided in Table IX. These were identified from the works previously cited defining each attack, and from a review of the grey literature.

VII. Evaluation

Our evaluation aims to assess the detection and diagnosis techniques independently, as these are distinct actions that must be performed when an attack occurs. For attack detection, we evaluate the anomaly detector's ability to identify network attacks while minimising false negatives, with Recall serving as the primary metric. For attack diagnosis, we assess the correctness of the abductive reasoning logic in identifying both the class of attack and the violated security requirement, using Precision as the evaluation metric. The rationale is that the anomaly detector may tolerate a higher number of false positives in order to maximise attack coverage, since the diagnosis step subsequently attempts to reason about the anomalies and filter them. Our evaluation aims to answer the following research questions:

- **RQ1:** Which attack detection algorithm performs best across the selected datasets?
- **RQ2:** How correct are the violated security requirements identified by the diagnosis technique for each detected anomaly?
- **RQ3:** How does our end-to-end technique for detecting and diagnosing unknown attacks compare with a baseline approach that combines machine learning and explainable AI?

TABLE III
Comparison of IoT and Smart Home Datasets

Dataset	Smart Home	Attack Coverage	Real Devices	Labelled
UNSW-NB15	✗	✓	✓	✓
Edge-IIoT	✗	✓	✓	✓
Ghost-IoT	✓	✗	✓	✗
IoT-23	✓	✓	✓	✓
CICIoT2023	✓	✓	✓	✓

A. Dataset Selection & Preparation

In a recent survey of 44 IoT and IIoT security datasets [80], the researchers found that the most common attack types are reconnaissance (e.g., port scans), (D)DoS, MiTM (e.g., brute force), and botnet attacks. They also highlighted that most datasets feature TCP- or UDP-based traffic and noted a lack of datasets containing the ZigBee, CoAP, and other IoT/IIoT-specific protocols. For our study, we considered several options, including UNSW-NB15 [81], Edge-IIoTset Cyber Security Dataset of IoT & IIoT [82], IoT-23 [18], CICIoT2023 [17], and Ghost IoT [83], to identify those that comprehensively represent the common attack types against IoT/IIoT networks [80], and smart homes [21]. We compared the datasets based on the following criteria: whether they were for smart home environments, covered representative attack classes from smart home attack topologies, included data collected from real devices, and contained labelled instances to enable evaluation. This comparison can be found in Table III.

We chose the CICIoT2023 and IoT-23 datasets for our validation, as they met all of our criteria, providing realistic and varied smart home data. The CICIoT2023 dataset includes traffic from 105 different IoT devices, simulating 33 types of attacks, and is well suited for investigating the various types of devices found in smart homes. The IoT-23 dataset captures network traffic from different phases of 14 botnet attacks on smart home IoT devices, making it one of the most comprehensive datasets for studying botnet behaviour.

To simulate unknown attacks, the attack labels in both datasets were excluded while training the anomaly detection model and implementing the abductive reasoning logic but were included during the evaluation. We chose one device from each category in the CICIoT2023 dataset (e.g., Lighting, Home Automation) since not all attacks were performed against all devices. From the IoT-23 dataset we excluded the Linux Mirai and Hajime botnets because they are platform-specific, whereas evaluating other attacks gives better cross-platform validation. Further, the attacks are performed by Raspberry Pi (RPi) devices within the network, and in contrast to the CICIoT2023 dataset, the devices are the perpetrators of the attacks rather than the victims.

The CICIoT2023 dataset is highly imbalanced, containing very little benign traffic relative to attack traffic in both subsets. Our initial experiments with a single anomaly detection model across all devices in Table I proved ineffective due to the wide variance in network features and the dataset's skewed distribution. Random Forest-based feature importance revealed no distinguishing features between benign and malicious traffic, indicating a near-random distribution. Negative bounds in Inter-Quartile Rate (IQR) lower bounds confirmed a strong right skew (larger volumes of traffic with higher packet rate or inter-arrival time (IAT)) and the presence of extreme outliers, likely exacerbated by small sample sizes for some devices. The dataset represents aggregated network flows, but the number of packets per flow was inconsistent, making flow metrics incomparable. To address this issue, we introduced a normalised packet rate using flow duration and packet count. While the CICIoT2023 paper describes all attacks being performed against all devices, we found that to be incorrect after multiple experiments which yielded no inferrable statistical correlations between benign and anomalous traffic, and upon inspection of the network packet captures (PCAP). We contacted the dataset authors to find the actual list of attacks against devices and created Table I based on it. Consequently, this led to us regenerating the datasets from the original PCAP files to isolate device-specific attack traffic.

To regenerate the PCAP files, we first identified the MAC addresses of the devices in Table I from the CICIoT2023 paper and, based on our correspondence with the authors, identified the attacks performed against those devices. We downloaded all the PCAP files provided and filtered them by device and traffic type (benign or attack) using tcpdump. We then modified the Generating_dataset.py file provided in the supplementary material of the dataset to process the new files and produce the CSVs used for model training and evaluation. As tcpdump outputs in PCAP-NG format, we converted the files to standard PCAP to ensure compatibility with the script. The final dataset that we prepared contains over 60 features including statistical features (e.g., flow duration, rate, average and standard deviation), protocol-specific flags (e.g., TCP, UDP, HTTP, DNS), and temporal characteristics (e.g., inter-arrival time, flow idle time).

The IoT-23 dataset consists of multiple files in the Zeek log format, which needed to be converted into CSV format for processing by the iForest algorithm. We found that the labels provided for the network traffic were inconsistent, and needed to be cleaned. To maintain consistency with the CICIoT2023 dataset, we removed fields such as IP source address and headers and computed packet rate from available features. Given the large size of the dataset, we chose to process it in chunks. As iForest requires numerical input, non-numeric fields were either converted or categorically encoded (e.g., protocol, service type, and connection state). The cleaned dataset includes 22 features that capture traffic volume statistics (e.g., duration, orig_bytes, resp_bytes, orig_pkts, resp_pkts), protocol and service indicators (e.g., proto_tcp, proto_udp, service_dns, service_http), and connection states (e.g., conn_state_SF, conn_state_S0). We have made both the cleaned and prepared datasets publicly available for future research.[2]

B. Evaluation Metrics

Given the unbalanced nature of the data in an anomaly detection problem, we apply the three-sigma rule, which states that for a normal distribution of data, about 99.7% of the data falls within two standard deviations (3σ) of the mean. Consequently, we chose a conservative split of 95% benign to 5% anomalous traffic (2σ), assuming that most of the traffic in a smart home is normal, with sporadic malicious activity. In this case, the accuracy metric would not be suitable for evaluation since mispredicting every anomaly would still result in a score of 95%. Instead, we have chosen the Area Under the Curve - Precision Recall (AUC-PR), Precision, Recall and F1 score metrics to evaluate our model [84].

The AUC-PR metric measures reparability or how well the model can distinguish between benign and anomalous traffic. Precision is defined as the proportion of true positives to the sum of true positives and false positives and evaluates the proportion of anomalies reported as true anomalies, i.e., a measure of the reported false positives. Recall is the ratio of true positive predictions to the actual number of positive instances in the dataset and evaluates the fraction of true anomalies identified, i.e., a measure of the reported false negatives. The F1 score is the harmonic mean of precision and recall and evaluates the balance of precision and recall in predicting anomalies.

Since diagnosis can only occur once the attack has been detected, the two steps are sequential and the diagnosis logic will not process any false negatives from the anomaly detector. Consequently, from a security context, the false negatives of the attack detection logic must be minimised to ensure that attacks are not missed. Although we assess the AUC-PR and F1 score for the attack diagnosis, the F1 score is more relevant because it indicates a balance between missed anomalies (false negatives) and incorrect diagnoses (false positives). Although our technique effectively diagnosed attacks in both datasets, each dataset presented unique challenges, especially in understanding and cleaning the data, as discussed later in this section.

C. Results

1) Attack Detection:

RQ1: *Which attack detection algorithm performs best across the selected datasets?*

[2]The cleaned and prepared datasets are available at https://zenodo.org/records/14380178

TABLE IV
COMPARISON OF IFOREST, ONE-SVM, AND LOF RESULTS WITH THE CICIOT2023 DATASET

Attack	Device	iForest				One-SVM				LOF			
		Precision	Recall	AUC PR	F1 Score	Precision	Recall	AUC PR	F1 Score	Precision	Recall	AUC PR	F1 Score
DDoS HTTP Flood	Philips Hue Bridge	0.9536	0.9774	0.9764	0.9653	0	0	0.6365	0	0.8857	0.1575	0.7680	0.2663
DNS Spoofing	iRobot Roomba	0.7188	0.9978	0.9991	0.8345	0.9935	0.9978	0.9980	0.9956	1.0000	0.6483	1.0000	0.7856
DoS HTTP Flood	Amcrest Camera	0.9997	0.6946	0.9954	0.8197	0.8621	0.2099	0.4764	0.3346	0.9097	0.0123	0.7086	0.0243
DoS HTTP Flood	Dlink Camera	0.9998	0.5012	0.9740	0.6666	0.9994	0.6640	0.9033	0.7979	0.9998	0.3588	0.9757	0.5281
Mirai UDP Plain	Alexa Echo Dot	0.9994	0.8092	0.9935	0.8902	0.9423	0.2488	0.4218	0.3936	0.7930	0.0087	0.4731	0.0172
Recon Port Scan	Amazon Plug	0.8414	0.9068	0.9193	0.8726	0.7554	0.9375	0.8607	0.8364	0.9849	0.9276	0.9449	0.9553
Recon Port Scan	Techkin Light Strip	0.5002	0.8072	0.9124	0.6174	0.4498	1.0000	0.9118	0.6202	0.9691	0.6788	0.8512	0.7949
Upload Attack	RPi	0.5320	1.0000	0.9697	0.6920	0.3558	1.0000	0.8906	0.5240	0.0571	0.0364	0.2151	0.0433

TABLE V
COMPARISON OF ANOMALY THRESHOLD METHODS USING ISOLATION FOREST WITH THE CICIOT2023 DATASETS

Attack	Device	Z-Score				IQR				95th Percentile			
		Precision	Recall	AUC PR	F1 Score	Precision	Recall	AUC PR	F1 Score	Precision	Recall	AUC PR	F1 Score
DDoS HTTP Flood	Philips Hue Bridge	0.9536	0.9774	0.9764	0.9653	0	0	0.9764	0	0	0	0.9764	0
DNS Spoofing	iRobot Roomba	0.7188	0.9978	0.9991	0.8345	0	0	0.9991	0	0	0	0.9991	0
DoS HTTP Flood	Amcrest Camera	0.9997	0.6946	0.9954	0.8197	0.0612	0.0033	0.9954	0.0062	0.0662	0.0035	0.9954	0.0067
DoS HTTP Flood	Dlink Camera	0.9998	0.5012	0.9740	0.6666	0.3017	0.0020	0.9740	0.0040	0.7776	0.0175	0.9740	0.0342
Mirai UDP Plain	Alexa Echo Dot	0.9994	0.8092	0.9935	0.8902	0.0794	0.0056	0.9935	0.0105	0.0041	0.0002	0.9935	0.0004
Recon Port Scan	Amazon Plug	0.8414	0.9068	0.9193	0.8726	0.6234	0.8000	0.9697	0.6991	0	0	0.9193	0
Recon Port Scan	Techkin Light Strip	0.5002	0.8072	0.9124	0.6174	0.8444	0.4001	0.9124	0.5084	0	0	0.9124	0
Upload Attack	RPi	0.5320	1.0000	0.9697	0.6920	0.6234	0.8000	0.9697	0.6991	0	0	0.9697	0

We compared the performance of the three machine learning algorithms at anomaly detection tasks using the CICIoT2023 dataset to identify the most suitable one for our use case, i.e., one with a consistently high F1 score demonstrating a good balance between precision and recall. We used k-fold cross-validation with a parameter of 10, which has generally been effective in practice [85] and present the results for One-SVM, LOF, and iForest in Table IV. This also allowed us to perform a more robust evaluation that provides an opportunity for different samples in the dataset to appear in both training and test datasets while also averaging the results of each iteration. While the One-SVM was the most effective at identifying DNS Spoofing attacks, and the LOF was better at identifying Port Scan attacks, the iForest showed better results overall, leading us to select it for the rest of our experiments.

To select the threshold for the anomaly scores reported by the iForest, we evaluated the IQR/Tukey fences, 95th and 99th percentile, and Z-Score threshold techniques (results shown in Table V. The IQR/Tukey Fences thresholding method showed varying results, often resulting in low precision and recall scores. Although this method was moderately effective for specific cases (e.g., Recon Port Scan on Techkin Light Strip with an F1 score of 0.508), it could not achieve consistently good performance across the whole set of attacks targeting different smart home devices.

Percentile-based thresholding techniques (e.g., 90th, 95th, and 99th) exhibited poor performance in capturing anomalous behaviour across the dataset. The 99th percentile was a very high threshold and excluded most anomalies. Consequently, its results were excluded from our study. The 90th and 95th percentile performed better and were very similar to each other, but they could not achieve meaningful precision or recall for most attack types. This was likely due to their inability to capture subtler deviations in anomaly scores indicative of anomalous behaviour. Consequently, the F1 scores were consistently low or zero across most cases, indicating the limitations of percentile-based approaches for our use case. Since the results were very similar, for brevity, we provide the results of the 95th percentile in Table V.

The Z-Score thresholding method demonstrated strong performance across all attack types, achieving high precision, recall, and F1 scores for most attacks. The flexibility of this method in detecting both lower- and upper-bounded anomalies makes it effective in discriminating benign behaviour from attacks. To select the Z-score thresholds, we inspected the histograms of the anomaly scores of each model. For the Isolation Forest, we found that a threshold $<$ -0.4 is slightly less than the mean of anomaly scores, and a threshold of 0 best suits the benign data. Similarly, we chose a threshold of 0 for the One-SVM model since the histograms showed positive values as inliers. With the LOF, thresholds were more widely spread depending on the attack, but we obtained very similar results with thresholds of 0, -0.4, -0.5, and -1.0.

The iForest was consistently able to correctly classify DDoS HTTP Flood, DNS Spoofing Mirai UDP Plain and Port Scan (against the Amazon Plug) as anomalies. However, we observed lower recall, i.e., a higher number of false negatives, in the case of DoS HTTP Flood attacks against Amcrest and Dlink cameras. This could be because the attack data traffic pattern appears similar to the benign ones for both cameras since they generate a continuous stream of network traffic. Similarly, the Port Scan attack traffic appears to be very similar to the benign traffic, explaining the poor precision, i.e., high false positives, for the TechkinLightStrip device. The results with the Amazon Plug were satisfactory despite having less data available for training.

The results of the anomaly detection experiments with the IoT-23 dataset are shown in Table VI. The iForest was very effective in identifying anomalies in most cases, but some discussion on exceptions is needed. The Torii and Trojan attacks did not have adequate data to train the iForest model as indicated by the poor performance. However, the abductive reasoning logic

TABLE VI
ISOLATION FOREST RESULTS USING Z-SCORE THRESHOLD WITH THE IOT-23 DATASET

Attack	Device	Precision	Recall	AUC PR	F1 score
Mirai	RPi	0.9704	0.6202	0.9476	0.7567
Torii	RPi	0.0103	1.0000	0.9237	0.0204
Trojan	RPi	0.0029	0.6000	nan	0.0058
Gagfyt	RPi	0.6946	1.0000	0.9842	0.8179
Kenjiro	RPi	0.9999	0.7274	0.8289	0.8421
Okiru	RPi	0.9220	0.0006	0.7305	0.0012
Hakai	RPi	0.9975	1.0000	0.9971	0.9987
IRCBot	RPi	0.9999	0.7500	0.7960	0.8571
Muhstik	RPi	0.9932	0.5178	0.8730	0.6807
Hide & Seek	RPi	0.8856	0.9999	0.9093	0.9393

was still effective at identifying false positives. The Okiru attack data had very few benign samples to train a model. The Linux Mirai and Hajime attacks were ignored from our study because they are platform-specific.

2) Attack Diagnosis:

RQ2: *How correct are the violated security requirements identified by the diagnosis technique for each detected anomaly?*

To evaluate the abductive reasoning technique, we modified the attack labels provided in the dataset to annotate the violated security requirements and the class of attack the anomaly could belong to. When the anomalies themselves are included in the ASP model of the smart home, we convert the anomalous network traces into ASP notation and augment it with contextual facts similar to how they would be obtained from monitoring the system or from a user of the system. This is done to simulate a situation where the attack is unknown, but we know the security requirements we would like to ensure. While augmenting anomalies with contextual factors can be viewed as similar to training with labeled data that may artificially improve the performance of the algorithm, we would like to point out that we only added contextual factors that are readily observable or verifiable, that have been studied before in academic & grey literature.

In all cases, the range of benign network packet rates is computed using IQR ranges, and if any anomaly trace has an abnormal rate, the flow is marked as one that exceeds the permissible limit. DDoS HTTP Flood and botnet attacks are typically performed by multiple devices as observed in the PCAP files of the CICIoT2023 dataset, although it is not included as a feature. The contextual factor added to aid in the diagnosis is that the source of the observed anomaly contains multiple endpoints. The malware Uploading Attack, communication with C&C servers in botnet attacks, and DNS Spoofing attacks are marked as occurring with malicious endpoints. This can be verified using IP reputation checkers and checking the DNS resolution for requests. In our experiments, for some cases of the DNS spoofing attack, we identified destinations with lower reputation scores, however, the same could not be done for packets from within the LAN. In the case of the various DoS, DDoS and botnet attacks, the two datasets do not indicate if the victims of the attack went offline. We have assumed that the attacks are successful and that the devices have gone offline to

TABLE VII
ABDUCTIVE REASONING RESULTS WITH THE CICIOT2023 DATASET

Attack	Device	Precision	Recall	F1 score
DDoS HTTP Flood	Philips Hue Bridge	0.8348	0.8348	0.8348
DNS Spoofing	iRobot Roomba	0.8710	0.8710	0.8710
DoS HTTP Flood	Amcrest Camera	0.1205	0.1205	0.1205
DoS HTTP Flood	Dlink Camera	0.9995	0.9995	0.9995
Mirai UDP Plain	Alexa Echo Dot	0.9995	0.9995	0.9995
Recon Port Scan	Amazon Plug	0.8000	0.8000	0.8000
Recon Port Scan	Techkin Light Strip	0.8148	0.8148	0.8148
Upload Attack	RPi	0.2105	0.2105	0.2105

TABLE VIII
ABDUCTIVE REASONING RESULTS WITH THE IOT-23 DATASET

Attack	Device	Precision	Recall	F1 score
Mirai	RPi	0.9933	0.9933	0.9933
Torii	RPi	0.8667	0.8667	0.8667
Trojan	RPi	1.0000	1.0000	1.0000
Gagfyt	RPi	0.8571	0.8571	0.8571
Kenjiro	RPi	0.5065	0.5065	0.5065
Okiru	RPi	0.9319	0.9319	0.9319
Hakai	RPi	1.0000	1.0000	1.0000
IRCBot	RPi	1.0000	1.0000	1.0000
Muhstik	RPi	0.9218	0.9218	0.9218
Hide&Seek	RPi	0.9625	0.9625	0.9625

aid in the diagnosis, which can easily be achieved by pinging the devices. If these attacks are performed against the devices but do not go offline (i.e., facts are wrong), we still diagnose them as Recon/Brute Force attacks since it is difficult to distinguish between them.

Since we simulate scenarios where the attacks are unknown, the diagnoses we generate are on the violated security requirement and the class of the attack that those violations indicate. In this context, we needed to relabel the attacks in the dataset to provide valuable results. We combined DDoS and Botnet attacks as a potential diagnosis since botnet attacks are often a type of DDoS attack. Due to the similarities in the manifestation of the attacks, Recon and Brute Force attacks are combined. Any communication with a malicious endpoint could indicate many different types of attacks, such as with C&C flows of botnet attacks and DNS Spoofing, and they were grouped as Man-in-the-Middle (MitM) or Malware. We used the precision, recall and F1 score metrics to evaluate the abductive reasoning technique as well and for the same reasons. The results of the abductive reasoning experiments for the CICIoT2023 are shown in Table VII, and those for the IoT-23 dataset are in Table VIII.

The diagnosis technique works well against most attacks, except the DoS HTTP Flood attack against the Amcrest camera and the malware upload attack against the Raspberry Pi. Upon inspecting the ASP representation of the anomalies, we found that the IQR technique was ineffective at distinguishing normal from attack packet rates which are an important criterion for diagnosing DoS attacks. We believe this is due to the similarity between the normal data streamed from the device and the DoS attack patterns in the Amcrest camera. We do not observe the same phenomenon in the Dlink camera against which the same attack was performed. In the future, we would like to explore other temporal analysis techniques that better model the flow of network packets to determine anomalous traffic rates [27]. In the case of the uploading attack, we could not identify any malicious endpoint using the previously described techniques, so we did not augment the anomalies with the data required to perform a diagnosis. To improve the malicious endpoint detection, we could perform a stricter filtering of the results of an IP reputation checker wherein all inconclusive results are considered potentially unsafe.

In the case of the IoT-23 dataset, each attack consisted of multiple phases, each considered as an individual attack. For example, the Botnet attack includes a Recon stage for probing the devices, communication with the C&C server, and finally a DDoS attack. While the results in Table VIII are labelled by the class of the attack, the diagnosis metrics were computed by aggregating the individual values obtained for each stage of the attacks. Our technique is effective at diagnosing all attacks except the Kenjiro attack. Since each anomaly originated from a single source, they were falsely diagnosed as DoS, not DDoS/Botnet. This underscores the need for accurate facts for diagnosis.

The diagnosis of anomalies took between 0.33s per anomaly in the best case to 0.8s in the worst case in our experiments. While this diagnosis time is sufficient to react to an attack, there is scope for improvement.

D. Comparison With Benchmarks

RQ3: *How does our end-to-end technique for detecting and diagnosing unknown attacks compare with a baseline approach that combines machine learning and explainable AI?*

Our approach to sequentially perform attack detection and diagnosis is novel, and we did not find any technique in the literature to directly benchmark our work against. To construct a comparable end-to-end benchmark, we selected an attack detection algorithm widely used for identifying unknown and zero-day attacks. We combined it with feature relevance and explainable AI (XAI) techniques to serve as the diagnosis component. We applied these techniques against the CICIoT2023 dataset, which we have cleaned and prepared, and compared the results of each with those of our approach.

For attack detection, we selected the Random Forest (RF) since (1) it was recommended by the CICIoT2023 dataset creators, who observed strong performance in their experiments [17], and (2) among non-deep learning (DL) techniques, RF is widely used for detecting zero-day attacks, as highlighted in recent surveys [14]. We excluded deep learning techniques because our dataset does not contain sufficient samples for all the attacks. We also compared One-SVM, LOF and iForest as described in Section VII-C1.

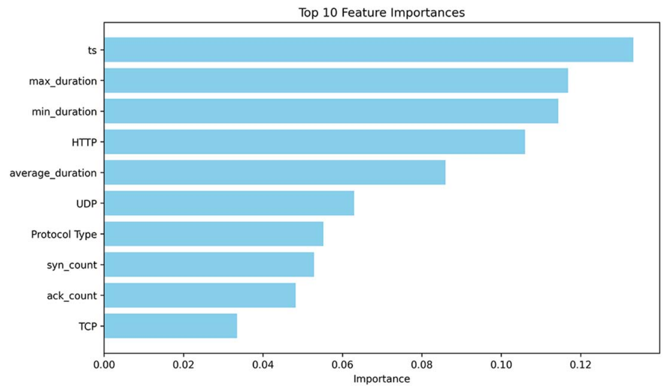

Fig. 2. Feature importance results of DDoS HTTP Flood attack against Philips Hue Bridge.

Our initial experiments with RF resulted in overfitting (precision, recall, and F1 score = 1), as we had hugely imbalanced data. To address this issue, we first ensured there was no data leakage (between training and testing) and then tried stratified sampling to ensure that the same class balance was maintained in both training and validation. That still resulted in an overfitted model. We next applied SMOTE to rebalance the minority class and a standard scaler to reduce the impact of large outliers. Then, we trained the RF classifier, which still resulted in overfitted classes for most attacks. While these results are better than our approach, an overfitted model is not generalisable to attack traces not present in the dataset.

Supervised learning techniques, such as Random Forest (RF), can accurately identify the class of attack since they typically excel at classification tasks when provided with sufficient labelled data. However, they cannot be applied when the attack traces broadly differ from the training data. Thus, we compared our attack diagnosis technique with feature relevance techniques, including feature and permutation importance, which explain which features contributed the most to the classification. We repeated the experiment for each device-attack pair in the dataset and included all the results in the replication package, but we discussed a few examples here.

We then applied two feature relevance techniques (Feature Importance and Permutation Importance), and two explainable AI techniques (SHAP and LIME) on the outputs of the attack detection phase, and compared the results with our attack diagnosis approach. For the feature importance experiments, we identified the top 10 important features of 62 available in the dataset for each attack-device pair. Looking at the Feature Importance (FI) results of the DDoS HTTP Flood attack against a Philips Hue Bridge (Fig. 2), it is unclear as to (1) what is abnormal about the timestamp (ts), (2) what the min/max/average durations of the flows should be (whether they have been exceeded or throttled), (3) why UDP is more important than TCP despite it being an HTTP based attack. Only if the type of attack is known in advance is it possible to surmise that the SYN/ACK counts could be due to multiple attempted connections, indicative of a DoS/Recon/Brute Force attack.

In addition to this, we also applied the Permutation Importance (PI) technique that attempts to identify the predictive features post-training, by assessing the performance of the model in the absence of selected features. PI was unable to identify any explanations, consistently returning a score of 0 for most features across the different devices and attacks. Since we had already attempted data rebalancing and cross-validation techniques, a search of the grey literature indicates that this is likely due to (1) small and insufficiently complex data patterns (which is the nature of these attacks), or (2) an overfitted model resulting from that dataset.

Despite having an overfitted model that should perfectly capture the dataset, neither approach provides enough information to diagnose the violated security requirement and the class of attack or allow us to select an appropriate mitigation strategy. In comparison, our diagnosis technique identifies the violated security requirements, as shown in Listing 12, and prints the diagnosis, as shown in Listing 14. This enables the selection of a mitigation strategy that prevents high traffic rates to a device and also inspects the reputation of the multiple sources sending nearly identical traffic to the device.

We then compared our attack diagnosis technique with posthoc-explainability XAI techniques. Compared with transparent-model approaches (by design), posthoc-explainability techniques (after prediction) are better for decision tree-based methods (such as RF and iForest) [86]. Of the explainable AI techniques commonly used in IDS, malware, phishing and botnet detection, SHAP and LIME are among the most commonly used for posthoc-explanations for machine learning-based approaches [19]. These two reasons prompted the selection of the SHAP and LIME techniques for our experiments. SHAP is better suited for global explanations (i.e., generalised explanations of the model prediction), and LIME is better suited for local explanations (i.e., why the model identified a particular instance as anomalous).

Compared to feature relevance techniques, which also give global explanations, SHAP values provide more comprehensive insights into how individual features contribute to the model's predictions and are better when there are complex interactions between features (as is the case with RF). The results obtained by applying SHAP for the DDoS HTTP Flood attack against a Philips Hue Bridge are shown in Fig. 3, and the results of applying the technique to all the attacks in our study are included in the replication package.

Using SHAP, we obtained similar results regarding feature and permutation importance in that we could not identify (1) why features were deemed important and (2) what the normal range of values for those features looked like. Consequently, they have similar limitations to feature relevance techniques (as discussed previously) regarding their diagnosis capabilities.

In comparison to the feature relevance-based approaches, LIME is better at reasoning about individual instances while providing the range of values that impacted the prediction and which class the features contributed to. Since it provides explanations for each anomaly, we randomly sampled 10 predictions from the RF model for each attack in the CICIoT2023 dataset to study the explanations provided. The explanations for the HTTP

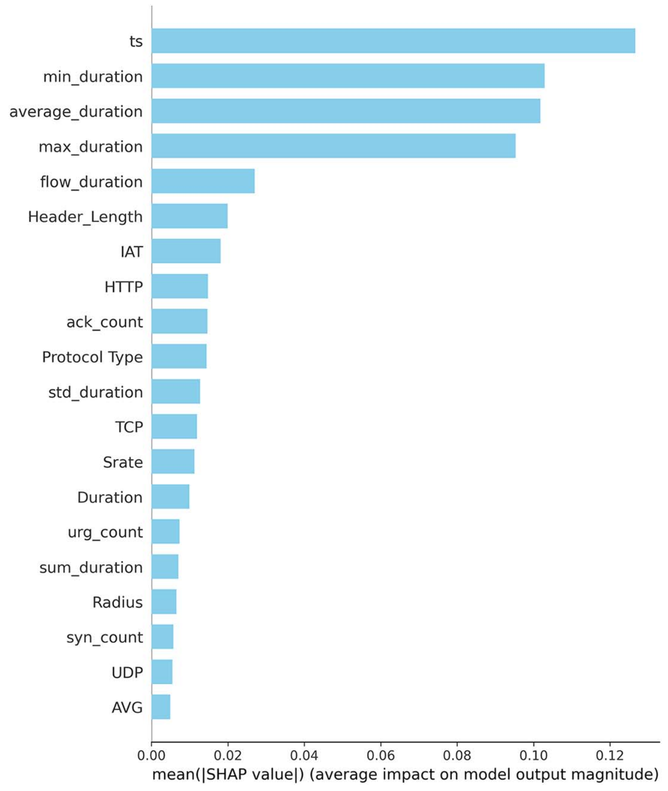

Fig. 3. SHAP results of DDoS HTTP Flood attack against Philips Hue Bridge.

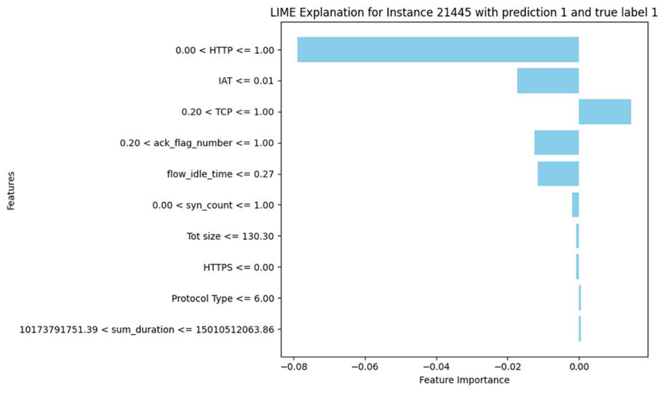

Fig. 4. LIME results of DDoS HTTP Flood attack against Philips Hue Bridge where IAT incorrectly indicates anomalous data to be benign.

DoS Flood attack against the Philips Hue Bridge indicate that flows containing HTTP are benign rather than attack samples despite having stratified sampling. In some cases (Fig. 4), inter-arrival time (IAT) was used to predict benign flows as well when it should have contributed to the positive class (attack). In another case (Fig. 5), IAT was used to correctly explain an HTTP flood and the range of values for the flow to be an attack was also provided. For mispredicted outcomes (Fig. 6), the model identified HTTP traffic with higher ACK flags and IAT as attack flows, when in fact, they were benign flows similar to an HTTP DoS attack.

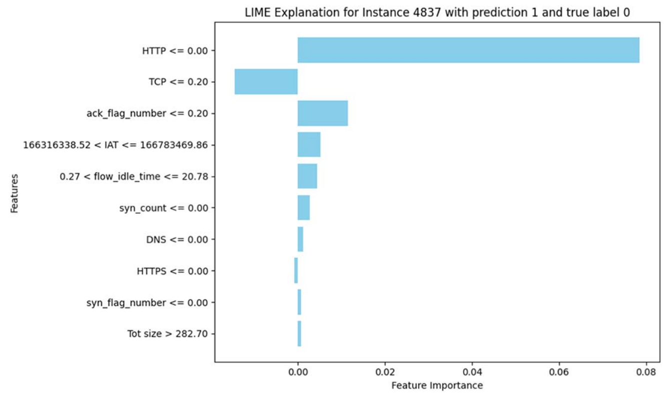

Fig. 5. LIME results of DDoS HTTP Flood attack against Philips Hue Bridge using IAT correctly to classify attack data.

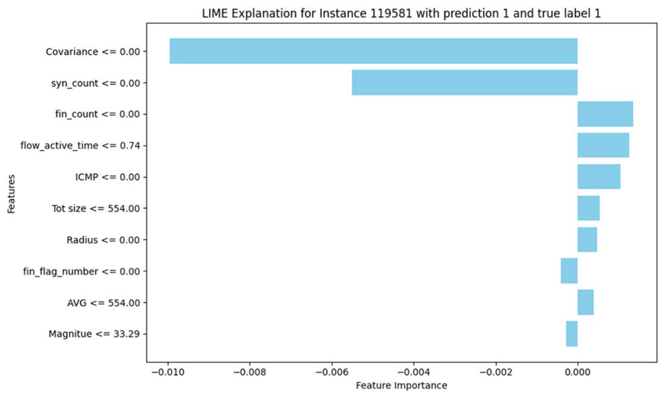

Fig. 7. LIME results of Mirai UDP Plain attack against Alexa Echo Dot.

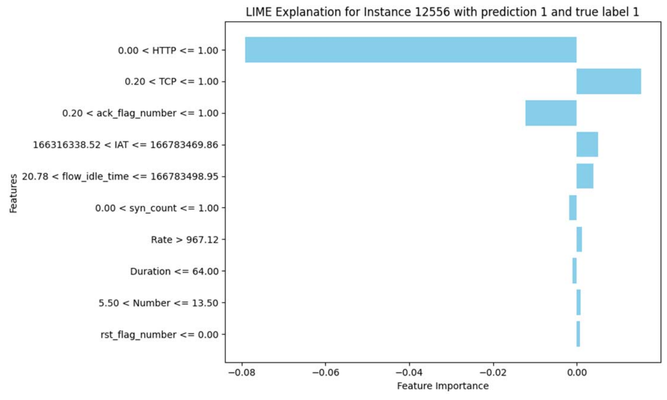

Fig. 6. LIME results of DDoS HTTP Flood attack against Philips Hue Bridge for mispredicted outcomes.

In the case of the Alexa Echo Dot against which the Mirai UDP Plain attacks were performed (Fig. 7), the top 10 features ranked by importance mostly indicate an inclination to the negative class (benign), even though the prediction was correctly made for the positive class (attack). This could be due to the complex relationships between the features in the Random Forest or the aggregate of the positive features outweighing the negative ones despite individual values of the positive features being low. In this case, despite correctly detecting the anomalies, the features with the highest importance do not give useful information to diagnose the attacks. Further, since LIME is better suited to local explanations (for that particular anomalous instance), its explanations could contain features that do not contribute to the prediction.

None of these explainability techniques make use of contextual data such as rate limits, endpoint reputation, availability of devices, and number of sources (Listing 6, 7, 8) from the system in order to reason about the anomalies detected. While it might appear that contextual data could be used to train a supervised learning model, in the case of unknown attacks, which data to use might not always be apparent and would require re-training the model if discovered after deployment. For example, a flood of packets may occur due to Recon attacks or DoS/DDoS attacks. A supervised learning algorithm would not use contextual data about the availability of the targeted device. Thus, it can misclassify a DoS attack as a Recon attack even if the attack makes the targeted device unavailable.

VIII. Discussion

Our approach combines behaviour- and behaviour-specification-based anomaly detection [6]. By detecting network anomalies (behaviour-based) and reasoning about them using the formalism of ASP (behaviour-specification-based), we can mitigate false positives of the anomaly detector while overcoming the need for complete system models. The expressivity of the modelling language allows easy extension of security requirements to include other user preferences or exceptions such as If-This-Then-That (IFTTT) [87] rules that the user might have configured, which are not modelled in our current work. The actual mechanisms in which user input may be sought are left for future work. We demonstrated the generalisability of our technique within the smart home security domain by evaluating it against 18 distinct attacks across 7 representative devices chosen to reflect a well-established taxonomy of threats in such environments [21]. Therefore, we suggest that the attack classes and device types included in our study provides sufficient coverage to support our claims about applicability across common smart home scenarios. In the rest of this section, we discuss how our approach can support the selection of security controls and the implications for human intervention.

A. *Selecting Security Controls*

Although we know and explicitly represent the security requirements that can be violated, we may not know how the attacks that violate them can materialise in specific smart home deployments. We contend that understanding the attacks in terms of the security goals and requirements they violate enables the selection of suitable security controls and eliminates the need for precise attack identification. While the automated selection of security controls after diagnosing an attack is left for future work, in this section, we provide examples of how the identification of the violated security requirements can inform

TABLE IX
LIST OF SECURITY CONTROLS

Index	Security Control
ICOM1	Filter traffic to/from malicious endpoints
ICOM2	Block network traffic between devices initiated by a device that is not whitelisted
ICOM2	Network sandbox the source of suspicious traffic
IDEV1	Enable voice matching features
IDEV1	Block all actuation outside permitted hours
ADEV1	Rate limit single-source traffic
ADEV2	Rate limit multi-source traffic
CDEV1	Filter traffic to/from malicious endpoints
CCOM1	Configure the device to use secure communication protocols
ALL	Update device firmware for patches found in vulnerability databases

the choice of appropriate mitigation strategies. In the case of a Port Scan attack, identifying the abnormal traffic enables us to better filter the network traffic sent to a device. If the perpetrator of the attack is within the home network, a network sandbox may be employed to isolate all traffic from that device. Knowing exactly the malicious behaviour exhibited by the device could enable a search within vulnerability databases to identify if the attack was known and/or mitigated. Based on this analysis, (1) the home owner could be requested to remove the device from the network, or (2) apply security updates if the vulnerability has been fixed by the vendor, or (3) even identify the trade-offs that allow continued operation of the device if the user chooses not to remove it. Some potential mitigations for the attacks included in our study are described in Table IX.

B. Adapting to Changes at Run Time

In a flat network configuration, modifications are limited to the addition and removal of devices, or the modification of device behaviour. In the case of adding a new device, it would be necessary to create a dedicated anomaly detection model for that device. However, provided the device operates over TCP/IP, the ASP model would remain unchanged. If a device's behavior is modified—due, for example, to an update or a shift in user interaction patterns—this can be addressed by (1) the user notifying the system of the update, or (2) the anomaly detector identifying a series of anomalies that do not violate the security requirements defined. This scenario necessitates retraining the anomaly detection model and may also introduce new security requirements depending on the extent of the changes.

We assume that changes in network topology are intentionally made by the smart home user and using secure devices (i.e., router, WiFi extender). When a conventional WiFi extender is used, all inbound and outbound traffic remains monitorable at the main router, allowing the approach to function without modification. In cases where a separate subnet is created within the home—either through an additional router or another WiFi extender—the approach would need to be applied individually within each subnet.

C. Implications for Human Intervention

Although our work focuses on automating the detection and diagnosis of unknown attacks using anomaly detection and abductive reasoning, it also sheds light on potential avenues for human intervention. We envision the involvement of different stakeholders: users (the homeowners or house tenants), security/software engineers responsible for securing the smart home devices or the home network, and pen testers tasked to discover new vulnerabilities by performing offensive testing. These stakeholders can support the execution of specific activities that, in our approach, cannot be automated and should be delegated to humans. They can also improve unknown attack detection and diagnosis and ultimately identify and even execute more robust security controls.

Our approach performs unknown attack detection based on the smart home network behaviour and on contextual information that can be monitored automatically: variation from normal network traffic rate, the number and reputation of the source(s) of the traffic, and the availability of the device affected by an anomaly. The presence of a user in the house opens up the possibility of monitoring some of the abovementioned contextual information (e.g., device availability) and information about the smart home device behaviour (e.g., failed Transport Layer Security (TLS) verification, unexpected pop-ups), which cannot be done automatically. This additional information has the potential to rule out false positives flagged by an anomaly detector (e.g., a DoS attack is flagged, but the targeted device is up and running). We also envision the possibility for the user to pro-actively monitor specific information that could help discover anomalies that would not be noticed from the network behaviour analysis, for example, situations when devices exhibit behaviour not instructed by the user (e.g., a smart speaker unexpectedly reproducing audible sounds, a smart light unexpectedly switching on and off) [88]. This monitored information could support the identification of anomalies overlooked by the anomaly detector.

There could be a situation when the abductive reasoning logic cannot identify a specific security requirement responsible for the anomaly. This may be because the anomaly requires a more complex diagnosis to be explained (more than one security requirement is violated simultaneously) or some domain assumptions present in the model are no longer valid. In such cases, a security/software engineer could be involved in supporting the diagnosis and identifying the classes of attacks that could have caused the anomaly and violated security requirements and domain assumptions. If a diagnosis is identified, an attacker may have exploited an unpatched vulnerability in the targeted device to cause the anomaly. A security/software engineer could help identify important security updates that should be performed on a smart home device. However, a zero-day vulnerability may be present if the device is up to date. Information about the anomaly, contextual information, and the target device could be shared with pen testers at the vendor company to focus their testing activities on identifying the vulnerability causing the anomaly.

The execution of some of the security controls shown in Table IX cannot be automated (e.g., applying security updates and secure device configuration). For example, users can update specific vulnerable devices, or security engineers can modify the network configuration to prevent attacks in specific households. Our approach is the first step towards enduring security by supporting activities that allow us to detect and reason about unknown attacks.

A human-machine collaborative approach would instil a culture of continuous improvement and resilience against evolving cyber threats. In the context of homes and groups of homes, the engagement of stakeholders becomes imperative, as this can support enduring security. To support involvement of users and security/software engineers, there is a need to increase their situational awareness and design human-machine interactions supporting different levels of agency [89] depending on the stakeholder's role and expertise. This is a promising research direction that we aim to explore in future work requiring efforts from multi-discipline collaboration, such as cybersecurity, HCI, psychology, and AI [90].

D. Threats to Validity

Internal Validity: All the attacks considered in our study are network attacks. Consequently, the technique hasn't been evaluated against web and physical attacks. The effectiveness of the diagnosis technique is dependent on the contextual factors that can be identified from the system.

External Validity: While the iForest is a capable technique to model normal device behaviour and identify anomalies, the available datasets did not allow us to evaluate its effectiveness against attacks subtly different from benign network traffic. Our model of the smart home is general enough that it can be used with minor modifications to specific smart home contexts. However, similar partial models must be created for different use cases (e.g., public infrastructures). Further, we created our model manually, but we are currently exploring using inductive and neuro-symbolic learning to automate the creation of such models from network traces at runtime [15].

Although the individual components of our approach are generalisable to other application domains, extending the evaluation to other domains (e.g., industrial IoT, automotive) would require creating the threat models and security requirements from scratch, identifying prevalent attack classes, and implementing domain-appropriate anomaly detection methods before applying our abductive reasoning component. Given that cybersecurity is inherently context-dependent, such cross-domain adaptation exceeds the scope of work for our paper but is something we look forward to exploring in future work.

The performance of the abductive reasoning logic could be improved using parallel processing since the Clingo utility is run as a separate process with inputs from the anomaly detector. In future work, we also plan to implement a hierarchical search of the security requirements that should yield performance gains in the case of related requirements. The approach is resource-intensive since an iForest model is needed per device. While model inference is not as computationally demanding as training, this technique is better suited to running on a dedicated device in the network rather than a resource-constrained device such as a router.

IX. Conclusion

Of the many challenges to providing sustainable security to a long-lived system such as a smart home, detecting unknown attacks as the system is used and diagnosing them in a manner that enables the enactment of suitable security controls is imperative. Unlike existing techniques focused on identifying new variants of known attacks or relying solely on anomaly detection, our approach prioritizes the system's security requirements as the foundation. By bridging the gap between anomalous behaviors indicating potential attacks and the desired security requirements, our technique shows promise in dealing with the vastly complicated category of unknown attacks.

In this paper, we proposed a three-step approach to address these challenges. Firstly, we modeled the smart home and formalised its security requirements using Answer Set Programming (ASP). Next, we developed an iForest-based anomaly detection mechanism to identify abnormal behaviors in the home network that could signify attacks. Finally, we implemented a diagnosis technique using abduction by refutation to diagnose the security requirements violated by network anomalies in the home. Evaluation of our technique using real-world datasets, including CICIoT2023 and IoT23, demonstrated its effectiveness in detecting & diagnosing threats. Our findings highlight the complex nature of security monitoring and decision-making tasks, emphasizing the importance of human intervention, particularly in attack detection and diagnosis.

References

[1] Checkpoint, "Check Point 2022 Security Report," 2022. Accessed: Sep. 20, 2022. [Online]. Available: https://go.checkpoint.com/security-report/page-introduction.php

[2] M. Stone, "Google Project Zero, "The More You Know, The More You Know You Don't Know" 2022. Accessed: Sep. 20, 2022. [Online]. Available: https://googleprojectzero.blogspot.com/2022/04/the-more-you-know-more-you-know-you.html

[3] J. Sadowski, "Zero Tolerance: More Zero-Days Exploited in 2021 Than Ever Before" 2022. Accessed: Sep. 20, 2022. [Online]. Available: https://www.mandiant.com/resources/zero-days-exploited-2021

[4] G. Sharkov, "From cybersecurity to collaborative resiliency," in *Proc. ACM Workshop Automated Decision Making Active Cyber Defense*, 2016, pp. 3–9.

[5] L. Pasquale, K. Ramkumar, W. Cai, J. McCarthy, G. Doherty, and B. Nuseibeh, "Sustainable adaptive security," 2023, *arXiv:2306.04481*.

[6] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–29, 2014, doi: 10.1145/2542049.

[7] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised network intrusion detection systems: detecting the unknown without knowledge," *Comput. Commun.*, vol. 35, no. 7, pp. 772–783, 2012.

[8] C. M. Patterson, J. R. Nurse, and V. N. Franqueira, "Learning from cyber security incidents: A systematic review and future research agenda," *Comput. & Secur.*, vol. 132, 2023, Art. no. 103309.

[9] VectraAI, "A new threat detection model that closes the cybersecurity gap," 2025. Accessed: May 20, 2025. [Online]. Available: https://www.vectra.ai/resources/ebook-a-new-threat-detection-model-that-closes-the-cybersecurity-gap

[10] G. Paul, "Approaches to abductive reasoning: An overview," *Artif. Intell. Rev.*, vol. 7, no. 2, pp. 109–152, 1993.

[11] A. Russo, R. Miller, B. Nuseibeh, and J. Kramer, "An abductive approach for analysing event-based requirements specifications," in *Proc. ICLP*, 2002, pp. 22–37.

[12] R. Kaminski, T. Schaub, and P. Wanko, "A tutorial on hybrid answer set solving with Clingo," in *Proc. Reasoning Web. Semantic Interoperability Web*, 2017, pp. 167–203.

[13] R. Koitz-Hristov and F. Wotawa, "Applying algorithm selection to abductive diagnostic reasoning," *Appl. Intell.*, vol. 48, no. 11, pp. 3976–3994, 2018.

[14] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "Zero-day attack detection: A systematic literature review," *Artif. Intell. Rev.*, vol. 54, no. 10, pp. 1–79, 2023.

[15] A. Drozdov, M. Law, J. Lobo, A. Russo, and M. W. Don, "Online symbolic learning of policies for explainable security," in *Proc. 3rd IEEE Int. Conf. Trust, Privacy Secur. Intell. Syst. Appl. (TPS-ISA)*, 2021, pp. 269–278.

[16] M. Ye, N. Jiang, H. Yang, and Q. Yan, "Security analysis of internet-of-things: A case study of august smart lock," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2017, pp. 499–504.

[17] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 13, 2023, Art. no. 5941.

[18] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic (Version 1.0.0)," 2020. Accessed: Jun. 12, 2023. [Online]. Available: https://www.stratosphereips.org/datasets-iot23

[19] N. Capuano, G. Fenza, V. Loia, and C. Stanzione, "Explainable artificial intelligence in cybersecurity: A survey," *IEEE Access*, vol. 10, pp. 93575–93600, 2022.

[20] E. Zeng, S. Mare, and F. Roesner, "End user security and privacy concerns with smart homes," in *Proc. 13th Symp. Usable Privacy Secur. (SOUPS)*, 2017, pp. 65–80.

[21] R. Heartfield et al., "A taxonomy of cyber-physical threats and impact in the smart home," *Comput. & Secur.*, vol. 78, pp. 398–428, Sep. 2018.

[22] C. Lévy-Bencheton, E. Darra, G. Tétu, G. Dufay, and M. Alattar, "Security and resilience of smart home environments: Good practices and recommendations," *Publications Office Eur. Union*, 2015.

[23] F. Li and V. Paxson, "A large-scale empirical study of security patches," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 2201–2215.

[24] Cisco, "Cisco Security Advisory," 2021. Accessed: Mar. 28, 2024. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/csa/cisco-sa-sb-rv-rce-overflow-ygHByAK.html

[25] A. Jones, Z. Kong, and C. Belta, "Anomaly detection in cyber-physical systems: A formal methods approach," in *Proc. 53rd IEEE Conf. Decis. Control*, 2014, pp. 848–853.

[26] J. Yang, C. Zhou, S. Yang, H. Xu, and B. Hu, "Anomaly detection based on zone partition for security protection of industrial cyber-physical systems," *IEEE Trans. Ind. Electron.*, vol. 65, pp. 4257–4267, 2018.

[27] W. Hao, T. Yang, and Q. Yang, "Hybrid statistical-machine learning for real-time anomaly detection in industrial cyber-physical systems," *IEEE Trans. Automat. Sci. Eng.*, vol. 20, no. 1, pp. 32–46, Jan. 2023.

[28] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, "Anomaly detection in cyber physical systems using recurrent neural networks," in *Proc. IEEE 18th Int. Symp. High Assurance Syst. Eng.*, 2017, pp. 140–145.

[29] P. Freitas de Araujo-Filho, G. Kaddoum, D. R. Campelo, A. Gondim Santos, D. Macêdo, and C. Zanchettin, "Intrusion detection for cyber–physical systems using generative adversarial networks in fog environment," *IEEE Internet Things J.*, vol. 8, pp. 6247–6256, 2021.

[30] M. Keshk, E. Sitnikova, N. Moustafa, J. Hu, and I. Khalil, "An integrated framework for privacy-preserving based anomaly detection for cyber-physical systems," *IEEE Trans. Sustain. Comput.*, vol. 6, pp. 66–79, 2021.

[31] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT-edge devices," *IEEE Internet Things J.*, vol. 9, pp. 3930–3944, 2021.

[32] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Trans. Ind. Inform.*, vol. 17, pp. 5790–5798, 2020.

[33] J. Zhao, S. Shetty, J. W. Pan, C. Kamhoua, and K. Kwiat, "Transfer learning for detecting unknown network attacks," *EURASIP J. Inf. Secur.*, vol. 2019, no. 1, pp. 1–13, 2019.

[34] D. Bekerman, B. Shapira, L. Rokach, and A. Bar, "Unknown malware detection using network traffic classification," in *Proc. IEEE Conf. Comm. Netw. Secur. (CNS)*, 2015, pp. 134–142.

[35] Y. Ye, L. Chen, S. Hou, W. Hardy, and X. Li, "DeepAM: A heterogeneous deep learning framework for intelligent malware detection," *Knowl. Inf. Syst.*, vol. 54, no. 2, pp. 265–285, 2018.

[36] M. Sewak, S. K. Sahay, and H. Rathore, "An investigation of a deep learning based malware detection system," in *Proc. 13th Int. Conf. Availability, Rel. Secur.*, 2018, pp. 1–5.

[37] R. Tang et al., "Zerowall: Detecting zero-day web attacks through encoder-decoder recurrent neural networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2020, pp. 2479–2488.

[38] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "A deep learning ensemble approach to detecting unknown network attacks," *J. Inf. Secur. Appl.*, vol. 67, 2022, Art. no. 103196.

[39] L. Sacramento, I. Medeiros, J. Bota, and M. Correia, "FlowHacker: Detecting unknown network attacks in big traffic data using network flows," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, 2018, pp. 567–572.

[40] P. Rieger, M. Chilese, R. Mohamed, M. Miettinen, H. Fereidooni, and A.-R. Sadeghi, "Argus:context-based detection of stealthy IoT infiltration attacks," in *Proc. 32nd USENIX Secur. Symp. (USENIX Secur.),* 2023, pp. 4301–4318.

[41] L. Bilge and T. Dumitraş, "Before we knew it: An empirical study of zero-day attacks in the real world," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2012, pp. 833–844.

[42] N. I. of Standards and Technology, "Guide to Operational Technology (OT) Security," 2023. Accessed: May 20, 2025. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r3.pdf

[43] I. Institute, "Diving deep into data analytics and its importance in cybersecurity," 2024. Accessed: May 20, 2025. [Online]. Available: https://www.infosecinstitute.com/resources/general-security/data-analytics-in-cybersecurity/

[44] Y. Chakhchoukh, H. Lei, and B. K. Johnson, "Diagnosis of outliers and cyber attacks in dynamic PMU-based power state estimation," *IEEE Trans. Power Syst.*, vol. 35, pp. 1188–1197, 2019.

[45] M. Dehghani, T. Niknam, M. Ghiasi, N. Bayati, and M. Savaghebi, "Cyber-attack detection in dc microgrids based on deep machine learning and wavelet singular values approach," *Electronics*, vol. 10, 2021, Art. no. 1914.

[46] K. Pan, P. Palensky, and P. M. Esfahani, "From static to dynamic anomaly detection with application to power system cyber security," *IEEE Trans. Power Syst.*, vol. 35, pp. 1584–1596, 2019.

[47] Y. Meidan, D. Avraham, H. Libhaber, and A. Shabtai, "CADeSH: Collaborative anomaly detection for smart homes," *IEEE Internet Things J.*, vol. 10, pp. 8514–8532, 2022.

[48] R. Heartfield, G. Loukas, A. Bezemskij, and E. Panaousis, "Self-configurable cyber-physical intrusion detection for smart homes using reinforcement learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 1720–1735, 2021.

[49] N. I. of Standards and Technology, "Guide for conducting risk assessments, special publication 800-30," 2012. Accessed: Oct. 24, 2024. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

[50] Y. Wang, P. Wang, Z. Wang, and M. Cao, "An explainable intrusion detection system," in *Proc. IEEE 23rd Int Conf High Perform. Comput. & Commun.; 7th Int Conf Data Sci. & Syst.; 19th Int Conf Smart City; 7th Int Conf Dependability Sensor, Cloud & Big Data Syst. & Appl. (HPCC/DSS/SmartCity/DependSys)*, 2021, pp. 1657–1662.

[51] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection system based on ensemble trees and Shap method," *Sensors*, vol. 22, 2022, Art. no. 1154.

[52] M. M. Alani, "BotStop: Packet-based efficient and explainable IoT botnet detection using machine learning," *Comput. Commun.*, vol. 193, pp. 53–62, Sep. 2022.

[53] R. Kumar and G. Subbiah, "Zero-day malware detection and effective malware analysis using Shapley ensemble boosting and bagging approach," *Sensors*, vol. 22, 2022, Art. no. 2798.

[54] E. Tcydenova, T. W. Kim, C. Lee, and J. H. Park, "Detection of adversarial attacks in AI-based intrusion detection systems using explainable AI," *Human-Centric Comput Inform Sci.*, vol. 11, p. 35, 2021.

[55] N. B. Rabah, B. L. Grand, and M. K. Pinheiro, "IoT botnet detection using black-box machine learning models: The trade-off between performance and interpretability," in *Proc. IEEE 30th Int. Conf. Enabling Technol.: Infrastruct. Collaborative Enterprises (WETICE),* 2021, pp. 101–106.

[56] J. Feichtner and S. Gruber, “Understanding privacy awareness in android app descriptions using deep learning,” in *Proc. 10th ACM Conf. Data Appl. Secur. Privacy*, 2020, pp. 203–214.

[57] G. Brewka, T. Eiter, and M. Truszczyński, “Answer set programming at a glance,” *Commun. ACM*, vol. 54, no. 12, pp. 92–103, 2011.

[58] D. Alrajeh, L. Pasquale, and B. Nuseibeh, “On evidence preservation requirements for forensic-ready systems,” in *Proc. 11th Joint Meeting Foundations Softw. Eng.*, 2017, pp. 559–569.

[59] Checkpoint, “What is Root Cause Analysis (RCA)?” 2023. Accessed: Oct. 24, 2024. [Online]. Available: https://www.checkpoint.com/cyber-hub/cyber-security/what-is-incident-response/what-is-root-cause-analysis-rca/

[60] T. OConnor, R. Mohamed, M. Miettinen, W. Enck, B. Reaves, and A.-R. Sadeghi, “HomeSnitch: Behavior transparency and control for smart home IoT devices,” in *Proc. 12th Conf. Secur. Privacy Wireless Mobile Netw.*, 2019, pp. 128–138.

[61] L. Pasquale, P. Spoletini, M. Salehie, L. Cavallaro, and B. Nuseibeh, “Automating trade-off analysis of security requirements,” *Requirements Eng.*, vol. 21, no. 4, pp. 481–504, 2016.

[62] L. Conklin, V. Drake, S. Strittmatter, and Z. Braiterman, “Threat modeling process,” Accessed: Mar. 29, 2024. [Online]. Available: https://owasp.org/www-community/Threat_Modeling_Process

[63] M. Salehie, L. Pasquale, I. Omoronyia, R. Ali, and B. Nuseibeh, “Requirements-driven adaptive security: Protecting variable assets at runtime,” in *Proc. 20th IEEE Int. Requirements Eng. Conf. (RE)*, 2012, pp. 111–120.

[64] J. Margulies, C. P. Pfleeger, and S. L. Pfleeger, *Security in Computing.* Englewood Cliffs, NJ, USA: Prentice-Hall, 2015.

[65] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, “Security Requirements Engineering: A Framework for Representation and Analysis,” *IEEE Trans. Softw. Eng.*, vol. 34, pp. 133–153, 2008.

[66] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[67] S. Raschka, “How to Select Support Vector Machine Kernels,” 2016. Accessed: Jan. 30, 2024. [Online]. Available: https://www.kdnuggets.com/2016/06/select-support-vector-machine-kernels.html

[68] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LoF: Identifying density-based local outliers,” in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 93–104.

[69] J. Yang, S. Rahardja, and P. Fränti, “Outlier detection: How to threshold outlier scores?” in *Proc. Int. Conf. Artif. Intell., Information Process. Cloud Comput.*, 2019, pp. 1–6.

[70] I. Ben-Gal, “Outlier detection,” *Data Mining and Knowledge Discovery Handbook*, Taylor & Francis, 2005, pp. 131–146.

[71] L. Davies and U. Gather, “The identification of multiple outliers,” *J. Am. Statist. Assoc.*, vol. 88, no. 423, pp. 782–792, 1993.

[72] I. Spanache, “Measures of variability and z-scores. Why, when and how to use them?” 2020. Accessed: Jan. 25, 2024. [Online]. Available: https://towardsdatascience.com/measures-of-variability-and-z-scores-why-when-and-how-to-use-them-a552005cc1a1

[73] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, “Optimal load distribution for the detection of VM-based DDOs attacks in the cloud,” *IEEE Trans. Services Comput.*, vol. 13, pp. 114–129, 2017.

[74] EasyDmarc, “IP/domain reputation check,” 2024. Accessed: Jan. 18, 2024. [Online]. Available: https://easydmarc.com/tools/ip-domain-reputation-check

[75] L. Martin, “The Cyber Kill Chain,” 2024. Accessed: Oct. 31, 2024. [Online]. Available: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[76] J. Mao, S. Zhu, X. Dai, Q. Lin, and J. Liu, “Watchdog: Detecting ultrasonic-based inaudible voice attacks to smart home systems,” *IEEE Internet Things J*, vol. 7, pp. 8025–8035, 2020.

[77] L. Zhang, Y. Meng, J. Yu, C. Xiang, B. Falk, and H. Zhu, “Voiceprint mimicry attack towards speaker verification system in smart home,” in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2020, pp. 377–386.

[78] Google, “Ways to Build Local Home SDK,” 2024. Accessed: Jan. 22, 2024. [Online]. Available: https://developers.home.google.com/local-home

[79] Amazon, “Alexa Commissionable Interface,” 2024. Accessed: Jan. 22, 2024. [Online]. Available: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-commissionable.html

[80] C. Alex, G. Creado, W. Almobaideen, O. A. Alghanam, and M. Saadeh, “A comprehensive survey for IoT security datasets taxonomy, classification and machine learning mechanisms,” *Comput. Secur.*, vol. 132, 2023, Art. no. 103283.

[81] N. Moustafa and J. Slay, “Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set),” in *Proc. Military Commun. Inf. Syst. Conf. (MilCIS)*, 2015, pp. 1–6.

[82] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, “Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning,” *IEEE Access*, vol. 10, pp. 40281–40306, 2022.

[83] M. Anagnostopoulos, G. Spathoulas, B. Viaño, and J. Augusto-Gonzalez, “Tracing your smart-home devices conversations: A real world IoT traffic data-set,” *Sensors*, vol. 20, 2020, Art. no. 6600.

[84] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf. Process. & Manage.*, vol. 45, no. 4, pp. 427–437, 2009.

[85] J. Brownlee, “A Gentle Introduction to k-fold Cross-Validation,” 2023. https://machinelearningmastery.com/k-fold-cross-validation/ [Accessed: 13.02.2024].

[86] A. Rawal, J. McCoy, D. B. Rawat, B. M. Sadler, and R. S. Amant, “Recent advances in trustworthy explainable artificial intelligence: Status, challenges, and perspectives,” *IEEE Trans. Artif. Intell.*, vol. 3, pp. 852–866, 2021.

[87] C. Cobb et al., “How risky are real users’ {IFTTT} applets?” in *Proc. 16th Symp. Usable Privacy Secur. (SOUPS)*, 2020, pp. 505–529.

[88] N. Huijts et al., “User experiences with simulated cyber-physical attacks on smart home IoT,” *Pers. Ubiquitous Comput.*, vol. 27, no. 6, pp. 2243–2266, 2023.

[89] J. McCarthy and P. Wright, “Putting ‘felt-life’at the centre of human–computer interaction (HCI),” *Cognit., Technol. & Work*, vol. 7, pp. 262–271, 2005.

[90] L. Pasquale, K. Ramkumar, W. Cai, J. McCarthy, G. Doherty, and B. Nuseibeh, “The rocky road to sustainable security,” *IEEE Secur. & Privacy*, vol. 22, pp. 82–86, 2024.

Kushal Ramkumar received the M.Sc. degree in computer engineering from Illinois Institute of Technology. He is currently working toward the Ph.D. degree with the University College Dublin and Lero—The Irish Research Centre for Software, Ireland. His research interests include sustainable security for cyber-physical systems, focusing on engineering adaptive security solutions, and logic-based learning for diagnosing and mitigating unknown attacks. He was inducted into the IEEE Eta Kappa Nu Honors Society for exemplary academic performance during his master’s.

Wanling Cai received the Ph.D. degree in computer science from Hong Kong Baptist University. She is a Postdoctoral Researcher with Trinity College Dublin and a Researcher with Lero—the SFI Research Centre for Software, Dublin, Ireland. Her research interests include human–computer interaction and human-centered AI, focusing on health technologies, recommender systems, and human-centered security and privacy. She has served on the Program and Organization Committees of ACM conferences, such as IUI, RecSys, UMAP, and MUM.

John McCarthy received the Ph.D. degree in applied psychology from the University College Cork. He is a Professor of applied psychology with the University College Cork and a Principal Investigator with Lero—the SFI Research Centre for Software, Cork, Ireland. His research interests include understanding the influence of emerging social, personal, and work technologies on people’s lived experiences and using that understanding to inform design of usable and enriching technologies. His recent work includes a project on responsible software engineering and others at the core of which is design for care in online platforms. He has served on Program Committees for a number of conferences including CHI, DIS, ECCE, and COOP.

Gavin Doherty received the Ph.D. degree in computer science from the University of York. He is a Professor and the Leader of the Health Technology Design Group, School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland. His research interests include supporting and extending the reach of mental health professionals, designing innovative and engaging systems that can be implemented in real-world clinical environments. He is a Distinguished Member of the ACM and the Chair of the ACM Distinguished Speaker Committee.

Bashar Nuseibeh (Member, IEEE) received the Ph.D. degree in software engineering from Imperial College London. He is a Professor of computing and the Head of the Software Engineering and Design Research Group, The Open University, U.K. His research interests include software requirements and design, engineering adaptive systems, and security and privacy. He is a former Editor-in-Chief of IEEE Transactions on Software Engineering, and recipient of the 2025 IEEE Computer Society Harlan D Mills Award. He is a fellow of The Royal Academy of Engineering and the ACM, and a member of the Royal Irish Academy and Academia Europaea.

Liliana Pasquale received the Ph.D. degree in information and software technologies from the Politecnico di Milano. She is an Associate Professor with the University College Dublin and a Funded Investigator with Lero—the SFI Research Centre for Software, Dublin, Ireland. Her research interests include requirements engineering and adaptive systems, focusing on security, privacy, and digital forensics. She is an Associate Editor of IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, a Department Editor of *IEEE Security & Privacy Magazine*, and a member of the review board of ACM *Transactions on Software Engineering and Methodology*.