Self-configurable cyber-physical intrusion detection for smart homes using reinforcement learning

Ryan Heartfield, *Member, IEEE*, George Loukas, Anatolij Bezemskij, Emmanouil Panaousis, *Member, IEEE*

***Abstract*—The modern Internet of Things (IoT)-based smart home is a challenging environment to secure: devices change, new vulnerabilities are discovered and often remain unpatched, and different users interact with their devices differently and have different cyber risk attitudes. A security breach's impact is not limited to cyberspace, as it can also affect or be facilitated in physical space, for example, via voice. In this environment, intrusion detection cannot rely solely on static models that remain the same over time and are the same for all users. We present MAGPIE, the first smart home intrusion detection system that is able to autonomously adjust the decision function of its underlying anomaly classification models to a smart home's changing conditions (e.g., new devices, new automation rules and user interaction with them). The method achieves this goal by applying a novel probabilistic cluster-based reward mechanism to non-stationary multi-armed bandit reinforcement learning. MAGPIE rewards the sets of hyperparameters of its underlying isolation forest unsupervised anomaly classifiers based on the cluster silhouette scores of their output.**

Experimental evaluation in a real household shows that MAGPIE exhibits high accuracy because of two further innovations: it takes into account both cyber and physical sources of data; and it detects human presence to utilise models that exhibit the highest accuracy in each case. MAGPIE is available in *open-source format*, together with its evaluation datasets, so it can benefit from future advances in unsupervised and reinforcement learning and be able to be enriched with further sources of data as smart home environments and attacks evolve.

***Index Terms*—Intrusion Detection System, Cyber-physical attacks, Smart Home, Reinforcement Learning.**

I. Introduction

The mass adoption of IoT technology in smart homes has made them attractive targets to cyber threats, from unlocking doors and eavesdropping on occupants through their own cameras to hijacking voice-controlled personal assistant devices. Commercial trends for protecting against such threats revolve mainly around preventive measures, such as encryption or two-factor authentication, but the assumption that these measures are sufficient is not well-grounded [1], as vulnerabilities for IoT devices are discovered and exploited routinely despite them. In environments involving multiple devices of varying levels of trustworthiness and likely inter-dependencies between them, such as those found in smart homes, it makes sense to try to detect security breaches when they occur.

Intrusion detection is not new to the IoT [2]. In fact, several solutions have been proposed specifically for smart city and industrial IoT environments [3, 4]. Smart homes, however, present unique challenges with very specific requirements that can make generalist approaches unsuitable. They consist of multiple commercial off-the-shelf (COTS) devices, each often using a different network protocol, sometimes directly connected to the household's Wi-Fi router, other times connected indirectly through a specialised hub, and usually in an encrypted format. Users tend to develop their own automation rules that virtually link otherwise unconnected devices, including external ones, in unpredictable ways.

Furthermore, new vulnerabilities are discovered on a daily basis, and it is unrealistic to expect a smart home intrusion detection system to always be aware of all threats. Additionally, cyber-physical attacks (i.e., cybersecurity breaches that have adverse physical impact in the form of unauthorised, delayed, incorrect or altogether prevented actuation, or in the form of physical privacy breaches [5]) can affect domestic life and a person's behaviour and psychological state in their own home [6]. Different users have different risk attitudes in this context and would wish to configure differently any security measures protecting their smart home. Finally, in most cases, the cost of COTS smart home devices is relatively low, so any added security provision introduced should not itself require expensive equipment to run on.

We have addressed the above requirements by designing and implementing MAGPIE (**m**onitoring **ag**ainst cyber**p**hys**i**cal thr**e**ats), an intrusion detection system (IDS) prototype for smart homes subjected to a variety of cyber-physical security threats, both known and (at the time of execution) unknown. For a smart home IDS to be effective against unknown attacks and in changing conditions, it must be able to adapt. We argue that the configuration of an unsupervised classifier can be adapted continuously via reinforcement learning as it provides dynamic capability to continuously adapt an IDS configuration via conceptualisation of "actions" within a detection adaptation process, guided by learning the relationships between anomalous and normal cyber-physical behaviour in the environment. The challenge here is that an unsupervised IDS system cannot know the groundtruth (i.e., whether there really was an attack or not); thus, reinforcement learning cannot reward a classifier's specific set of hyperparameter values based on the groundtruth. However, in most attacks on a smart home, the less confident a classifier is, the more inaccurate it is in practice. Based on this observation, MAGPIE applies a simple idea for the first time: the reward function of reinforcement learning on an unsupervised classifier's hyperparameters can be based on the classifier's own confidence in its output, as expressed through its cluster silhouette scores. We have tested and confirmed the validity of this idea experimentally.

All authors are with the School of Computing and Mathematical Sciences, University of Greenwich, Old Royal Naval College, SE10 9LS, London, UK. E-mails: {r.heartfield, g.loukas, a.bezemskij, e.panaousis}@gre.ac.uk.

In addition, MAGPIE introduces three more innovations to ensure its practicality in a household, including taking into account users' risk tolerance, human presence and cyber-physical sources of data. In summary, MAGPIE implements the following contributions:

- Ability to **continuously adapt unsupervised smart home threat detection to changing conditions**. MAGPIE self-adapts by applying reinforcement learning on the unsupervised classifier's hyperparameters based on a probabilistic reward function without an a priori model or knowledge of the household configuration.
- Experimental evaluation with **both cyber and physical sources of data**. From a threat monitoring perspective, the physical impact of some security breaches constitutes an opportunity because, in conjunction with traditional cyber sources of data, it can provide valuable information about the system's security state.
- **Self-configuration based on automated inference of human presence**. In a smart home, the models of what is normal or not depend on human presence. For example, a voice-activated action being triggered when a human is present carries different significance to one triggered in the absence of a human.

We provide MAGPIE in open-source format for installation on a low-cost Linux computer, such as a Raspberry PI*.

II. Related Work

Traditionally, the vast majority of IoT security research applicable to current smart homes has focused on authentication and access control [7, 8, 9]. Lately, there has been a growing body of work tackling the challenge of detection, whether knowledge-based (utilising signatures of known attacks) or behaviour-based (detecting deviation from normal behaviour).

A. Knowledge-based smart home IDS

Anthi et al. [10] utilised standard machine learning classifiers, such as naive Bayes, to categorise IoT activity as normal or malicious. The features used were limited to network traffic and were similar to those used for non-IoT traffic, including timestamp, destination IP, protocol and packet size. In [11], the authors specifically classified which types of attacks have occurred based on supervised learning. This information can be very useful for triggering response mechanisms, but it is only applicable for known attacks and requires an extensive period of training under attack conditions (two weeks in the cited paper), which may be impractical for a household's smart home network.

Brun et al. [12] focused on detecting attacks on smart home IoT gateways. They employed a deep learning-based approach using dense random neural networks. However, the attacks utilised in the performance evaluation were simple TCP SYN denial of service attacks, which were shown to be almost as easily detectable by a simple threshold detector. Moustafa et al. [13] started with generalist datasets for botnets but enriched them with simulated IoT sensor data. Their learning approach was based on an Adaboost ensemble of decision trees, naive Bayes and artificial neural networks. However, the approach has not been evaluated with actual smart home devices and does not account for changes in usage patterns over time.

Nobakht et al. [14] employed a method based on software-defined networking technology, specifically OpenFlow, for providing modularity in intrusion detection for smart homes. Their experimental evaluation however was on a single light bulb, and the technique itself was based on known signatures of attacks, which limited its wider potential for large smart home setups or previously unseen attacks.

Trimananda et al. [20] addressed the specific challenge of information inference attacks in smart homes. Their tool is able to automatically extract packet-level signatures for device events based only on packet lengths and durations to predict which device is activated. Although very useful in anomaly detection, this approach has not yet been employed in this fashion. Additionally, it is naturally limited to attacks related to the unauthorised activation of devices.

B. Behaviour-based smart home IDS

Wan et al. [21] introduced IoTArgos, which in addition to supervised classification of the data communications of different smart home devices, has a "second stage" of detection using unsupervised learning for unknown attacks. This is a meaningful direction and has been evaluated on a wide range of COTS smart home devices. However, the cost of the two detection stages has not been evaluated, and the method does not take into account the presence of the user or the smart home's changing conditions.

A very interesting idea was developed in EclipseIoT [22], which in addition to authentication and access control, features an early detection provision based on canary files. These are forged files with enticing names (e.g., "SmartLock.py") placed amongst genuine ones. Modification of a canary file is an indication of unauthorised access.

Procopiou et al. [15] proposed a lightweight algorithm based on forecasting and chaos theory to identify flooding and DDoS attacks launched by compromised smart home devices. For every time-series behaviour collected, a forecast is generated, and the error of the forecast against the actual value is assessed by the Lyapunov exponent to determine if an attack has occurred. The evaluation conducted in NS-3 simulation involved low-rate and flooding attacks, but the method has not been extended beyond availability threats.

Novak et al. [16] proposed an intrusion detection technique that focuses on identifying unusually short and unusually long activities based on self-organising maps. While the approach of taking into account the length of activities proved to be useful, it is not sufficient by itself and can lead to considerable false positives.

Ramapatruni et al. [17] employed a hidden Markov model-based approach that learns what is normal in a smart home. In terms of context, if the user is recognised as being out (based on their mobile device's Wi-Fi connectivity), then any activity related to doors will result in an abnormal state. A strength of

*The code and datasets used here are provided at https://github.com/isec-greenwich/magpie

Table I: Limitations in existing intrusion detection research for IoT and Smart homes Vs. MAGPIE

IDS	Cyber sources	Physical sources	Self-configuration	Testbed
[10]	IP	✗	✗	Laboratory
[12]	IP	✗	✗	Laboratory
[13]	IP	✗	✗	UNSW-NB 15, NIMS datasets
[15]	IP	✗	✗	Simulation
[16]	ZigBee	✗	✗	Simulation
[17]	IP	Sensor readings (HTTP API)	✗	Smart home testbed
[18]	IP	✗	✗	Laboratory
[19]	IP, WiFi, BLE	✗	✗	Laboratory
[11]	IP	✗	✗	Smart home testbed
[20]	IP	✗	✗	Smart home testbed
[14]	IP	✗	✗	Smart home testbed
[21]	IP	✗	✗	Smart home testbed
MAGPIE	**Modular (IP, ZigBee, WiFi tested)**	**Modular (RF, Audio tested)**	**Continuous***	**Real household**

* via RL-based hyperparameter adaptation and Human presence inference

this work is that it can take into account the traffic generated by several diverse sensors, but it has been evaluated only in simulations in the form of artificial state changes.

Yamauchi et al. [23] expanded consideration of the user by modelling user behaviour as a sequence of events, including the operation of IoT devices and other behaviour monitored by sensors. Their method learns sequences of events for a predefined set of conditions and detects attacks by comparing the sequences of events, including the current operation, with the learned sequences. This work was extended in [18] and compared with a technique based on a hidden Markov model. It was tested on four users using smart home devices, but in a laboratory setting. Naturally, any legitimate behaviour that had not been previously observed would erroneously be flagged as anomalous.

C. *Critique of related work*

We observe that there is a wide variety of machine learning classifiers utilised in the literature, but there has been no emphasis on allowing configuration of intrusion detection beyond the design stage or based on the user's preferences. In addition, existing smart home IDSs have largely ignored the fact that cyber attacks in smart homes have an observable physical impact, which can be useful in detection. Finally, with the exception of [23], human presence has not been taken into account in smart home IDS research, although normal IoT device and network activity differ when the users are at home versus when they are not. In Table I, we provide an overview of the existing literature on intrusion detection approaches and MAGPIE, and in the following sections, we present, in detail, how MAGPIE addresses all four limitations.

III. MAGPIE DESIGN

Figure 1 summarises the MAGPIE architecture. Its *collection phase* captures and decodes the data coming from cyber (computation, communication) or physical feeds (e.g., audio, signal strength). It can dynamically activate or deactivate interfaces and decode the corresponding raw feeds, such as sensor readings or network datagrams.

Smart homes generate large volumes of usually encrypted data [24] that may differ considerably between different environments. In the *transcription phase*, MAGPIE considers only meta-data that are consistent across different smart homes. We argue that alternative approaches, such as authenticated and encrypted device API queries or passive interception of content with decryption keys, would render the defence mechanism a single point of failure and a target for attack. Moreover, by reading only smart home network communication flow meta-data, MAGPIE is better positioned to preserve privacy. MAGPIE extracts meta-data streams (MDS) based on specific interface datastream parsing logic (e.g., communication/application/sensor protocol) (Figure 2). Rolling window-based parser extraction and buffering allow appropriate performance and volume of data samples for processing. After aggregating, statistical information on the extracted meta-data features, such as the mean, standard deviation, min and max of sample frequency, content/message type, size, length, delay and flow direction, is considered. We define the delay meta-data feature as the inter-arrival rate in milliseconds between packets/frames for the same source-destination message type pairs.

Table II shows the volume and inter-arrival rate for samples collected in a 5-min window in our smart home testbed during periods of relatively low occupant activity with an aggregate average sample inter-arrival rate of 0.49 s and average sample volume of 3456, with extremes of under 1 ns between input samples in some cases. On the basis of these observations, it is clear that analysis on datastream samples in real time, without windowing and buffering, is impractical on a resource-constrained platform. Moreover, it may prove impractical to offload such volumes of data due to file size, upstream network bandwidth saturation and throttling [25].

Table II: Datastream sample inter-arrival time in seconds with no smart home occupant present (5-min capture)

Datastream	Samples	Avg.	Min	Max	Stdev.
IP	569	0.5369	0.00002	4.0338	0.9539
WiFi	3555	0.0892	0.0008	0.2048	0.1023
Sound	12465	0.0240	0.0121	0.0718	0.0146
Zigbee	390	0.7848	0.0002	5.3879	1.3882
RF	300	1	1	1	0

As part of windowing, a synchronised "end of window" datastream buffer is the stage where a parsing instance should initiate feature extraction, interpolation, discretisation and generation of statistical data (Figure 2). The datastream window in the buffer is then forwarded to a parsing logic and interpolation phase, where meta-data extraction and feature interpolation are performed on raw datastreams and are enumerated with protocol mapping identifiers (e.g., addressing, data type). The

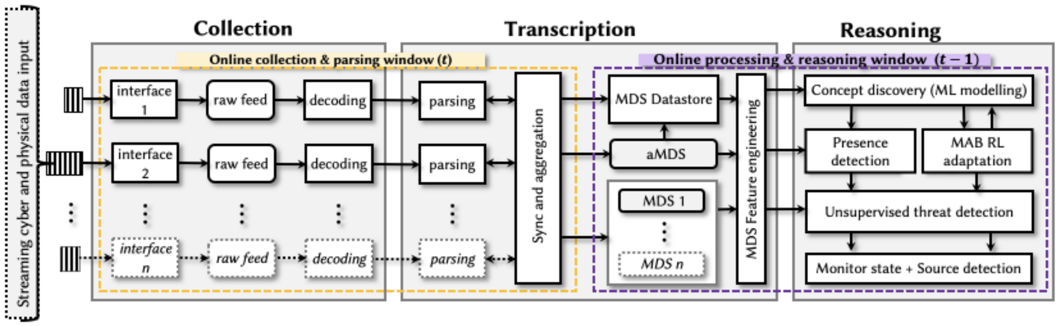


Figure 1: The *MAGPIE* architecture

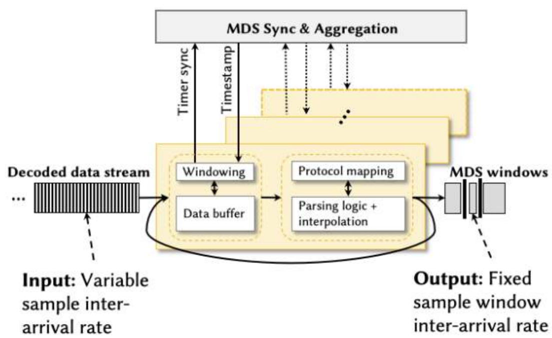


Figure 2: Parsing datastreams with variable inter-arrival rate

output is the MDS window feed, which is forwarded for storage to the MDS datastore to fuse data points across all MDS datastreams into an aggregated sample (aMDS). The datastore serves as a data historian for anomaly detection training (e.g., concept discovery) and captures snapshots of MDS data samples used in reinforcement learning-based adaptation (Section III-A). The aMDS fusion extracts common statistical features across all MDS feeds, which are later used to train a presence inference function within the smart home. The aMDS dataset combines common features across MDS feeds to generate a single feature-vector sample per window t for presence inference (whilst each MDS can have different sample rates for t) by omitting source-destination address and message type pairs for network data sources and compressing some physical MDS input. The average, mode, cumulative sum and standard deviation metrics are obtained for each feature extracted across each MDS feed.

The real-time threat monitoring latency is the window buffering latency plus the reasoning engine's prediction latency. All received MDS feeds are processed, interpolated, normalised and scaled in real time during each monitoring window interval. This process provides the required feature structures for concept discovery training data to learn "normal" behaviour and generate an independent anomaly detection model for each interface. Note that the complexity of the MAGPIE transcription phase is variable based on the cyber or physical data source, the sample rate and whether the data source is connection-oriented. For example, for network data sources (IP, WiFi, ZigBee), the computational complexity of the end-to-end parsing logic and interpolation is $O(n^{\delta})$, where n is the number of samples (or data set size) per window t and δ represents the computation of distinct source-destination pairs by connection address, port and message type. For physical data sources (RF and Audio), the computational complexity is $O(n)$. For training, the individual linear time complexity for each isolation forest model is $O(\zeta\psi\, log\psi)$ [26]. During real-time detection, the computational complexity of each isolation forest model is $O(n\zeta\, log\psi)$. From a technical implementation perspective, MAGPIE's processing efficiency is achieved by running parallel transcription processes for each data source and anomaly model. During the course of testing on a Raspberry Pi3, on average, the transcription phase did not exceed 1 s for each monitoring window t or 2.5 s for the end-to-end processing phases (collection, transcription and reasoning) at peak loads across five data sources.

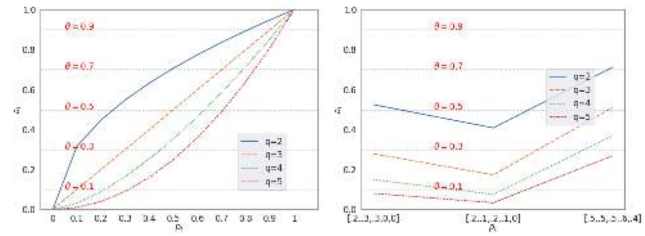
A. *Reasoning*

MAGPIE employs (i) real-time unsupervised anomaly detection, (ii) adaptation based on reinforcement learning, and (iii) model selection based on human presence inference.

1) Unsupervised anomaly detection: The first building block of MAGPIE's reasoning is the real-time detection of anomalies on individual interfaces. Here, supervised machine learning techniques are impractical because attack dataset labelling is unlikely to apply across households with different system configurations and different automation rules defined by their users. Let us consider the general case of an MDS feed $k \in [1, K]$ monitored during time window t.

During that time window, M samples are collected, A of which are classified by an anomaly detection process as abnormal and N of which are classified as normal, for example, based on an isolation forest [26] classifier, one-class support vector machines [27] or support vector data description [28]. We denote by A_t (resp. N_t) the number of *abnormal* (resp. *normal*), according to MAGPIE, samples investigated during time window t. Therefore, regardless of the choice of anomaly classifier used, for each MDS feed k in window t, we denote by $A_{k,t}$ (resp. $N_{k,t}$) the number of anomalous (resp. normal) samples found in feed k during time t. We then define the *anomaly ratio* $p_{k,t}$ as $p_{k,t} = \frac{A_t}{A_t + N_t}$.

Extending this approach across all feeds, we derive an aggregate *anomaly score* a_t for time window t. We have



(a) All models p_t equal or 1 model $\geq$0.1 with all others = 0 (b) Variable p_t across all models

Figure 3: q parameter anomaly score bias example

chosen

$$a_t = \left(\frac{\sum_{k=1}^{K} p_{k,t}^{q}}{\sum_{k=1}^{K} p_{k,t}} \right)^{\frac{1}{q-1}} \tag{1}$$

This transformation is required because different data sources can exhibit highly variable behaviour, and a single source's anomaly score is not a reliable means for determining an attack state. As described in [29], a simple approach, such as a weighted sum, cannot deliver reliable results because individual anomaly scores are typically contradictory. Therefore, we use the aggregate anomaly score a_t to address skewness. As square roots are commonly used for left skewness while cube roots are used for right skewness, the introduction of q allows for flexibility in the transformation. In practice, the control parameter q configures a higher, lower or balanced anomaly score bias across the ensemble. Favouring higher or lower scoring bias, however, can have an adverse effect on the anomaly score threshold θ defined by the smart home occupants for when to report a suspected attack. For example, in Figure 3, we demonstrate how an ensemble of five data sources, as in our experiments, can affect the detection accuracy if the user has defined a specific θ when q is too low or too high. For the top graph in Figure 3, for each aggregate score a_t (where all models are equal to the same p_t, or one model $p_t > 0$ with all other models $p_t = 0$), we show that depending on the user's defined θ, the value of q results in higher false positives or false negatives. For balanced accuracy, in this case, $q = 3$ is an optimal configuration for an ensemble of five anomaly models. In the bottom graph in Figure 3, we provide three general cases with variable anomaly model ratio scores p_t. If an occupant defines $\theta = 0.3$ as their attack threshold, then $q = 2$ would result in more false positives, while $q = 5$ would result in more false negatives. Thus, a middle-ground q parameter is preferable (again, in this example with five models, $q = 3$). Consequently, q must be increased or decreased as the number of data sources changes (where a minimum of two data sources, e.g., cyber + physical, is assumed and $K \geq 2$). In our experiments, we found that a middle-ground value, i.e., $q = \lceil \frac{K}{2} \rceil$, is appropriate and can be set automatically.

As $p_{k,t} \in [0, 1]$, this conveniently ensures that $a_t \in [0, 1]$. Thus, according to (1), the higher the $q \in [2, \inf]$ is, the more important one abnormal feed is to the overall anomaly score. As discussed, we have found experimentally that $q = 3$ provides a good balance between ensuring that the overall score does not unduly fluctuate between excessively high values caused by small numbers of anomalous MDS feeds (e.g., one cyber and one physical interface) or excessively low values caused by large numbers of normal MDS feeds (e.g., ten cyber/physical interfaces). Thus, for simplicity, equation (1) becomes $a_t = \sqrt{\frac{\sum_{k=1}^{K} p_{k_t}^{3}}{\sum_{k=1}^{K} p_{k_t}}}$.

Finally, the anomaly score is interpreted as an overall monitoring state $\mathcal{S}_t$ for the smart home, abnormal (if $a_t > \theta$) or normal (if $a_t < \theta$), based on an anomaly score *threshold* $\theta \in [0, 1]$, which is selected by the occupants.

In practice, θ is a threshold that represents the *risk profile* of the household. For example, a very high value, such as $\theta = 0.9$, would mean that the household would not want to be warned unless there are multiple strong indications of anomalies (risk-seeking profile). Intuitively, this is a household that would prefer to minimise false positives at the expense of a greater number of false negatives. In MAGPIE, a global θ threshold represents a single configuration parameter that occupants configure for attack detection. While model-specific θ definitions would increase the flexibility and control for the user, it would also increase the configuration complexity. First, users would require technical expertise to determine which θ to use for each data source, as it is unrealistic to assume a priori knowledge on the mapping between which θ values would bias one source over another. Furthermore, considering a preference for cyber or physical attack detection in cyber-physical IDS, it may be ineffective to define specific θ values that bias one cyber or physical source over another, specifically because attacks against either may be initiated via cyber or physical space [5]. By comparison, a global θ definition specifies a required threshold for an aggregate anomaly alert to be considered significant enough to be a substantive attack, regardless of whether the attack source is cyber or physical.

Further, to improve the processing of MDS samples as timeseries data, a sliding window of MDS samples is defined per MDS feed. The sliding window enables the anomaly score ratio calculation to take into account a previous window (or windows) of MDS activity.

2) Adaptation based on reinforcement learning: A newly installed smart home's dataset is typically free from contamination. After some time, however, the MDS datastore is likely to contain adversarial data samples from historic attacks or compromised devices. Therefore, it is important to adapt the learning process to cope with adversarial datapoints. This requirement is addressed naturally in certain unsupervised learning approaches through the application of contamination hyperparameters that adjust the decision threshold used for anomaly detection. Furthermore, what is considered normal in a household can change continuously as devices are added, removed or updated and as people add or remove automation rules or simply change how they use the devices. Therefore, changes in the datastream distribution require continuous adaptation of the anomaly detection threshold. Concept discovery is unique for each smart home, even for households with identical smart home configurations, because the network, sensor and actuation activity depends on the human factor [30]. Therefore, an unsupervised anomaly detection model

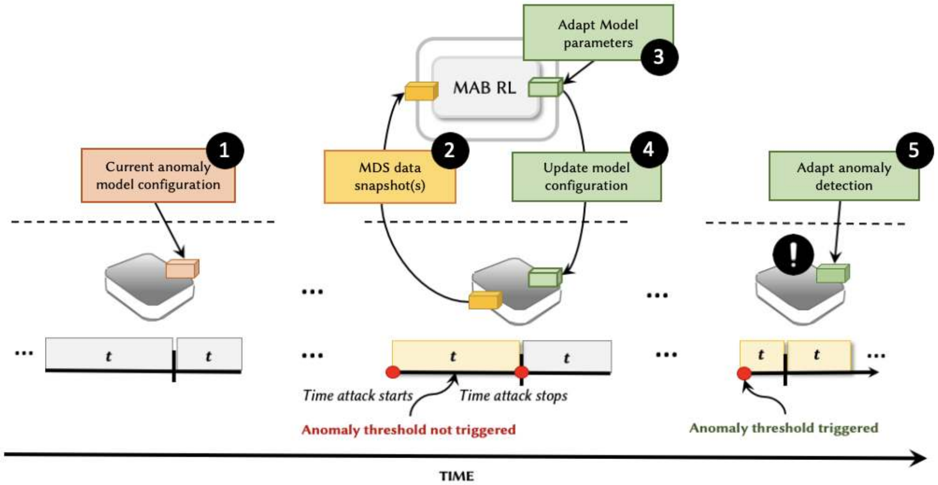


Figure 4: MAGPIE's reinforcement learning for threat detection re-configuration

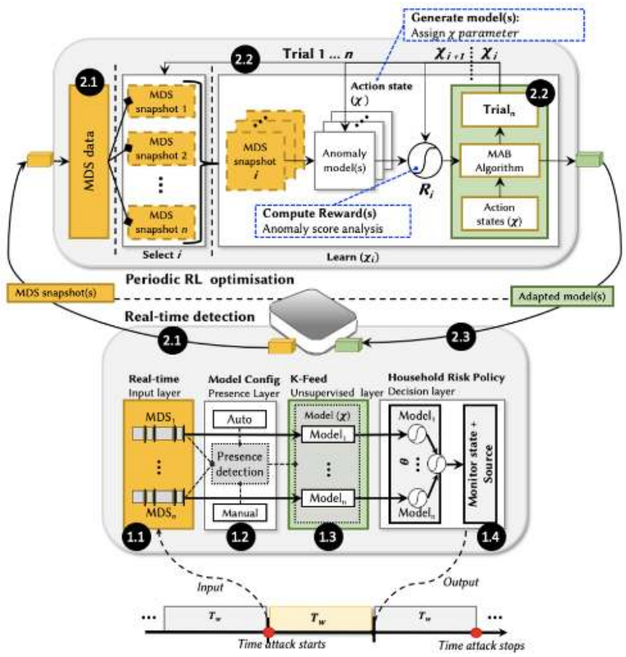


Figure 5: *MAGPIE Reasoning Engine* overview

developed for one smart home is not portable to another. As it is generally infeasible to obtain a priori knowledge of the correct contamination level, we propose the use of reinforcement learning to continuously update the anomaly classifier's hyperparameters.

The reinforcement learning mechanism in MAGPIE recursively explores and exploits detection reward feedback across different anomaly classifier configurations, where a single action-state (i.e., the anomaly classifier hyperparameter configuration) is selected during each step. The process treats the continuous capture and analysis of each MDS snapshot as an adversarial multi-armed bandit (MAB) environment [31, 32] because the composition of an individual dataset snapshot collected and analysed by MAGPIE (e.g., in terms of volume and data points) is continuously changing during real-time operation. This approach enables the anomaly detection to adapt to previously unseen data and to identify legitimate changes that occur in the environment that are expected to stabilise over time, whereas attack anomalies remain distinct because of their sparse occurrences.

For each MAB iteration, we define a probabilistic reward feedback based on the cluster silhouette scores of the anomaly detection results generated for each analysed dataset snapshot. In practice, the reinforcement learning process rewards the action-states (e.g., bandit arms) that reduce uncertainty in its own decision. Below, we describe the bandit environment, action-state parameters and reward generation algorithm:

[Bandit environment]: Defined as an adversarial bandit [33], where for each MAB action-state parameter iteration 1 to N (where N is the step horizon for a bandit episode), MDS data snapshot $i \in [J]$ is selected at random, and $[J]$ is the set containing all current MDS feed datastore snapshots.

[Action-state parameter]: The bandit arms are defined by the *anomaly model contamination hyperparameter* χ, which corresponds to the proportion of outliers in snapshot i used for anomaly modelling. The χ hyperparameter controls the anomaly detection decision threshold based on the anomaly detection classifier. For example, for our MAGPIE implementation, χ controls the decision threshold of an isolation forest classifier based on the decision function described in [26].

[MAB reward generation algorithm 1)]: The reward logic is as follows: for a given window $t \in [1, T]$, each MAB iteration corresponds to a given action-state (i.e., contamination hyperparameter) χ. We define $a_{t,\chi}$ as the anomaly score value of time window t for a contamination hyperparameter χ, and we denote $\vec{a}_\chi$ as the vector of all anomaly scores for χ for all the different time windows t.

Next, using K-means clustering with the Euclidean distance, we generate two clusters that contain higher and lower anomaly scores. We define the reward value $R_{\chi,i}$ for snapshot i as the silhouette score from the dataset clusters that represents a measure of cluster similarity.

Figure 4 shows a high-level illustration of the reinforcement learning role. During real-time operation, 1) the current anomaly model's configured detection threshold classifies

ALGORITHM 1: Algorithm for IDS RL Reward

Input : Anomaly scores produced by χ for each window in dataset i ($a_{t,\chi}$)
Output: MAB reward value $R_{\chi,i}$

Function `MABReward`:
  for $t \in [1, T]$ **do**
    $\vec{A}_\chi \leftarrow a_{t,\chi}$;
  end
  $k = 2$;
  $\vec{C}_\chi$ = KMeans($\vec{A}_\chi, k$) ;
  $R_{\chi,i}$ = Silhouette($\vec{C}_\chi$) ;
  return $R_{\chi,i}$;
End Function

MDS samples during each monitoring window t. Depending on either the number of samples collected or the time delta between collection periods, 2) MDS sample snapshots (e.g., M samples across K feeds) are sent to a "MAB RL" function to determine anomaly model hyperparameter χ based on the newly updated data sample of recent and historic MDS samples. Once the updated MDS samples are processed by the "MAB RL" function, 3) the model configuration computed by the RL process is issued as an updated model configuration (χ) for real-time anomaly detection. This process enables the anomaly model configuration to adapt its detection threshold via the RL process to respond to changes in the MDS data sample distribution and to discover previously unidentified threats.

Figure 5 shows the reinforcement learning process.

- **Real-time detection**. During each detection window t, following collection and transcription processing (1.1), the presence inference layer classifies the aMDS sample to determine the presence inference state (1.2) and then selects the most appropriate anomaly detection model to use for each subsequent MDS sample forwarded in the window (1.3). The user's risk threshold θ is used to compute the window's aggregated anomaly score by means of formula (1).
- **Continuous RL adaptation**. Once the sample input threshold (which is defined by the sample data size or time delta) is met, the received MDS samples are stored as a snapshot consisting of n windows in the MDS datastore set $[J]$ (2.1). MDS samples are selected at random during each MAB arm iteration. During each trial (2.2), each MDS model is trained with contamination hyperparameter χ (i.e., the MAB action-state parameter), with the MDS dataset excluding the randomly selected MDS snapshot i (i.e., the non-stationary bandit environment). Snapshot i is then used as test data for the trained anomaly model, producing an array of anomaly scores for each window in the snapshot, where the reward $R_{\chi,i}$ is computed by the reward Algorithm 1. Once the number of predefined RL trials has been reached, the χ hyperparameter "tuned" model is selected for use in the real-time anomaly detection process. The RL process is re-initiated once a new snapshot is stored.

To estimate the reliability of MAGPIE's reward mechanism, in section V, we experimentally evaluate two popular MAB algorithms that are commonly applied in non-stationary and adversarial bandit problems against a random selection method and compare their the cumulative average regret, cumulative average reward and average regret. We then proceed to establish the quality of the arm (χ) selection for the optimal bandit method using a fixed step horizon and evaluate the selection by generating an AUC-ROC model score for the corresponding isolation forest anomaly classifier configurations.

Monitor state and source detection - The state and source variables are intended to inform the household of whether the smart home is subject to anomalous behaviour after computing the detection score (e.g., under attack), as well as the MDS feeds with the highest anomaly score, which is indicative of the utilised attack vector (e.g., IP network, Zigbee network).

3) Human presence inference: Certain smart home system activity is observed irrespective of occupant presence. For example, a voice-controlled home assistant sends a continuous keep alive IP packet to the cloud regardless of whether occupants are using it. Other cases of system activity would be unusual if no human is present. For example, consider the case where network traffic from a ZigBee motion sensor increases dramatically or a voice command is activated even though no one is home. Therefore, it may make sense to train different machine learning models for the two cases of presence and no presence. Human presence inference can be based on a simple manual process, where occupants set it manually when they go to sleep or leave home, or it can be performed automatically before anomaly detection to select the machine learning model that corresponds to the presence state identified.

IV. PROTOTYPE IMPLEMENTATION AND SETUP

We have evaluated our prototype implementation by integrating it within the smart home of a real household with three members. Figure 6 shows the layout of the devices, which are described in Table IV, referenced by number ID in Figure 6, and accompanied by the smart home automation rules specified by the household, which are summarised in Table V. The setup includes a common home Internet router (2) with a WiFi LAN for WiFi-enabled devices (3-7, 13) and a ZigBee gateway (8) for Zigbee devices (9-12) connected to the home router via Ethernet. Remote connectivity to WiFi and ZigBee devices is facilitated by respective cloud services via the Internet. MAGPIE (1) collects all local and Internet traffic traversing the home router via an Ethernet SPAN port. Its WiFi and ZigBee interfaces passively monitor WiFi and Zigbee network frames on their configured RF channels. The software defined radio (SDR) interface captures spectrum readings in the respective WiFi and ZigBee 2.4 GHz ranges. MAGPIE's microphone is directly connected via USB. An adversary within wireless range of the smart home can execute ZigBee and WiFi attacks using an attack laptop, SDR peripherals and ZigBee antennas with customised firmware. The adversary can also target the smart home remotely via compromised cloud services or via command and control of compromised devices. See Table IX for a list and descriptions of the attacks.

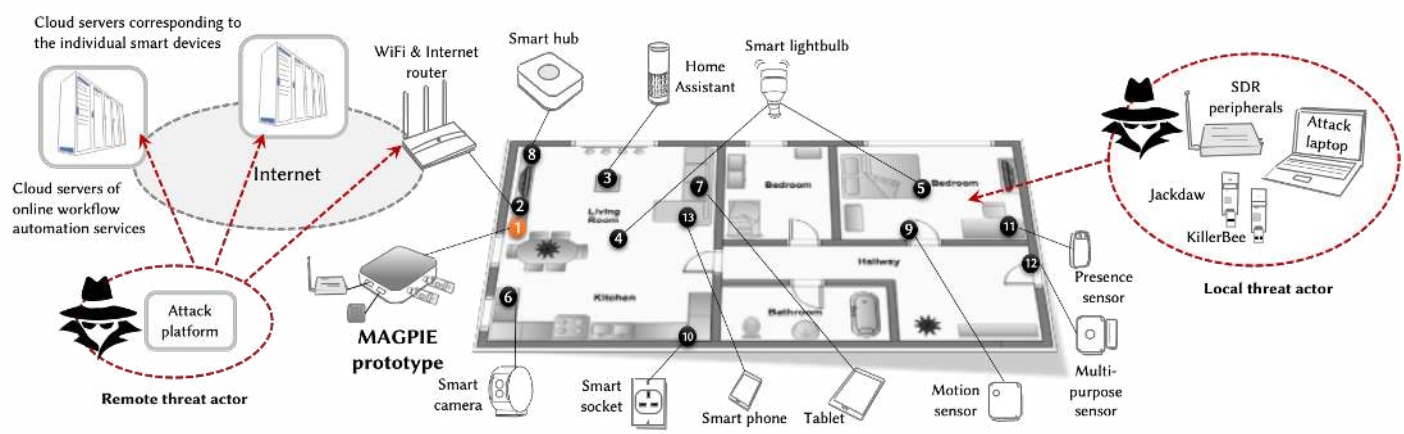


Figure 6: Smart home testbed for MAGPIE prototype experiments

Table III: Live capture sample dataset statistics

Dataset	Mean length	Attack States*	Stdev
45 x Normal	713.7 s	N/A	N/A
10 x A1	36.4 s	.524	0.007
10 x A2	39 s	.542	0.051
10 x A3	25.2 s	.527	0.038
10 x A4	37.8 s	.526	0.018
10 x A5	27.6 s	.572	0.033
10 x A6	25.6 s	.551	0.054
10 x A7	29.2 s	.615	0.283

* Average ratio of attack to normal states during attacks

Figure 7 presents the MAGPIE prototype's technical schematic. Each MAGPIE interface (1: Ethernet, 2: ZigBee antenna, 3: WiFi antenna, 4. Microphone, 5: SDR RF scanner) has individual data collection and parser processes managed by a window synchronisation daemon to forward all MDS datastreams to a message queue (ZMQ publisher) based on a defined time window. The ZMQ message queue forwards each MDS datastream to a datastore. Then, MDS pre-processing subscribers pull each MDS datastream from the ZMQ message queue, preprocess and forward the prepared MDS feature vectors to each respective MDS Isolation Forest model for anomaly detection and aggregated threat detection output. In parallel, the aMDS feed is forwarded to the random forest presence classifier for presence model selection. MAB RL adapted isolation forest models are trained, stored in the datastore and then loaded into the anomaly model selection and detection process after every MAB RL iteration.

The behaviour of occupants in the testbed and their interactions with the smart home devices and automation rules was allowed to occur naturally, with the addition of some requested actions to ensure that all automation rules or devices were activated during data collection. Training data collection was conducted intermittently during a 1-month period. The locations of the MAGPIE prototype and IoT devices remained static, with the exception of the mobile and tablet devices, which moved with the occupants using them. In total, there were 45 normal data collection runs with an average length of 713.7 s each and 70 attack data collection runs with an average length of 31.5 s each. In Table III, we provide summary statistics related to the datasets collected during the 1-month experiment.

A. *Cyber-physical meta-data features in the smart home*

In Table VI, we present each of the cyber-physical MDS feeds and the corresponding features collected. Further statistical flow information, such as sample frequency, average and standard deviation metrics, are added during parsing. We utilise tshark's display filter at run-time for standard input into the MDS parser, applying only regex operations to input data. Note that for physical data sources such as audio and radio frequency spectrum, we utilise custom (a python application) and open-source libraries (rx_power from rx_tools [34]) for feature collection. On the testbed, we apply the following constraints based on observation:

- WiFi data frames are redundant and ignored as they provide the network footprint, which is already monitored in encapsulated IP packets. WiFi "Request/Clear to Send" control frames are ignored as these are mainly used to avoid hidden-node collisions. Therefore, only WiFi management frames, which can be exploited to disrupt or infiltrate a WiFi network, are monitored.
- ZigBee sensors and actuators, with the exception of coordinator nodes (e.g., gateways), use dynamic network addressing. Therefore, all non-coordinator nodes are addressed using the same numerical value. This does not impact the ability to model anomalous ZigBee patterns as sensor and actuators generate a fairly predictable network footprint.
- Radio frequency spectrum analysis covers the 2.4 GHz frequency band for 802.11G and ZigBee, which can also include Bluetooth and other 802.15.4 wireless protocols.

1) *Risk-based unsupervised threat monitoring with reinforcement learning adaptation:*

- **Isolation forest anomaly detection**. We have opted to implement unsupervised anomaly detection using the isolation forest algorithm in Python with the Scikit Learn library [35]. It performs anomaly detection by isolating sample data points through random feature selection and value splitting, selecting a random value between the maximum and minimum bounds of a data sample feature. No prior assumptions are made regarding the distribution of feature values. Therefore, randomised feature splitting is effective for hybrid feature-sets of both continuous and categorical

Table IV: Smart home testbed devices and network connectivity

ID	Device Type	Interface	Description
1	MAGPIE prototype	ZigBee, WiFi, RF, Audio, Ethernet (IP)	Raspberry PI 3 “MAGPIE-enabled” prototype
2	WiFi router	WiFi 802.11G, Ethernet (IP)	Vodafone Broadband home router
3	Home assistant	WiFi (IP)	Amazon Echo voice-controlled home assistant connected to WiFi router
4, 5	Smart lightbulb	WiFi	LIFX smart light bulb connected to WiFi router & Amazon Echo via Cloud
6	Smart camera	WiFi (IP)	Somfy Protect smart camera connected to WiFi router & IFTTT via Cloud
7	Tablet	WiFi	Samsung S2 tablet connected to WiFi router
8	Smart hub	Ethernet (IP), ZigBee, Zwave	SmartThings hub connected nodes via ZigBee & WiFi router via Ethernet (IP)
9	Motion sensor	ZigBee	SmartThings motion sensor connected to smart hub via ZigBee
10	Smart outlet	ZigBee	SmartThings power outlet connected to smart hub via ZigBee & Amazon Echo via Cloud
11	Presence	ZigBee	SmartThings presence sensor key chain dongle connected to smart hub via ZigBee
12	Multi-purpose sensor	ZigBee	SmartThings multi-purpose sensor connected to smart hub via ZigBee
13	Smart phone	WiFi, Cellular (3/4G)	Connected to WiFi router (external connectivity to smart home control software via 3/4G)

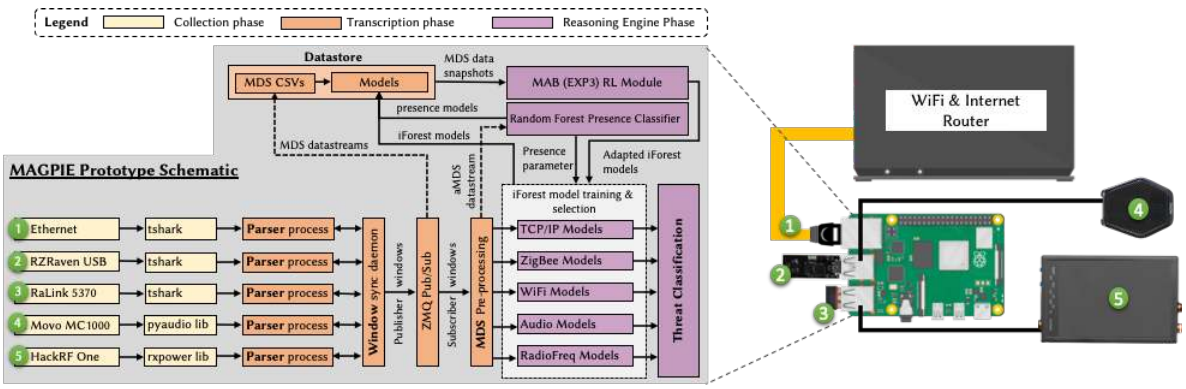


Figure 7: MAGPIE prototype technical schematic

Table V: Smart Home testbed automation integration and rules

Trigger Platform	Action	WAS	Antecedent
Amazon Echo	Voice trigger command	IFTTT	Somfy Protect actuation
Amazon Echo	Voice trigger command	Alexa Skills	LIFX bulb actuation
Amazon Echo	Voice trigger command	Alexa Skills	Outlet actuation
SmartThings	Mobile app “On” button	SmartThings	Outlet actuation
SmartThings	Door-open detection	SmartThings	SmartThings Multi-sensor
Amazon Echo	Motion sensor detection	IFTTT	Somfy Protect Arm/Disarm
LIFX	Mobile app “Bulb” button	LIFX	LIFX bulb actuation

WAS: Workflow Automation Service

Table VI: MAGPIE prototype data feeds and meta-data stream (MDS) features

Data feed	Input	Base Features
IPv4 (TCP/IP) pkts	C	src*, dest*, port, pkt type, pkt sz, ttl*, pkt delay, flow dir*.
WiFi 802.11 frames	C+P	src*, dest*, port, frame type, frame sz, rssi, frame delay
ZigBee (802.15.4) pkts	C	src*, dest*, pkt type, pkt sz, pkt delay
Microphone audio	P	rms frequency threshold, rms
RF (2.4GHz) Spectrum	P	† dB power level (per frequency bin)

C = Cyber, P = Physical, *aMDS: not used, † aMDS: Avg/Stdev of all bins

data, as is the case with MDS feeds. The recursive feature partitioning represents a tree structure, whereby the number of times a feature is split to isolate a sample follows a traditional tree path length from the root to a terminating node. The average tree path length represents the decision function used to classify observations as normal or anomalous [26]. For each MDS feed generated, an independent feed-specific isolation forest model is created. Together, the forests form an ensemble of models used to produce an aggregate anomaly score during each monitoring window.

- **Adversarial multi-armed bandit reinforcement learning**. MAGPIE models threat detection adaptation in a smart home as an adversarial bandit environment based on the premise that what is normal behaviour (e.g., devices, network traffic, user interaction) may frequently change in a smart home. Therefore, MAGPIE trains its RL-based anomaly classifiers on a continuously changing series of collected dataset snapshots. At each time step, for each arm pull (i.e., isolation forest χ hyperparameter selection), the smart home bandit chooses at random a dataset to test. Therefore, on the basis of algorithm 1, the distribution of reward $R_{\chi,i}$ for each action state arm is drawn from an i.i.d. distribution based on randomly selected MDS dataset snapshots. In this case, as each arm's reward distribution changes at random in the adversarial bandit environment, the EXP3 (exponential-weight algorithm for exploration and exploitation [33, 36]) algorithm is a natural and suitable choice to establish the optimal χ configuration for the isolation forest. For comparison against EXP3, we also select a non-stationary sliding-window based UCB (upper-confidence bound) algorithm [37], where the reward policy is weighted according to a constant step size used to update the reward estimate (we defined a step size of 0.1, which moves the agents estimate 10% closer to the most recent observed reward). UCB is a popular choice for traditional stationary MAB reinforcement learning problems, achieving

MAB Algorithm	Parameters
EXP3	γ = 0.1, arms* = 10, iters.†=6000
Non-stationary UCB1	exploration (C) = 2, λ = 0.1, arms* = 10, iters.†=6000

*hyperparameter configurations, † action-step horizon

Table VII: EXP3 and non-stationary UCB1 MAB parameters

Table VIII: MAGPIE prototype *Reasoning Engine* model configuration parameters

Anomaly detection (Unsupervised learning) configuration	
Algorithm	Isolation Forest
Trees (ζ)	200 (per MDS model)
Sub-sampling (ψ)	250 (per MDS model)
Bootstrap	Sampling without replacement
Max Features	All features (randomly selected per split)
Contamination (χ)	Bins = 0.001, 0.002, 0.005, 0.01, 0.2, 0.05, 0.1, 0.2, 0.5, 0.7
θ	0.1, 0.3, 0.5, 0.7, 0.9
Presence inference (Supervised learning)	
Algorithm	Random Forest
Trees (ζ)	100
Sub-sampling (ψ)	250 (per MDS model)
Max Depth	15
Bootstrap	Sampling without replacement
Max Features	Auto (max features=$\sqrt{features}$)
θ	0.3, 0.5, 0.7
Detection adaptation (Reinforcement learning) configuration	
Algorithm	EXP3 (γ=0.1, 6000 step-horizon)
Reward metric	As per section III-A
Presence	1 (Activity), 0 (No activity)
MAB Arms	10 - (Contamination χ Bins)

logarithmic regret for the number of actions/arm pulls (in this case, χ parameters) selected over time [38], where regret refers to the expected decrease in reward gained during execution of the learning algorithm instead of acting optimally [39]. In other words, the regret is the difference between the reward of a given policy (i.e., the learning algorithm) and that of the optimal static policy in hindsight. In this case, to adjust to stochastic, non-stationary bandit behaviour, a discounting factor (λ) based on a step-size sliding window is applied to a UCB1 policy reward estimate. The EXP3 and non-stationary UCB1 implementations were adapted from the bandit algorithms developed in [40] and [41], respectively. In Table VII, we summarise the configuration parameters for the EXP3 and non-stationary UCB1 algorithms.

2) Human presence inference: In our implementation, the presence of people is detected with a supervised random forest (RF) classifier using ground-truth labels defined by the user, as RF is commonly used for lightweight machine learning in the IoT [42]. Other lightweight supervised machine learning classifiers are similarly useful. Labelling of aMDS data samples (user presence (1)/no user presence (0)) is pre-configured for the initial training of the prototype during the start-up learning phase. Subsequent training requires users to actively inform MAGPIE of the time periods in which they are actively present in the household (unsupervised presence inference is outside the scope of the prototype development). We define a simplified household environment for presence inference based on whether occupants are actively or passively interacting with the smart home network. By observing the behaviour of a sub-sample of collected aMDS data points (Figure 8), we see a distinguishable impact of presence and no presence for most datastreams. However, aside from other physical sources, basic sound measurements as a feature may be problematic in the face of an audio injection attack. We evaluate this impact on automated presence inference in section V-A, where we also assess overall detection results, showing that presence inference helps increase accuracy.

B. Smart home cyber-physical attack vectors

We have subjected our testbed to attacks targeting WiFi, ZigBee and voice-enabled home assistant communication technologies, as well as corresponding smart home device control software and third-party apps, all of which are commonly deployed within today's smart home environments. WiFi is currently the primary connectivity medium in most smart homes, not only for device-to-device communication but also as a network gateway to the cloud services on which most smart home devices rely. ZigBee is a low-powered wireless medium that provides energy-efficient connectivity for low-resource devices that connect to more capable control gateways (e.g., ZigBee hub with Ethernet or WiFi backhaul). However, it has limited bandwidth for data communication (250 kbit/s per channel in the 2.4 GHz band used in the testbed). Security-wise, the speaker-microphone pair of a voice-enabled home assistant is typically an unmonitored communication link, which has been shown to be vulnerable to exploitation [44]. Smart home devices such as security cameras offer physical monitoring protection of the household; however, recent high-profile compromises of these types of systems have demonstrated how their exploitation can lead to significant breaches of the physical privacy of occupants and, in some cases, impact their emotional well-being [6].

In Table IX, we describe the attack vectors and their cyber-physical impact based on [5]. All attacks were executed in both the presence and no presence conditions. Note that localised attack vectors (namely, WiFi deauth (A1), Evil twin (A2), ZigBee jamming (A3) and Node amplification (A4)) could also be launched remotely if a target device were compromised through third-party apps, cloud-based control software, or compromised software and hardware supply chains [6]. For example, both home assistants and smart lightbulbs provide the ability to host their own WiFi access point, whilst ZigBee devices are capable of reconfiguring themselves as ZigBee network coordinators.

C. Experimental scenario, settings and parameters

Our experimental process consisted of three phases. Phase 1 was related to (i) live sample data collection of smart home behaviour (in terms of the data sources monitored) when not under attack and (ii) execution of each attack vector. This phase comprised two different types of experiments: one where users were present during data collection and another where no users were present in the household. Phase 2 was related to the adaptation of the offline reinforcement learning anomaly detection. Phase 3 was related to live monitoring of attack detection using the RL-optimised MAGPIE configuration.

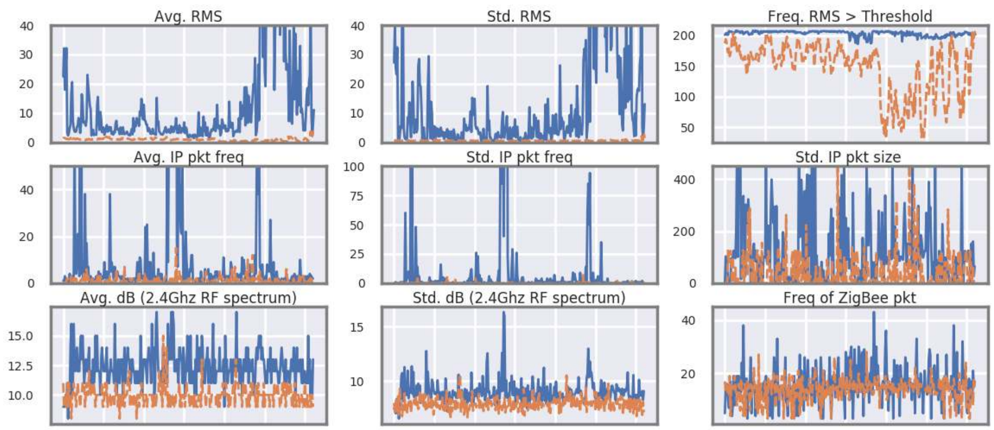


Figure 8: Aggregated MDS (aMDS) behaviour during “No presence” (orange dashed line) and “Presence” (blue line) states.

Table IX: Experiment attack vectors with cyber-physical impact classification [6] in the smart home

Attack*	**Layer**	**C***	**P***	**Description**
(A1) WiFi deauth	Data Link	A	PA, DA	WiFi deauthentication frame flood, resulting in **prevented or delayed actuation** and DoS for all WiFi hosts. Executed with *aircrack-ng* suite.
(A2) WiFi Evil Twin	Data Link	C, I, A	PA, IA	Evil twin (ET) spoofs WiFi network disrupting WiFi-connected devices; entices connection to attacker-controlled access point. Results in **prevented actuation** through WiFi beacon frame interference and **incorrect actuation** for ET-connected devices under the control of the attacker. Executed using the *aircrack-ng* suite.
(A3) ZigBee jamming	Physical	A	PA	ZigBee communication is jammed on current radio frequency. Results in **prevented or delayed actuation**. Executed with the RZRAVEN USB Stick flashed with Jackdaw firmware [43].
(A4) ZigBee node amplification	Network	I, A	DA	Targets a vulnerability in *Samsung SmartThings* smart outlet, which acts as a router/relay in the ZigBee PAN. An unsolicited ZigBee data request sent to the *SmartThings outlet* returns four encrypted data packets in response. Replay of a doctored PCAP containing a large volume of data requests triggers exponential traffic amplification against the outlet. The resulting volume of data packets returned quickly overwhelms the *SmartThings* network bandwidth resulting in **prevented or delayed actuation**. Executed using the RZRAVEN USB Stick with KillerBee suite https://github.com/riverloopsec/killerbee. After our ethical disclosure to the manufacturer, the vulnerability has since been patched.
(A5) Malware audio injection	Physical & Application	I	UA	Compromised smart device (Samsung Tablet) with its on speaker-microphone pair injects malicious commands into the *Amazon Echo* home assistant, eliciting **unauthorised actuation**. The commands continuously arm or disarm the smart camera and turn on or off any household lights. A second-order impact is the DoS of smart home systems that may also be used to detect physical intrusion. A remote command and control channel directs the execution of the audio injection. Executed using *Stringify* API as the command and control channel to a custom Android app that plays and records audio (for audio-based camera actuation).
(A6) Security camera compromise	Application	C	UA, BP	Compromised smart home security camera user credentials (e.g., phishing/insecure network) used to create a remote connection to the camera video feed (**breaching physical privacy**). Executed using the user credentials to obtain an authentication token issued by the *Somfy Protect* API.
(A7) Workflow automation compromise	Application	C	UA, BP	Compromised smart home IFTTT user credentials (e.g., phishing or insecure network). Issues actuation commands via occupant workflow automation rules (“Arm/Disarm camera”, “Turn on light bulb”, etc.), disrupting smart home devices via **unauthorised actuation**. Smart outlet, camera and bulbs are continuously issued actuation commands at high frequency; light bulb set to “flicker”, which could lead to medical impact in the form of seizures for occupants with photosensitive epilepsy or to electrical damage caused by surges in voltage.

***See attack cyber-physical impact graphs: https://github.com/isec-greenwich/magpie**
C* (Cyber impact)- C: Confidentiality, I: Integrity, A: Availability
P* (Physical impact)- PA: Prevented Actuation, IA: Incorrect Actuation, UA: Unauthorised Actuation, DA: Delayed Actuation
BP:Breach of physical privacy

Table III provides statistics about the live capture sample dataset for normal and attack execution experiments. Some attack vectors (WiFi de-authentication and ZigBee jamming) were observed to have a persistent effect on specific device behaviour, such as total connectivity loss to the WiFi network or disconnection of ZigBee nodes from the PAN, even after the attack had stopped. To ensure that persistent symptoms of one experiment did not interfere with another, after each attack execution, we reconnected affected devices and nodes to their respective networks and tested the automation rules to ensure that the smart home had returned to a known good state. For phase 1, each attack vector was executed independently so that normal and attack data samples were equally distributed with respect to the amount of time the smart home was monitored by MAGPIE under normal conditions and during attack execution. This process ensured that the captured dataset had a balanced set of normal and attack samples for testing. All live sample collection experiments were conducted on the training data for phase 2 reinforcement learning adaptation of the MAPGIE’s anomaly models, whereas phase 3 consisted of executing live attack vectors against the MAGPIE prototype in a real-time monitoring state with the optimised anomaly model configuration. During the experiment, the users interacted with the smart home according to their normal routine. This activity generated a dataset that represented natural smart home user behaviour. Table X shows the different types of interactions

Table X: Summary of occupant device and network interaction in the smart home testbed

Device / Platform	Action	Activity description
SmartThings Multi-Sensor	P, T, S	Opening/closing door & status check on app
SmartThings Motion-Sensor	P, T, S	Moving in range of motion sensor & status check on app
Smart Outlet	P, T, S	Turning on/off via button & app & status check on app
Amazon Echo (via Voice)	P	Asking questions, playing music, triggering LifX, Somfy
Amazon Echo (via app)	P, T, S	Playing music, Amazon Echo activity & changing Alexa skills
Somfy Protect	P, T, S	Reviewing camera feed, triggering security mode via Somfy app
LifX lightbulbs	P, T, S	Triggering LifX bulbs via the LifX mobile app

*P=Physical, T=Tablet, S=Smart phone

P*	θ	χ Mode*	ACC	TPR	TNR	PPV	NIR
✓	0.3	Random†	0.67	0.78	0.55	0.66	0.60
		RL	**0.85**	**0.99**	**0.82**	**0.64**	**0.60**
✗	0.1	Random†	0.68	0.79	0.56	0.68	0.61
		RL	**0.87**	**0.90**	**0.87**	**0.83**	**0.61**

Table XI: Average detection accuracy for RL χ adaptation Vs. randomly selected χ, P* Occupant presence in smart home testbed, † Average over 100 runs, selecting from 10 bandit Arms (i.e., χ bins - see Table VIII)

performed by the users.

V. Experimental results

The MAGPIE prototype's performance is evaluated in terms of the i) attack detection accuracy with reinforcement learning χ adaptation Vs. random χ configuration, and the effect on performance with and without cyber-physical sources of data; ii) accurate detection of presence in the smart home and its effect on threat detection performance for dynamic anomaly model selection; iii) attack detection latency, which refers to the time delay before the prototype system correctly identify an attack, taking into account correct interface *source detection*; and iv) end-to-end monitoring latency, which measures the processing delay for each of the MAGPIE prototype's *transcription* and *analysis* phases to complete, according to the prescribed collection window interval.

For each measure of detection performance, we employ timestamp window labelling to indicate whether a predicted data sample's label belongs to an attack window or a non-attack window. In terms of accuracy, the Jaccard similarity coefficient is used as a measure of prediction performance to compare a set of predicted data sample labels to a corresponding set of ground truth labels.

A. *(Contribution 1) Ability to recognise new smart home threats by continuous adaptation to changing conditions*

At its inception, a newly installed smart home can be safely assumed to be free from attacks. Therefore, a low anomaly detection sensitivity (e.g., χ - contamination value for isolation forest anomaly decision function) is a sensible choice for system initialisation. However, over time, the level of data contamination supporting this condition will drift due to changes in the smart home configuration or actual attacks (which may be undetectable at the time of occurrence due to the current detection sensitivity). The same applies to selecting the anomaly detection sensitivity for an existing smart home.

In Figures 11 and 10, the experimental results of the RL training show that EXP3 achieved the lowest average cumulative regret and highest average cumulative reward compared to a non-stationary UCB strategy, whereas both EXP3 and UCB comfortably outperformed a naive random arm selection strategy. In terms of cumulative reward, initially, minor increases in observed reward occur due to the relatively small distribution range between rewards (see Figure 10). The effect on performance for both EXP3 and UCB therefore indicates that a sufficiently large step horizon is required to reach an optimal and reliable arm selection state, as per the objectives of the MAGPIE MAB RL process. Following our experimental comparison of MAB algorithms, EXP3 was selected as the optimal bandit algorithm for solving the anomaly classifier detection adaptation objective in MAGPIE. Following a fixed-step horizon selection policy, in Figure 12, EXP3 reported optimal arm weights χ=0.01 (ARM 4) and χ=0.005 (ARM 3) for the presence and non-presence anomaly classifier configurations, respectively. To analyse the quality of the EXP3 selected χ configuration parameters, we derived AUC-ROC curves for models trained with each specific χ parameter in Figure 13. The results show that for overall model detection accuracy, when applying the θ threshold across all attacks we evaluated, EXP3 selected the optimal χ parameter for the presence and no-presence anomaly models. In terms of overall AUC, the selected arms were ranked first (AUC=0.90) and third (AUC=0.88) for the presence and no-presence models, respectively.

In summary, the results presented in Table XI demonstrate that the combination of EXP3 with our probabilistic reward algorithm is a reliable mechanism for optimising detection performance. Figure 9 shows RL optimised the unsupervised detection accuracy for each θ when tested against the attack vector in our testbed.

Analysis of the presence datasets (orange radar) shows poor detection accuracy for attack A5 (malware-enabled audio injection; ACC: 60% Vs. 55% no information rate - NIR). This decrease in accuracy when occupants are present is likely due to the attack pattern of A5 blending in with the occupants' own use of the home assistant. Therefore, to improve detection of A5 in future work, more expressive features are required (such as voice command recognition or individual modelling of sound and IP/WiFi traffic mean absolute deviation according to the time of day) to provide greater behavioural context to the anomaly detection process. Omitting A5 from the aggregated results yields an overall detection accuracy for occupant presence models of 89%, an F1 score of 81%, TPR of 99%, TNR of 84% and precision of 72%. On the other hand, A5 is easily detectable by no-presence models (for the static presence model configuration) due to the abnormal occurrence of sustained levels of sound, with a detection accuracy of 93%. The no-presence model achieved an F1 score of 85%, TPR of 90%, TNR of 87% and precision of 83%. For individual attack adaptation, RL also reports fairly low accuracy for attack A7 in the presence models and attack A6 in both the presence and no-presence models.

In practice, the optimal θ for presence and no-presence threat detection may not be selected as the preferred value. Importantly, the results show that a range of different θ settings influence the RL adaptation process, whereby high θ values

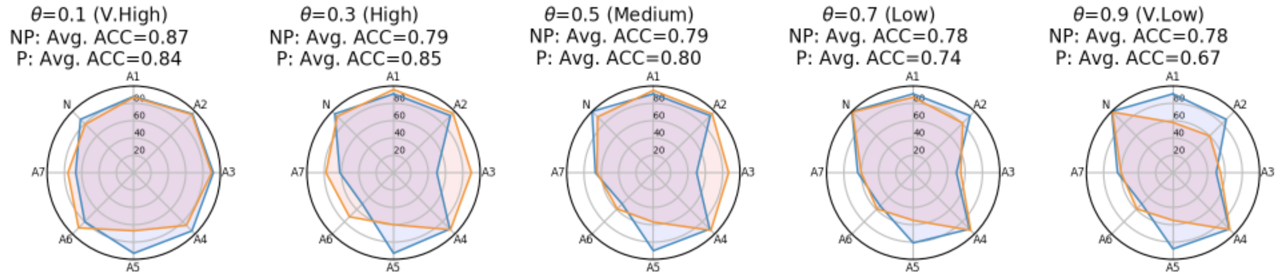


Figure 9: Risk-based RL-optimised anomaly detection accuracy [%] (Presence = Orange, No Presence = Blue, MAB Arm 3: χ = 0.005, MAB Arm 4: χ = 0.01)

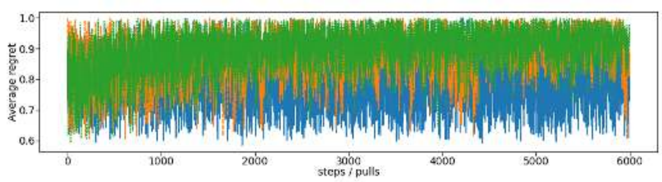
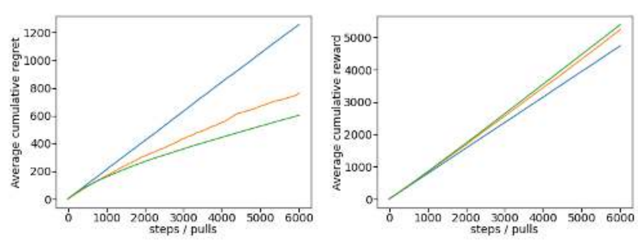
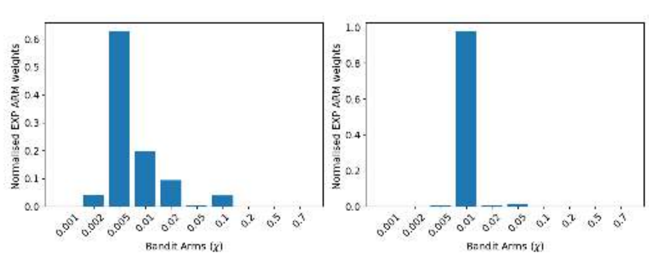


Figure 10: Average cumulative χ selection regret for random (blue), non-stationary UCB (orange dash), and EXP3 (green points) MAB algorithms over a 6000 step horizon across 3 runs



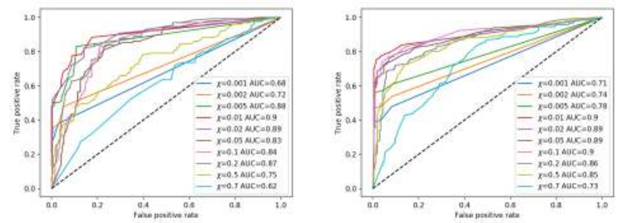
(a) Average cumulative regret (b) Average cumulative reward

Figure 11: Cumulative χ selection reward and regret for random (blue), non-stationary UCB (orange) and EXP3 (green) MAB algorithms over a 6000 step horizon across 3 runs



(a) No-presence arm weights (b) Presence arm weights

Figure 12: EXP3 weights for bandit arms (χ) over a 6000 step horizon



(a) No-presence AUC ROC (b) Presence AUC ROC

Figure 13: AUC ROC curves for MAGPIE bandit arms (χ hyperparameters

(0.7 and 0.9) are the least effective for both presence and no-presence models. In general, high θ favours high anomaly scores, which reduces false positives but may increase false negatives. This characteristic can be observed for attack A7 (workflow automation compromise), where MAGPIE reduces each set of attack data points to a single sample per window (based on source and destination identity), which in turn is saturated by a high volume of normal traffic, thus lowering the anomaly score. Overall, the detection results illustrated in Figure 9 demonstrate that the non-stationary UCB implementation is effective at adapting the detection sensitivity to optimise the threat detection performance according to the occupants' θ configuration. From here on, we assess the MAGPIE prototype's threat detection performance according to the best-performing θ and RL optimised isolation forest χ parameters.

B. (Contribution 2) Considering both cyber and physical sources of data

Applying the best θ and χ parameters, for both individual and aggregate attack dataset RL adaptation, in Figure 14 the detection accuracy for the presence and no-presence models across different MDS cyber and cyber-physical feature models is presented. Here, MAGPIE prototype threat detection is demonstrably more accurate, on average, when both cyber and physical smart home data sources are used compared to cyber features only. However, even without physical features (in this case RF, audio and WiFi RSSI), extending the collection of cyber features beyond traditional monitoring of TCP/IP traffic significantly improves the detection accuracy across a wide range of attack vectors in the smart home.

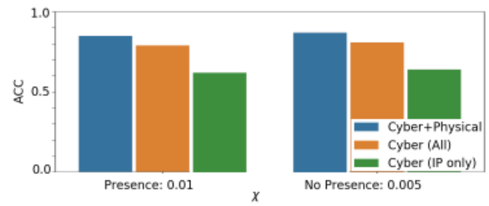


Figure 14: Attack detection performance results comparison for MDS models with cyber+physical features, multiple cyber features and cyber features based on the TCP/IP stack only

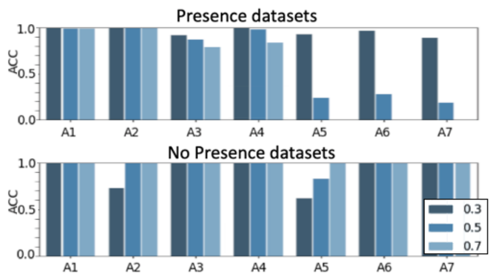


Figure 15: Presence inference accuracy for each attack (Avg. ACC: θ 0.3=0.93, θ 0.5=0.81, θ 0.7=0.76)

C. (Contribution 3) Self-configuration based on automated inference of human presence

Figure 15 shows that presence inference returned high classification accuracy during both attack and non-attack scenarios. A choice of $\theta = 0.3$ yielded the highest overall accuracy (93%) across both the presence and no-presence datasets. However, there is a noticeable accuracy drop compared to static detection model assignment for correctly detecting presence during the audio injection attack (A5) when there is no presence (62%). This is because the random forest detection model has determined higher audio values to be associated with presence state and thus incorrectly identifies the audio injection attack as occupant presence. Consequently, this failure has a negative impact on the detection of audio injection with a high sensitivity for presence inference. On the other hand, whilst increasing θ to 0.5 increases the detection accuracy for audio injection during no presence (83%; increasing further to 97% for $\theta = 0.54$ - not shown in Figure 15), this change has a negative effect on detection accuracy for attacks A6 and A7 and further reduces the A5 detection accuracy during the occupant presence state.

Figure 17 shows a noticeable advantage of dynamic reconfiguration based on presence inference. When utilising the best high and very high θ values for presence and no-presence anomaly models, respectively, the method achieves slightly lower detection accuracy overall (significantly lower in the case of audio injection - A5 for non-presence) compared to static model assignment (which requires explicit occupant reconfiguration to function, e.g., the occupant informing the system when they are no longer present or active). Crucially, however, without static or dynamic anomaly model assignment, for both the presence and no-presence models, individual detection performance for alternate datasets is considerably worse overall.

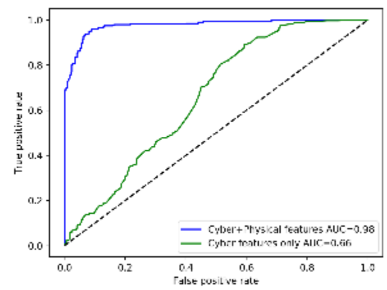


Figure 16: AUC ROC curve (TPR Vs. FPR) performance for presence inference during smart home attacks

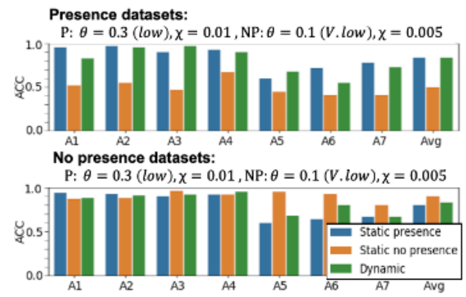


Figure 17: Performance for static presence, no presence and dynamic real-time presence anomaly model selection

The experimental results are promising, especially as the fusion of cyber and physical MDS features has proven to be valuable for improving presence inference, as further evidenced in the area under curve (AUC) receiver operating characteristic analysis on new, unseen MDS data shown in Figure 16. Compared to the AUC score of 0.66 when only cyber data sources are used, the aggregation of cyber and physical data sources (i.e., the aMDS feed) yields an AUC of 0.98 for presence inference. Here, instead of training a single MDS model for both presence and anomaly detection, affected by increases in model dimensionality and feature-masking for window synchronisation, these results demonstrate that a dedicated presence classifier supports the selection of presence-specific MDS models that directly benefit from the feature context to detect attacks more accurately.

D. Threat detection latency

Analysis of the detection latency using the best θ and χ RL parameters for the presence and no-presence datasets produced variable results for each attack. These differences were expected, as the impact of different attacks on cyber and physical feature behaviour may only become noticeable later in the course of execution. For attacks A1, A2 and A7, during human presence, detection was immediately triggered (detection latency = 1 monitoring window) when the attack was executed in the collection window, but the initial discovery of A3 and A4 was three times slower and that of A6 was five times slower. In the presence condition, A5 was undetectable. In the no human presence condition, attacks A2, A4 and A5 reported immediate detection, A1 and A3 required an additional collection window for identification, and A6 and

A7 were four and three times slower, respectively. MAGPIE has demonstrated that it is able to detect an attack soon after it is executed, but there remains a trade-off in detection latency and detection accuracy to be explored in the future.

VI. Future Work

In future work, the q parameter is an interesting and potential candidate for further exploration within the RL action-space for dynamic q assignments, alongside unsupervised model hyperparameter adaptation. However, a consideration when introducing q in this manner is that it increases the computational complexity for the RL action-space. As observed in Figure 3, the benefit of the increased complexity does not clearly outweigh the simpler static q definition. Therefore, dynamic determination of the optimal q is an area that would ideally be explored in the context of collaborative learning, such as federated threat detection adaptation in experiments across multiple coordinating households and MAGPIE agents, with varying MDS configurations and contrasting attack vectors.

It would also be interesting to explore how MAGPIE's detection adaptation might be applied in the form of cyber resilience capability for machine-learning-based intrusion detection itself, for example, as a proactive defence mechanism against emerging adversarial machine learning attacks [45] that disrupt detection accuracy in cyber-physical systems.

VII. Conclusion

We have evaluated MAGPIE in terms of four primary contributions: the ability to detect previously unseen attacks while taking into account the user's risk tolerance; the ability to adapt to changing conditions via reinforcement learning; the benefit of using both cyber and physical sources of data; and self-configuration of the choice of models based on whether user presence is detected or not. The prototype has performed well across a range of attack vectors at the application, network, data link and physical layers. We have observed that the incorporation of physical sources of data can noticeably improve the performance for most of the attacks, especially for attacks that are normally undetectable by systems that monitor only TCP/IP traffic. We have also observed that by leveraging the same data sources as for anomaly detection, we can detect user presence sufficiently reliably, which in turn helps tailor the anomaly detection models to the two cases of presence and no presence, thereby improving their accuracy. Most importantly, we have successfully tested our intuition that in the context of smart home attacks, reinforcement learning can meaningfully adapt an unsupervised anomaly classifier's hyperparameters based on its own confidence in its output.

We have made the source code of the MAGPIE implementation available to the research community to facilitate extensions and experimental evaluation comparisons with new methods. A natural extension would be to add real-time response capabilities, such as isolating offending nodes or reconfiguring the radio frequency channel. Additionally, MAGPIE can be extended to incorporate feedback from the user, for example, to confirm whether a suspected anomalous device or network behaviour is a result of their own activity or not, to further improve the accuracy.

References

[1] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella. IoT internet of threats? A survey of practical security vulnerabilities in real ioT devices. *IEEE Internet of Things Journal (early access)*, 2019.

[2] M. Conti, A. Dehghantanha, K. Franke, and S. Watson. Internet of things security and forensics: Challenges and opportunities, 2018.

[3] D. Formby and R. Beyah. Temporal execution behavior for host anomaly detection in programmable logic controllers. *IEEE Transactions on Information Forensics and Security*, 15:1455–1469, 2019.

[4] A. Ameli, A. Hooshyar, A. H. Yazdavar, E. F. El-Saadany, and A. Youssef. Attack detection for load frequency control systems using stochastic unknown input estimators. *IEEE Transactions on Information Forensics and Security*, 13(10):2575–2590, 2018.

[5] G. Loukas. *Cyber-physical attacks: A growing invisible threat*. Butterworth-Heinemann, 2015.

[6] R. Heartfield, G. Loukas, S. Budimir, A. Bezemskij, J. RJ. Fontaine, A. Filippoupolitis, and E. Roesch. A taxonomy of cyber-physical threats and impact in the smart home. *Computers & Security*, 78:398–428, 2018.

[7] M. Alshahrani and I. Traore. Secure mutual authentication and automated access control for IoT smart home using cumulative keyed-hash chain. *Journal of Inf. Security and Applications*, 45:156–175, 2019.

[8] B. Chifor, I. Bica, V. Patriciu, and F. Pop. A security authorization scheme for smart home Internet of Things devices. *FGCS*, 86:740–749, 2018.

[9] B. Baruah and S. Dhal. A two-factor authentication scheme against fdm attack in IFTTT based smart home system. *Computers & Security*, 77:21–35, 2018.

[10] E. Anthi, L. Williams, and P. Burnap. Pulse: An adaptive intrusion detection for the Internet of Things. In *Living in the Internet of Things: Cybersecurity of the IoT*, 2018.

[11] E. Anthi, L. Williams, M. Sowiska, G. Theodorakopoulos, and P. Burnap. A supervised intrusion detection system for smart home iot devices. *IEEE Internet of Things Journal*, 6(5):9042–9053, 2019.

[12] O. Brun, Y. Yin, and E. Gelenbe. Deep learning with dense random neural network for detecting attacks against IoT-connected home environments. *Procedia computer science*, 134:458–463, 2018.

[13] N. Moustafa, B. Turnbull, and K. R. Choo. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. *IEEE IoT Journal*, 6(3):4815–4830, 2018.

[14] M. Nobakht, V. Sivaraman, and R. Boreli. A host-based intrusion detection and mitigation framework for smart home iot using openflow. In *ARES*, pages 147–156. IEEE, 2016.

[15] A. Procopiou, N. Komninos, and C. Douligeris. Forchaos: Real time application DDoS detection using forecasting and chaos theory in smart home IoT network. *Wireless Communications and Mobile Computing*, 2019.

[16] M. Novák, F. Jakab, and L. Lain. Anomaly detection in

user daily patterns in smart-home environment. *J. Sel. Areas Health Inform*, 3(6):1–11, 2013.

[17] S. Ramapatruni, S. N. Narayanan, S. Mittal, A. Joshi, K. P. Joshi, et al. Anomaly detection models for smart home security. In *IEEE BigDataSecurity*, 2019.

[18] M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato. Anomaly detection in smart home operation from user behaviors and home conditions. *IEEE Trans. Consumer Electronics*, 66(2):183–192, 2020.

[19] A. Acar, H. Fereidooni, T. Abera, A.K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.R. Sadeghi, and A.S. Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! *arXiv preprint arXiv:1808.02741*, 2018.

[20] R. Trimananda, J. Varmarken, A. Markopoulou, and B. Demsky. Packet-level signatures for smart home devices. *Signature*, 10(13):54, 2020.

[21] Y. Wan, K. Xu, G. Xue, and F. Wang. IoTArgos: A multi-layer security monitoring system for internet-of-things in smart homes. In *INFOCOM*, pages 874–883. IEEE, 2020.

[22] E. Anthi, S. Ahmad, O. Rana, G. Theodorakopoulos, and P. Burnap. EclipseIoT: A secure and adaptive hub for the Internet of Things. *Comp. & Security*, 78:477–490, 2018.

[23] M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato. Anomaly detection for smart home based on user behavior. In *ICCE*, pages 1–6. IEEE, 2019.

[24] Google Inc. HTTPS encryption on the web, 2019. URL https://transparencyreport.google.com/https/.

[25] G. Loukas, Y. Yoon, G. Sakellari, T. Vuong, and R. Heartfield. Computation offloading of a vehicle's continuous intrusion detection workload for energy efficiency and performance. *SIMPAT*, 73:83–94, 2017.

[26] F.T. Liu, K.M. Ting, and Z. Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[27] B. Scholkopf, R.C. Williamson, A.J. Smola, J. Shawe-Taylor, and J.C. Platt. Support vector method for novelty detection. In *Proceedings of the 12th International Conference on Neural Information Processing*, pages 582–588. ACM, 2999.

[28] D. Tax and R. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[29] D. Mutz, F. Valeur, G. Vigna, and C. Kruegel. Anomalous system call detection. *ACM Transactions on Information and System Security*, 9(1):61–93, 2006.

[30] S. M. Beyer, B. E. Mullins, S. R. Graham, and J. M. Bindewald. Pattern-of-life modeling in smart homes. *IEEE Internet of Things Journal*, 5(6):5317–5325, 2018.

[31] M. N. Katehakis and A. F. Veinott. The multi-armed bandit problem: decomposition and computation. *Mathematics of Operations Research*, 12(2):262–268, 1987.

[32] J. Vermorel and M. Mehryar. Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*, pages 437–448. Springer, Berlin, Heidelberg, 2005.

[33] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Annual Foundations of Computer Science*, pages 322–331. IEEE, 1995.

[34] R. Seger. rx_tools, 2018. URL https://github.com/rxseger/rx_tools.

[35] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

[36] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2016.

[37] L. Kocsis and C. Szepesvári. Discounted UCB. In *2nd PASCAL Challenges Workshop*, volume 2, 2006.

[38] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[39] L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[40] K. Shah. Multi armed bandit algorithms, 2018. URL https://github.com/kulinshah98/Multi-Armed-Bandit-Algorithms.

[41] D. Quail. Introduction to reinforcement learning via non-stationary bandits, 2017. URL https://github.com/dquail/NonStationaryBandit.

[42] J. Li, Z. Zhao, R. Li, and H. Zhang. AI-based two-stage intrusion detection for software defined iot networks. *IEEE Internet of Things Journal*, 6(2):2093–2102, 2018.

[43] Contiki. The contiki operating system, 2019. URL https://github.com/contiki-os/contiki/.

[44] Checkmarx. Amazon echo: Alexa leveraged as a silent eavesdropper. Technical report, Checkmarx, 10 2018. URL https://info.checkmarx.com/wp-alexa.

[45] N. Pitropakis, E. Panaousis, T. Giannetsos, E. Anastasiadis, and G. Loukas. A taxonomy and survey of attacks against machine learning. *Computer Science Review*, 34, 2019.