International Journal of Information Security (2022) 21:611–636
https://doi.org/10.1007/s10207-021-00571-6

REGULAR CONTRIBUTION

Anomalous behavior detection-based approach for authenticating smart home system users

Noureddine Amraoui[1] · **Belhassen Zouari**[1]

Published online: 20 November 2021

Abstract
This paper presents Duenna, an authentication framework for smart home systems (SHSs). When using controlling apps (e.g., a smartphone app), Duenna makes sure that only legitimate SHS users are allowed to operate their Internet of things (IoT) devices. Duenna is built upon a behavioral anomaly detection (BAD)-based approach. In particular, we hypothesize that SHS users usually operate their home IoT devices in typical and distinctive patterns. Therefore, users that attempt to operate devices differently from such a regular behavior are considered malicious. Technically, Duenna operates in two modes. In an initialization operation, Duenna first collects and processes the historical cyber and physical activities of an SHS user in addition to the historical states of the SHS itself to build a set of incremental anomaly detection (AD) models. Then, in an interactive operation, the trained AD models are, then, used as a baseline from which anomalous commands (i.e., outliers) are detected and rejected, while regular commands (i.e., targets) are considered legitimate and allowed to be executed. Through an empirical evaluation conducted on real-world data, Duenna exhibits high authentication rates ensuring both security and user experience. The findings obtained from such evaluation show that a user behavior-based approach is a promising security scheme that could be integrated into existing SHS platforms.

Keywords Internet of things · Smart home systems · User authentication · Behavioral anomaly detection · Intrusion detection

1 Introduction

The Internet of things (IoT) is becoming increasingly widespread in home environments. Inhabitants are transforming their homes into smart spaces with Internet-connected sensors and actuators such as motion detectors, lights, and door locks. Due to the growing interest in smart home environments, the number of systems designed to support them has risen considerably and has received significant attention [1]. Smart home systems (SHSs) provide several intelligent services to consumers such as energy-saving, physical security and safety, and elderly people assistance. To take advantage of these services, consumers can operate their devices in several ways. In particular, SHSs may provide consumers with companion applications and web portals that can be run on end-user devices such as tablets and smartphones, so consumers can operate their devices on their own either from the inside when connected to the local network, or from any outside location via the Internet.

While providing a significant convenience to consumers, unauthorized and unexpected operation of SHS devices brings new security and safety concerns. Furthermore, these latter have a great impact compared to the malicious operation of a classical computer system. In other words, the maximal damage a computer attacker could inflict is limited to data loss. With the IoT-based SHSs, attackers can have physical effects in the world, such as opening doors, causing fake fire alarms, and disrupting electricity supply [2]. In particular, several vectors could allow an attacker to operate SHS devices. To begin with, the latest results show that IoT devices have been suffering from the open-port problem, weak encryption, and lack of authentication/authorization to their web interfaces [3]. Also, end-user control devices can easily be compromised if they are not secured properly [4]. Finally, the account which an SHS owner uses to access companion applications could be compromised in several ways

✉ Noureddine Amraoui
houcine1amraoui@gmail.com

Belhassen Zouari
belhassen.zouari@supcom.tn

[1] Mediatron Research Laboratory, Higher School of Communications of Tunis, University of Carthage, Technology City of Communications, El Ghazala 2083, Ariana, Tunisia

such as reverse engineering, password guessing, and malware infection [5].

Preventing malicious operation of SHS devices requires a robust security scheme. In this perspective, three types of conventional security mechanisms have gained a particular attention viz., signature-based intrusion detection systems (IDSs) [6,7], user authentication [8,9], and access control models [10–12]. Unfortunately, these approaches suffer from several limitations. Firstly, although existing signature-based IDSs assume that legitimate and anomalous traffic signatures are notably different, operation commands sent via a compromised end-user device will have the same IP addresses and packets' signatures [13]. Secondly, although recent works have been combining authentication factors to identify SHS users (e.g., passwords, smart-cards, fingerprint, etc.) [14], these factors remain vulnerable to social engineering attacks and the overall usability of the SHS decreases since users are forced to carry around specific authentication tokens [9]. Thirdly, traditional access control models such as capability-based access control (CapBAC) have not been considered as an effective security mechanism for emerging technology such as SHS [11] since they cannot prevent the misuse of legitimate privileges by a malicious user [15] and they are not expressive enough to handle complex access control needs of an IoT-based system [10].

Recently, the limitations of conventional security mechanisms have been warranting researchers to integrate the behavioral analysis of both SHS and its users to devise new schemes that are self-learning, personalized for each SHS configuration, and allow more intelligent authentication and authorization decisions. Tracing then assessing the behavioral patterns of users and entities to secure cyber systems is better known as behavioral anomaly detection (BAD) [16]. A BAD-based security approach attempts to identify security threats and behaviors that are not known and do not match the predetermined patterns. Although some works have been leveraging BAD-based security [13,15,17,18], such approach is still poorly adopted among the SHS security frameworks.

In this work, we aim to address the lack of relevant BAD-based techniques to secure SHSs. To reach this goal, we hypothesize that SHS users usually operate their home IoT devices in typical and distinctive patterns that can be described by many behavioral features such as rate (i.e., number of devices operated in a time window), sequencing (i.e., the order of operation). Also, operating SHS devices is generally related to the daily activities of inhabitants. For instance, a user usually turns off the heater in the mornings when nobody is home (i.e., no physical activity). Therefore, correlating the cyber-operation with the physical-activity is a strong behavioral pattern. In Fig. 1, we present a motivating example to show how the integration of the user and SHS behavior can efficiently help in preventing the unauthorized/malicious operation of devices. In this example, we

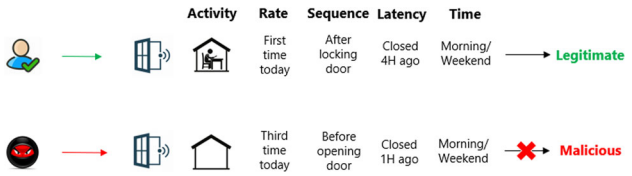

Fig. 1 A scenario of a legitimate/malicious user-driven operated smart window

consider the scenario (which has been reported in real world [19]) of a smart window open command performed by a legitimate and an adversary. The operation command issued by the legitimate user is considered as authorized and executed since he/she frequently opens the window in the morning of weekends when the home is occupied, and after closing the door. An adversary, however, did not follow such a legitimate behavioral pattern and attempts to open the window to break into the house in a different situation, i.e., when the home is unoccupied and before opening the door, etc. Thus, the operation command is considered malicious and rejected.

In contrast to afore discussed conventional techniques, authenticating users based on their behaviors has many advantages [20]. Indeed, user behavioral patterns could be discreetly collected and monitored since they depend on mental characteristics and do not require user interaction. Consequently, neither authorized nor unauthorized users are aware of being forensically monitored, thus they do not interfere with their experience. Additionally, user behavior-based verification could be continuously performed throughout the entire user's operation session.

This paper introduces Duenna, a BAD-based security framework. Duenna continuously authenticates SHS users and makes sure that only legitimate ones are allowed to operate SHS devices. To do so, Duenna operates in two stages. During an initial operation mode, the historical cyber and physical activities of the user in addition to the historical states of SHS itself are first collected. This data is then used to construct a set of vectors and matrices called probabilistic models that summarize the behavioral patterns seen in these historical data in a form of probability values. After that, a set of incremental anomaly detection (AD) models is trained on a set of behavioral scores calculated from the constructed probabilistic models. Later in an interactive operation mode, the trained baseline AD models are then used to continuously verify the legitimacy of the user requested operation commands and take security actions accordingly.

During the interactive operation, Duenna ensures an SA-TRR-ICA: self-adaptive, trust-based, risk-aware, recoverable, implicit, and continuous authentication. In particular, based on a set of behavioral scores that can be discreetly calculated and assessed, Duenna can continuously evaluate users' behaviors during the entire SHS operation. Also, Duenna calculates a confidence score that can be continu-

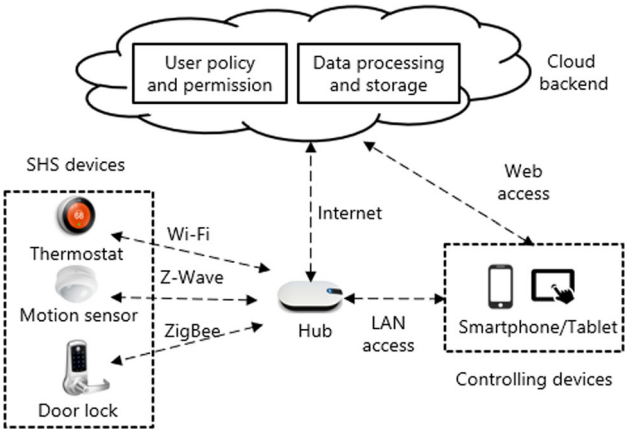

Fig. 2 Architecture of a user-driven operated SHS

ously evaluated to ban users that show anomalous behavior. In addition, Duenna switches to a degraded operation mode where highly sensitive SHS devices are disabled from being operated. This functionality makes Duenna aware of the risk that may come with false executed commands when the legitimacy of the current user is uncertain. Finally, Duenna can automatically adapt to the potential change of legitimate user behavior.

Some of Duenna's features were firstly introduced in [17]. In the previous work, the ensured features were only limited to implication and continuity; moreover, the evaluation experiments were only conducted on artificial datasets. However, this paper brings new proposed concepts and features and makes the following contributions:

- Full design and evaluation of an authentication framework for SHS users that is built upon a BAD-based security approach.
- Performance evaluation experiments have been conducted on real-world SHS datasets. The obtained high authentication rates successfully validate that such an approach is a promising security scheme that could be integrated into existing commercial SHSs.
- To the best of our knowledge, this work is the first to adapt SHS user's physical activities history data that is publicly available to be used as the history data of SHS devices operation. Such an adaptation will provide public benchmark data usable for validating our framework as well as other similar behavior-based techniques.
- Extraction of new behavioral scores to describe SHS user behavior when operating his/her IoT devices by drawing inspiration from other applications such as social networks and web applications.
- Proposing new security and design features to reinforce the conventional authentication schemes used in the-state-of-art SHS platforms.
- Leveraging incremental anomaly detection to propose a new self-adaptation feature to cope with the potential change in the behavior of SHS users.

The remainder of this paper is structured as follows. Section 2 provides the necessary background, presents the considered security threats, and situates the paper within the state-of-the-art. Section 3 gives a detailed description of the core modules underlying the operation of Duenna. Section 4 presents the empirical evaluation of our proposed framework. Finally, Sect. 5 draws the main research conclusions and underlines some future directions.

2 Background, threat model, and positioning to the state of the art

2.1 Internet of things-based smart home systems

Smart home systems (SHSs), also called Connected Homes or Domotics, represent a class of the most prevailing IoT-based systems [21]. The plethora of services and applications that SHSs can provide (e.g., energy-saving, physical security and safety, elderly people assistance, etc.) has been promoting the enthusiasm of consumers toward this type of system. To take advantage of different intelligent services, a consumer can configure or operate an SHS in several ways. Today's SHSs are either user-driven, programmable, or self-learned.

In this paper, we focus on user-driven SHSs. In particular, users may prefer to have full control of their IoT SHS devices, in a way similar to his/her full control of conventional home appliances, without reliance on external parties [22]. Such preference is currently available in today's SHSs such as SmartThings [23]. More specifically, such type of SHS provides consumers with companion applications and web portals that can be run on end-user devices such as tablets and smartphones so consumers can operate their devices on their own either from the inside when connected to the local network, or from any outside location via the Internet.

A typical architecture of a user-driven SHS is depicted in Fig. 2 comprising four basic building blocks [24]. The first block is formed by the IoT devices that could be a sensor (e.g., motion detector) or an actuator (e.g., door-lock). Since there is no generic interoperability standard, SHS devices are connected via a smart hub that works as a coordinator among them and provides a common access point for all the entities in the SHS. End-user devices (smartphones, tablets, etc.) can be either connected to the hub through the local network or to the cloud backend through the Internet for local or remote operation, respectively. Moreover, the hub is also connected to the cloud to offload the big processing

Table 1 Threat vectors of malicious SHS devices operation

Level	Type	Vector
SHS device	Direct remote operation	Open ports accept operation without authentication.
		Web interfaces have a lack of authentication or authorization, a lack or weak encryption, and a lack of input and output filtering.
Network	User impersonation	Interception of login requests and modification of these requests to log in on behalf of the legitimate user.
End-user device	End-user device compromise	An attacker can lure a victim to install a malicious app that runs on his smartphone
		Manipulation by informed adversaries such as friends and coworkers
		Mobile end-user control devices such as smartphones could be dropped and found by adversaries.
Network and end-user device	User account compromise	Reverse engineering, password guessing, malware infection, etc.
		Users, in many cases, utilize the same password across different systems and applications.

and data storage, in addition to managing security policy and end-user permissions.

2.2 Threat model

Even though the market of SHSs has been prospering in the last few years, their security condition is disappointing [25]. Current designs of SHSs suffer from many security problems that may manifest at different levels of an SHS architecture. Table 1 summarizes the common security threats that may lead to malicious operation of SHS devices, whereas Table 2 presents some of these types of attacks that have been recently reported in the real world.

Because the adoption of any new computing technology is usually hindered by the security challenges it brings, the success of SHS is no doubt related to the confidence degree of SHS consumers toward the operation of their devices. This paper aims to partially contribute to the success of such technology by proposing a BAD-based security framework called Duenna. This latter continuously evaluates the legitimacy of devices' operation commands to prevent malicious ones. Specifically, commands analyzed as abnormal are blocked whatever the threat vector exploited to initiate them.

2.3 Existing behavioral anomaly detection-based security approaches

A BAD-based security solution could be relevant for multiple cyber systems and environments such as web applications and databases [31]. In particular, it is meant to deal with multiple security threats such as malicious insider and compromised user account. As depicted in Fig. 3, a BAD-based security solution operates in two main stages, namely, profiling and analysis. In the first one, the baseline models are first constructed over the regular conduct of the profiled targets (i.e., users or system entities). This stage includes two main processes: (i) collecting historical data on the profiled target, and (ii) building baseline profiles by training anomaly detection (AD) models, whereas, in the analysis stage, the behavior of users/entities is continuously monitored and compared to the established baseline AD models to detect abnormal behaviors, recognize security threats, and take security actions according to the output of the AD model.

In this section, we review some of the state-of-the-art BAD-based security approaches. We first review approaches to secure non-IoT-based systems, then, we review those specifically proposed to secure IoT-based SHSs.

2.3.1 Existing approaches for securing non-IoT-based systems

To have a broader understanding of the usefulness of a BAD-based approach, we review existing proposed works to secure

Table 2 Recent real-world reported malicious SHS devices operation

Device	Example
Smart door	The attacker opens a smart door to break into the house [26].
Smart heater	An attacker operates the heater to overheat the bedroom [27].
Smart stove	An attacker operates the stove to cause fire [28].
Smart window	An attacker opens the smart window to break into the house [19].
Philips Hue smart light	An attacker uses a drone to turn off smart light to cause a blackout [29].
Smart toilet	An attacker remotely operates a smart toilet to cause water overflow [30].

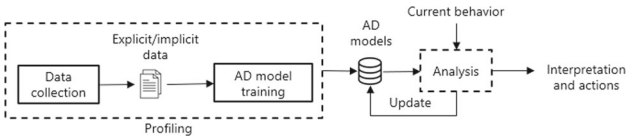

Fig. 3 Basic operational scheme of a BAD-based security approach

non-IoT-based systems. This study allows us to understand different security threats thwarted, the types of behavioral analysis features and the data sources used for their extraction, and the AD techniques leveraged. In the following, we will discuss some of the works proposed in the context of web-based applications, databases, and online social networks (OSNs), as summarized in Table 3.

Web applications When a web user requests a resource (e.g., HTML file), the webserver records these requests in web logs. When making these requests, user behavior can be described by many features that could be extracted from the weblogs. By building a baseline over regular requesting patterns, any anomalous request could be then detected. In this context, several methods have been proposed to detect many security threats such as Application layer Distributed Denial of Service (DDoS) attacks. Liao et al. extracted web user behavioral features based on the idea that users access and spend more time on pages of their interests [32]. Examples of the extracted features include the number of requests in a time window and the duration between them. The AD technique used is a hybrid classification algorithm based on Sparse Vector Decomposition and Rhythm Matching (SVD-RM) algorithm. Najafabadi et al. applied another anomaly detection technique based on the Principal Component Analysis (PCA) to detect DDoS attacks [33]. The idea is to identify the N-top principal components that better describe the regular user behavior. Then, the data projection on the remaining components captures anomalies and noise in the data.

Databases Database servers typically maintain an audit log in which they record SQL requests on DB resources (i.e., Tables). By learning the behavior of clients when requesting resources, any changes detected in this behavior could be a sign of abnormal access. In this context, Mathew et al. proposed a clustering-based AD technique to detect users' abuse of privileges [34]. To profile the user behavior, the proposed approach consists of first summarizing the query's result tuples in a fixed vector. Then, all user's past queries can be thought of as one cluster. When a new query arrives, if it belongs to the user's cluster, it will be classified as normal, or abnormal otherwise. More recently, Mazzawi et al. presented a novel approach to determine whether a DB user activity is malicious or not [35]. The idea consists of checking the user's self-consistency with its previous actions and global consistency with the past actions of similar users. Three main behavioral features were used to describe users' activities: (1) rarity: represented by the probability of appearance of the action in a new timeframe, (2) volume: characterized by the number of occurrences of an action given that it appeared in the timeframe, and (3) new object: amounts of new objects being accessed.

Online social networks (OSNs) OSNs provide a variety of online features for their users to engage in, such as sending messages, commenting, updating a status. OSNs users can thus be distinguished according to their interaction with these features to detect anomalous behaviors. In this context, Viswanath et al. proposed three types of features to characterize the social behaviors of OSN users viz., (1) temporal: number of Likes per day, (2) spatial: number of Likes in different categories (e.g., politics, sports), and spatiotemporal: summary of the distribution of Like categories using entropy [36]. To detect fake and colluding Facebook identities, a PCA-based AD technique was used. Ruan et al. proposed to characterize a user's social behavior by his/her extroversive activities (e.g., action latency) and introversive activities (e.g., browsing sequence) [37]. To differentiate between different user profiles, two measures were used. Euclidean distance to quantify the difference between two feature vectors of two users' profiles and self-variance to differentiate between each pair in a collection of profiles.

Table 3 Summary of some of the existing BAD-based approaches for securing non-IoT-based systems

System	Ref.	Security threat	Behavioral features	AD technique	Data source
Web applications	[32]	Application layer DDoS	Requests number per time window, duration between requests	SVDD, RM	Web logs
	[33]		Request rate	PCA	
Databases	[34]	Abuse of privileges	Summarizing queries result tuples in clusters for each user	Clustering	Audit logs
	[35]	Credentials hijacking and abuse of privileges	Rarity, volume, new object accessed	Anomaly scores assignment	
*OSN	[36]	Fake and compromised Facebook identities	Number of likes per day, number of likes in different categories	PCA	UGC
	[37]	User account compromise	First activity, action latency, visit duration, browsing sequence	Euclidean distance, self-variance	

2.3.2 Existing approaches for securing user-driven operation of IoT-based SHSs

The limitations of conventional security mechanisms such as authentication, access control have been warranting researchers to integrate the behavioral profiling of both SHS and its users to make more security intelligent decisions. Continuous Authentication (CA) is one of the main emergent techniques from the BAD-based security approach. Also known as permanent authentication, CA is supposed to increase the level of security by keeping SHS users authenticated permanently and enhance the users' quality of experience by being non-intrusive and minimizing the usage of credentials during the authentication processes [38].

User cyber behavior-based authentication is a category of CA. It provides a safer and more convenient way to identify users based on their behavioral interaction with cyber systems. Some recent works have been leveraging such an approach to secure the operation of SHSs. Rath and Colin [18] proposed an access control framework to authenticate the operation of SHS devices in case of user account compromise using association rules as a means to learn user behavior. However, the framework does not use any behavioral features that may efficiently describe SHS user behavior. Ghosh et al. [15] proposed SoftAuthZ, a framework for estimating the confidence associated with a device access request. SoftAuthZ computes the belief of a requester based on his/her historical request patterns using a linear regression model. In particular, an access request with low variability is more likely to be legitimate in contrast to an abnormal request that should have high variability. However, SoftAuthZ uses variability in device access requests as the only user behavioral feature besides other non-behavioral attributes such as environmental context and nature of the requested device. Moreover, operation commands are not transformed into feature-based numerical data and only treated with their original categorical nature. This obliged authors to use a variability calculation method specifically for categorical variables.

More recently, Yamauchi et al. [13] proposed a method to detect the exceptional operation of SHS devices. The method first learns sequences of events performed by the user to construct a tree as a baseline. Then, anomalous event sequences are detected by checking whether the sequence is included in the constructed tree. However, the proposed method uses the operation sequence as the only user behavioral feature, thus it cannot accurately identify single commands for which related commands are not observed. Moreover, the proposal only considers the SHS devices separately and does have a global view of the SHS.

In light of the previous study on existing BAD-based security approaches, Duenna works analogously to prior works and draws inspiration from the behavioral characteristics

used by them. Specifically, characteristics such as browsing rate and sequence in Web applications, action latency, and inter-Like delay in OSN, could also be employed in the context of SHS to continuously authenticate users when operating IoT devices. Besides, although there have been some BAD-based works to secure SHS user-driven operation, such an approach is still poorly adopted among SHS security frameworks.

3 Duenna: a user authentication framework for smart home systems

Built on the assumption that there is distinguishable behavior between users of the same or different SHS's, Duenna leverages BAD-based security approach to build probabilistic and anomaly detection (AD) models that summarize the behavioral patterns of the legitimate SHS user. These models are then used as a baseline from which a requested operation command is accepted or rejected based on its deviation degree from this baseline.

In this section, we first provide an overview of Duenna by explaining its operational architecture. Then, we concretely show its operation to secure user-driven SHSs.

3.1 Overview

3.1.1 Operational architecture

Figure 4 depicts the operational architecture of our proposed framework. In particular, Duenna operates on an enhanced architecture beyond the basic BAD-based approach (cf. Fig. 3). More specifically, before Duenna starts securing SHS devices from the malicious operation, an initiation stage is first performed on the historical data of both user and SHS itself including two processes viz., Raw Logs Collection and Enrollment (cf. Sect. 3.2). The result of the initial stage is the AD models summarizing the behavioral patterns seen in the collected logs. Once AD models are built, Duenna starts analyzing requested operation commands in an interactive mode since the user is engaged in requesting operation commands and receiving prompts and analysis responses. This operation includes two modules viz., anomaly analyzer and action manager (cf. Sect. 3.3).

3.1.2 Instantiation of Duenna on a user-driven SHS architecture

To show the concrete operation of Duenna, Fig. 5 depicts its instantiation on a user-driven SHS architecture (cf. Fig. 3):

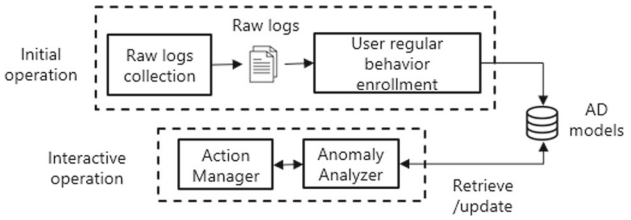

Fig. 4 Operational architecture of Duenna

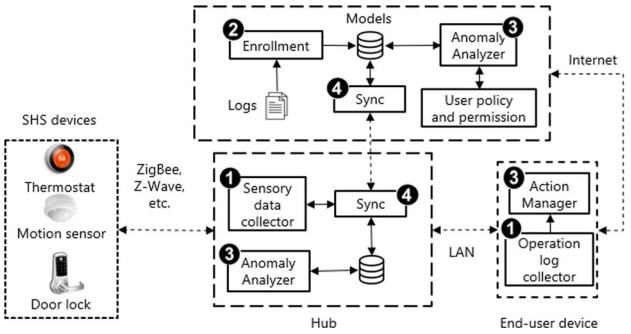

Fig. 5 Instantiation of Duenna on a user-driven SHS architecture

1. During the initial operation, the historical cyber and physical activities of the user in addition to the historical states of SHS itself are first collected. The collection of the two types of data is performed by collection modules that are deployed both on the Hub and end-user controlling device.
2. This data is then used to enroll the regular behavioral patterns of the SHS user seen in these historical data. The enrollment process includes the construction of probabilistic models and the training of incremental AD models on the behavioral scores calculated from the probabilistic models.
3. Later in an interactive operation mode, the trained baseline AD models are then used to continuously verify the legitimacy of the user's requested operation commands and take security actions accordingly. This operation is managed by an anomaly analyzer deployed both in the hub and the cloud back as well as action manager deployed on the end-user operating device.
4. Once the data collection is completed, a module called Sync is responsible for sending the collected sensory and operation logs to the cloud backend both during the stage of Enrollment as well as during the interactive operation. This module is also responsible for migrating AD models to the hub to be used by the local anomaly analyzer when the enrollment stage is accomplished. Moreover, to keep AD models synchronized on both sides after being updated, the mini-batch used in the update on one side (e.g., hub) is pushed to the other one, and vice versa. This synchronization makes sure that recent user behavior is always known. For instance, if a user has been remotely

Table 4 Structure of user/SHS raw logs

Log	Structure	Collector
User operation log	Command ID, operated device, operation action, timestamp	Operation log collector
User activity log	Activity, start time, end time	Sensory logs collector
SHS states log	timestamp, state of devicei, ..., state of devicen	

operating his/her devices (i.e., cloud-stored AD models have been updated), hub-stored AD models must also be up to date so the user's recent behavior is known if he/she then performs a local operation, later.

3.2 Initial operation

Once Duenna is instantiated and different modules have been deployed, the initial operation is the first process Duenna has to perform to be ready for its interactive operation. The processes executed in this stage include raw logs collection and user regular behavior enrollment as detailed in the following.

3.2.1 Raw logs collection

The first step toward the building of regular user behavioral patterns is the collection of historical data of both user and SHS itself. As previously presented, this task is accomplished through two modules.

Sensory data collector Since the hub is the coordinator of SHS, this module is deployed on the hub and is responsible for collecting sensory events of IoT devices to trace both devices' states (i.e., only concerns devices of type actuator, e.g., door-lock) and user physical activities (using devices of type sensor, e.g., motion sensor).

Operation log collector This module is deployed on the user controlling app, and is responsible for tracking user operation commands then sending them back to the Sync module on the hub or directly to the cloud backend if commands are requested remotely.

As described in Table 4, the collected data consists of different types of logs that trace historical information about both user and devices:

User operation log A log file is a common way to trace the history of manipulation/operation of computing resources by their authenticated users by saving information such as who has operated what, and at what time. Analyzing this data provides a rich understanding of what users have been doing, and how they have been using resources. In this work, the Operation Log Collector is responsible for tracking local and remotely executed commands. Both types of commands are saved in one User Operation Log file stored in the cloud backend (cf. Fig. 5), since they concern the same user actions.

User activity log Recent developments in sensing technology have led to the development of wireless sensor networks [39]. These non-intrusive sensors can be used to recognize and trace different human physical activities (e.g., pressure mats to measure sitting on a couch or lying on the bed, mercury contacts for movement of objects such as drawers, etc.). Collected by Sensory Logs Collector, a User Activity Log saves daily user activities with their start and end timestamps.

SHS states log An SHS state in a given time is the status of the hub-connected devices Collected by the Sensory Logs Collector, this log traces the different status of all devices while being operated at the given timestamp.

3.2.2 User regular behavior enrollment

Once sufficient raw logs collection is gathered, the baseline models summarizing user patterns seen in the collected raw logs have to be enrolled. The output of the Enrollment process is the trained of AD models that are saved to be retrieved in the interactive operation. The Enrollment process is executed on the cloud as it is a resource-consuming task.

Figure 6 depicts the workflow of the enrollment process including four sub-processes: raw logs preparation, probabilistic models construction, behavioral scores extraction, and AD models training. The subsequent sections explain each sub-process in detail.

Logs preparation Before being used in the construction of probabilistic models and the training of AD models, the collected user/SHS raw logs need first to be prepared. This task allows the extraction of more information about the user patterns from the collected raw logs. The resulting prepared logs are given in Table 5 and are generated by the four following sub-processes:

1. Sessions log extraction: in the context of SHS, an operation session is the set of operated IoT devices within a time window. In this work, we follow an inactivity-based identification strategy as it is the most commonly used [40]. Specifically, this strategy implies that if there is a

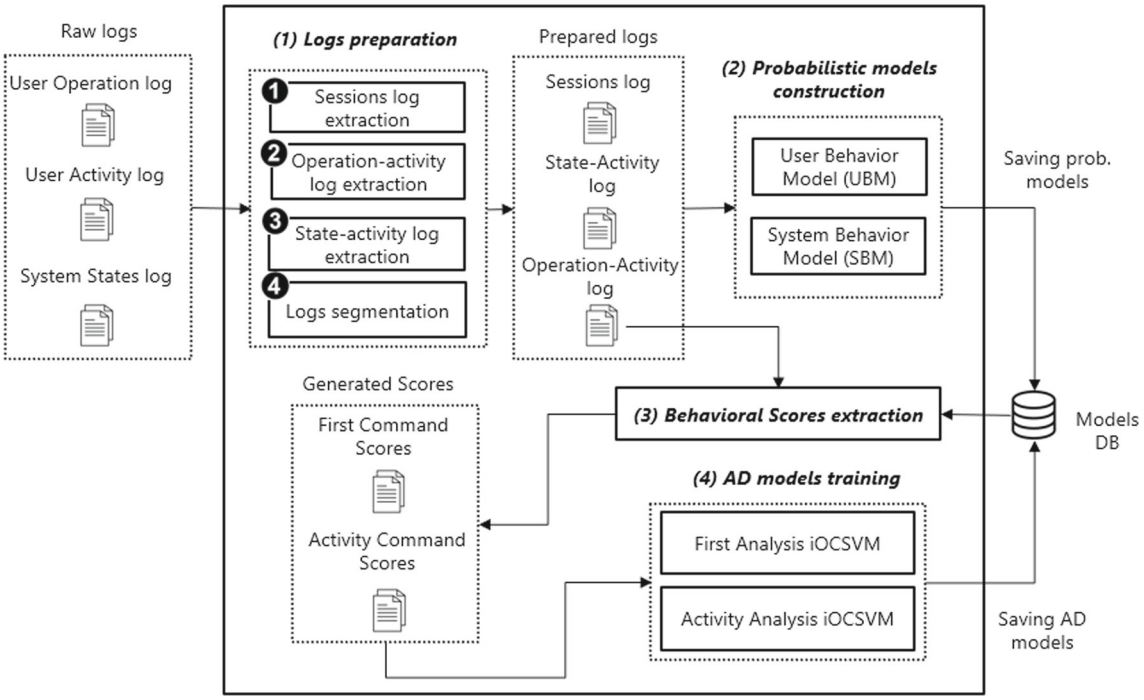

Fig. 6 Workflow of user regular behavior enrollment

Table 5 Structure of user/SHS prepared logs

Log	Structure
Sessions log	Starting timestamp, period, operation rate, operation sequence
Operation-activity log	Command ID, operated device, operation action, user activity, period, timestamp
State-activity log	Timestamp, period, user activity, state of devicei, ..., state of devicen

break between the user's operation commands which is sufficiently long, it's likely that the user is no longer active and a new session is created when the next operation command is initiated. We use the following information to describe an SHS user session. Starting Timestamp: timestamp of the first command of the session, operation rate: number of operated devices during the session, and operation sequence: order of operation during the session.

2. Operation-activity log extraction: since the operation of a particular device is generally related to a particular physical activity (or activities), the mapping of an operation command captured in the user operation log with the corresponding user physical activity (or activities) captured in the user activity log gives a strong insight about the user operation-activity patterns. For instance, the inhabitant always turn-on TV while relaxing, turning-on surveillance camera while leaving the house, etc. If no activity is performed at the time of operation command, the 'no activity' label is added.
3. State-activity log extraction: the mapping of an SHS state captured in SHS States Log with the corresponding user physical activity (or activities) captured in the activity log also provides an insight about SHS-user activity patterns as the status of device (or a set of devices) is the consequence of particular user activity (or activities). For instance, all doors are locked while the user is not at home.
4. Logs segmentation: segmenting the logs consists of adding the corresponding time interval of the day (i.e., period) to each row record in the logs. Adding this information gives another precision to understand the user's frequent patterns. Indeed, the behavior of an SHS inhabitant through the 24 hours of the day is generally segmented into a set of frequent periods wherein the user has some specific behavioral routines (e.g., waking up and going to work in the morning, sleeping at night, etc.). In recent years, two main time-series segmentation techniques have been used for various purposes (e.g., mobile user data) viz., equal- and unequal-interval strategies [41]. In this work, we follow an unequal interval-based

segmentation strategy in which the 24 hours of a day are divided into 2 or more periods with unequal lengths. In the empirical evaluation (cf. Sect. 6.4), the optimal segmentation that most fits user behavior patterns seen in the logs will be selected among different candidates.

Probabilistic models construction A probabilistic model consists of a set of vectors and matrices containing different probabilities that describe both user and SHS behavioral patterns seen in the prepared logs. These models are distinguished into one related to user behavior viz., User Behavior Model (User-BM) and another related to SHS behavior viz., SHS Behavior Model (SHS-BM) as depicted in Tables 6 and 7, respectively.

Behavioral scores extraction We call the behavioral scores, the data on which the AD models are trained. Extracting the SHS user behavioral scores consists of calculating a tuple of numeric values for each command seen in the operation-activity log using the constructed probabilistic models (i.e., User-BM and SHS-BM) (cf. Fig. 6), in addition to other scores that can be calculated independently. To this end, user operation commands are transformed from categorical data (i.e., a tuple of multiple attributes viz., operated device, action, timestamp) to feature-based numerical data that is suitable for an AD model training.

In particular, we distinguish two types of commands each of which is represented with a specific and common set of scores. If the time since the last operation command (called it inactivity time) is bigger than a specific threshold (called it inactivity threshold), the requested command is considered as a Starting Command since it represents the beginning of a new operation session. However, if the inactivity time is still below the inactivity threshold, the requested command is considered as an Activity Command since it belongs to the same operation session.

Thus, starting command scores are used to describe user behavior when starting an operating session:

- Session starting: probability by which the user starts a session by requesting to operate the given device with the given operation action.
- Inter-session latency: delay between the session starting's requested command and the last command of the previous session that belongs to the same period of the day.
- Operation-activity: probability by which the user requests to operate the given device for the given period of the day while doing the given physical activity.
- State transition: probability by which the SHS transits to the state resultant from the execution of the requested command.
- State-activity: probability by which the SHS transits to the state resultant from the execution of the requested command while the user is doing the given physical activity.
- Device frequent state: probability by which the operated device would be in the state resulting from the requested operation action for the given period of the day.

Activity command scores allow to describe a user behavior when he/she initiate an operation command preceded by other commands belonging to the same operating session:

- Intra-session transition: probability by which the previously operated device would be followed by the device requested in the given command in the same operation session.
- Current sequence: probability to see the current operation sequence.
- Intra-session latency: delay between the requested operation command and its antecedent in the same session.
- Operation rate: number of current operated devices.
- Operation-activity: probability by which the user requests to operate the given device for the given period of the day while doing the given physical activity.
- State transition: probability by which the SHS transits to the state resulting from the execution of the requested command.
- State-activity: probability by which the SHS transits to the state resultant from the execution of the requested command while the user is doing the given physical activity.
- Device frequent state: probability by which the operated device would be in the state resulting from the requested operation action for the given period of the day.

Anomaly detection models training Training AD models on the set of extracted regular behavioral scores is the fruit of all the previous enrollment sub-processes. Again, Duenna is a BAD-based security framework that discriminates legitimate user's SHS operation from anomalous one by detecting abnormal target behaviors. However, in this AD scenario, anomalous SHS users' behaviors are often unknown and not available at the beginning of the behavior enrollment stage, and only legitimate SHS users' behaviors are available. This setup is recognized as semi-supervised or often as an unsupervised AD [42]. Hence, the AD models are only trained on the regular data of legitimate users captured when operating their SHSs. After that, behaviors deviating from such regular baseline are considered as non-legitimate.

Furthermore, user operation commands need to be analyzed as they are issued in real time and the initialized AD models during the enrollment stage need to be updated. Yet,

Table 6 User Behavioral Model (User-BM)

Parameter	Type	Used log	Description
Session starting	Vector	Sessions log	Probability values by which a user starts to operate a device at a given period of the day for all existing devices. For example, during the evening, a user frequently starts a session by setting up the temperature on the thermostat with a 0.39 probability.
Intra-session transition	Matrix		Probability values by which a user transits between every two devices. For example, during the morning, a user frequently opens the garage door-lock after unlocking the front door-lock with a 0.6 probability.
Operation-activity	Vector	Operation-activity log	Probability values that a user requests to operate a given device while performing a given physical activity. For example, the user frequently set up the thermostat while outside.

Table 7 SHS Behavioral Model (SHS-BM)

Parameter	Type	Used log	Description
States transition	Matrix	State-activity log	Probability values by which SHS transits between every two states for a given period of the day. For example, when the camera is off, the door-lock is never opened.
State-activity	Vector		Probability values for SHS to be in a given state resultant from executing the requested operation command, while the user is doing a given activity. For example, when the inhabitant is sleeping, doors are locked, lights are off, etc.
Devices frequent state	Vector		Probability values for each device to be in a specific state at a specific period of the day. For example, in the morning, the door lock remains locked 80% of the time.

re-training an AD model on all new and past data is an intensive task in terms of time and computational resources especially when the update is done on resource-constrained devices (e.g., hub). This setup is recognized as an online (incremental) AD [43]. Moreover, since two types of operation commands are distinguished viz., starting and activity command, two types of AD models are trained each of which on the corresponding extracted starting and activity behavioral scores called Starting AD model (S-AD), and the Activity AD model (A-AD). The training of both types of AD models is performed on the cloud-backend. Then, the trained models are migrated to the hub via the Sync module. Once AD models training and migration are done, the interactive operation is ready.

3.3 Interactive operation

To handle user-initiated operation commands and only execute authorized ones, we instantiate the two modules responsible for managing the interactive operation. As previously shown in Fig. 5, anomaly analyzer (AA) is deployed both in the cloud backend and on the hub and is responsible for retrieving the trained AD models and applies them to initiated commands to analyze their anomaly. It is also responsible for user login verification by interacting with the existing user policy and permission module. On the other hand, action manager (AM) is deployed in the end-user device and is responsible for handling user-requested commands, triggering AA, and taking security actions according to the anomaly analysis result. Specifically, if an operation command is locally initiated, AM triggers A deployed on the Hub which uses locally stored AD models. However, if an operation command is remotely initiated, the anomaly analysis is performed on the cloud backend using cloud-stored AD models.

As explained in the following, AM and AA may either operate in a basic mode or in an enhanced mode to secure SHS devices from unauthorized operation. The two modes are dependent on the assumption put on the maliciousness of the SHS adversarial user as well as on the stationarity of the user behavior.

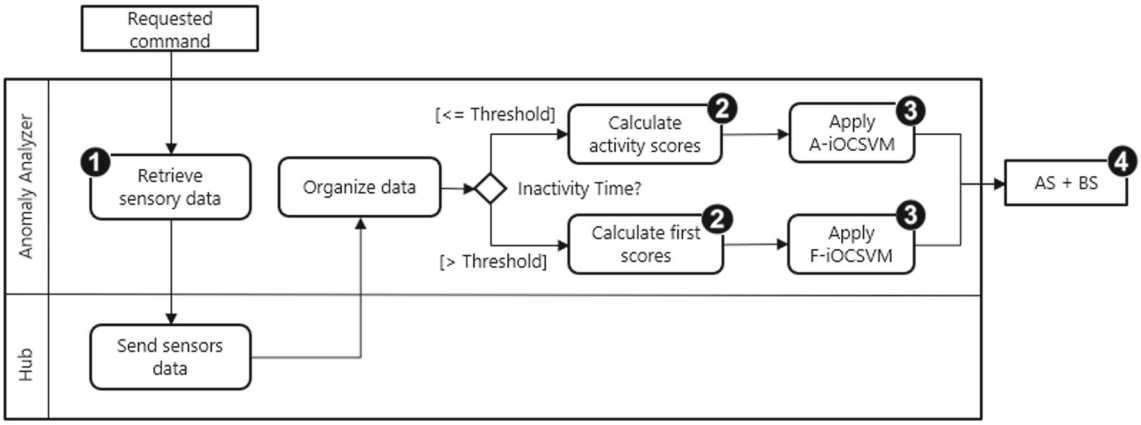

Fig. 7 Workflow of ICA-anomaly analyzer (ICA-AA)

3.3.1 ICA: Implicit and continuous authentication

Assuming that an adversarial user may only compromise the controlling device but not the security credentials, Duenna ensures an Implicit and Continuous Authentication called ICA to prevent the assumed adversary from taking control over the controlling device. In the following, we present the ICA operation of the anomaly analyzer (ICA-AA) and action manager (ICA-AM).

ICA-Anomaly Analyzer (ICA-AA) Fig. 7 depicts the workflow of the ICA Anomaly Analyzer (ICA-AA). Upon receiving an initiated operation command from the ICA action manager (ICA-AM), the anomaly analysis sub-process is triggered.

1. The anomaly analyzer first pulls the hub to get current sensory data and organize it to extract current devices' states and user physical activity.
2. According to its nature (i.e., starting or activity command) which depends on the current user's inactivity time as explained earlier, ICA-AA calculates the corresponding behavioral scores (BSs) of the command using the extracted operation commands parameters (e.g., operated devices, timestamp, etc.).
3. Then, the corresponding AD model is used for analyzing the calculated BSs (i.e., S-AD or A-AD) producing an anomaly score (AS) in the range of [- 1,1].
4. ICA-AA sends back (BSs and AS) to ICA-AM.

ICA-Action Manager (ICA-AM) Fig. 8 describes the workflow of ICA action manager (ICA-AM) and it includes the following sub-processes:

– User login: as depicted in Fig. 9, by providing the correct credentials (e.g., e-mail and a password), the user should be authenticated to the controlling app and authorized to operate the SHS devices. In particular, to verify user login, ICA-AM sends user credentials to ICA-AA deployed in the cloud backend which in turn sends them the existing User policy and permission module to be verified. However, if the controlling app has been closed and opened once again, no login is required.
– Re-authentication: backing to Fig. 8, the wait function means that the continuous analysis is triggered by user-requested commands. Once a user requests to operate a device, the action manager handles the requested command and sends it to the anomaly analyzer which sends back the command's behavioral scores (BS) and anomaly score (AS) to be verified by the action manager. If this score is below a predefined threshold (we call it anomaly threshold), the requested command is considered malicious and an explicit re-authentication (i.e., password verification) is prompted to the user. This is a common technique used in implicit authentication schemes when there is uncertainty about the current user identity [39], whereas, if the credentials provided by the user are not valid, he/she is directly logged out from the main interface of the controlling app.
– Command execution and models update: If the anomaly score is above the anomaly threshold (AT), or if the user succeeds in the re-authentication, the requested command is sent to the hub to be executed as shown in Fig. 10. Upon executing the commands analyzed as legitimate, both probabilistic and AD models should be updated incrementally without being re-trained from scratch. On one hand, vectors and matrices of user behavior and SHS behavior models (i.e., User-BM and SHS-BM) are both updated from the parameters of the executed command (i.e., time, device, action, the period of the day). Updating probabilistic models guarantees that behavioral scores of future requested commands are calculated with recent user behavior. On the other hand, AD models are incrementally updated directly with the already calculated behavioral scores. The update is done after each

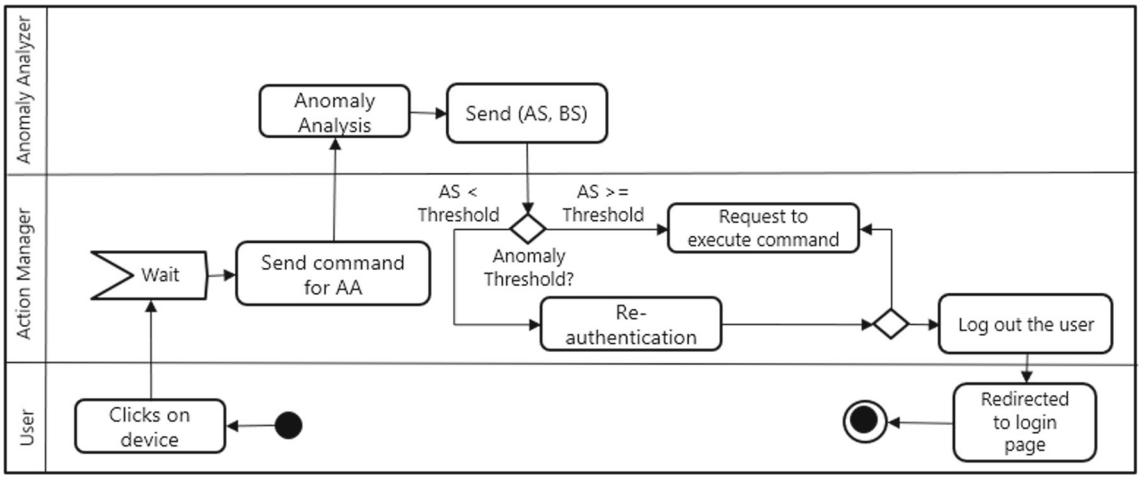

Fig. 8 Workflow of ICA-action manager (ICA-AM)

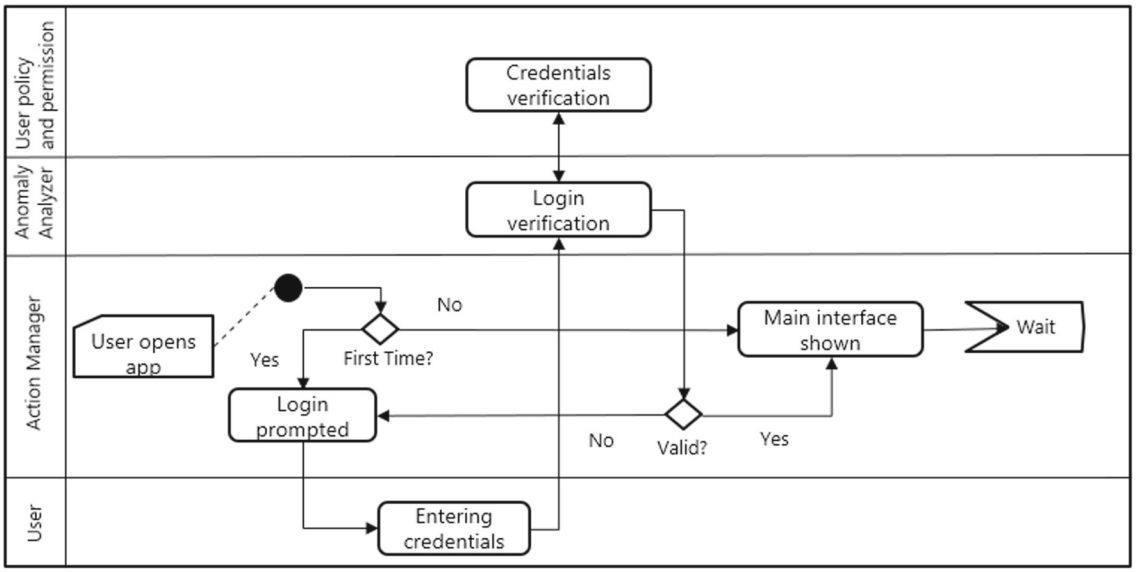

Fig. 9 Workflow of user login

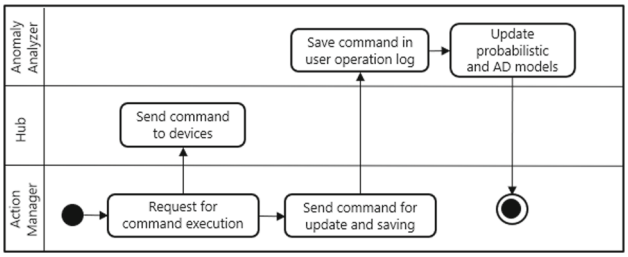

Fig. 10 Workflow command execution and models update

user session (i.e., a set of consecutive requested commands) considering that the same user never changes his/her operating way in one single session, i.e., cannot be in two places at the same time.

3.3.2 TRR-ICA: trust-based, risk-aware, recoverable, implicit, and continuous authentication

Assuming that an adversarial user may either compromise the controlling device or the security credentials, Duenna ensures a Trust-based, Risk-aware, Recoverable, Implicit, and Continuous Authentication called TRR-ICA to prevent such adversary from operating SHS devices. In the following, we present the TRR-ICA operation of the action manager and anomaly analyzer.

TRR-ICA-Action Manager (TRR-ICA-AM) Fig. 11 shows the process of the TRR-ICA action manager (TRR-ICA-AM). Compared to ICA action manager, in this operation mode, more security features are employed viz., Trust-based verifi-

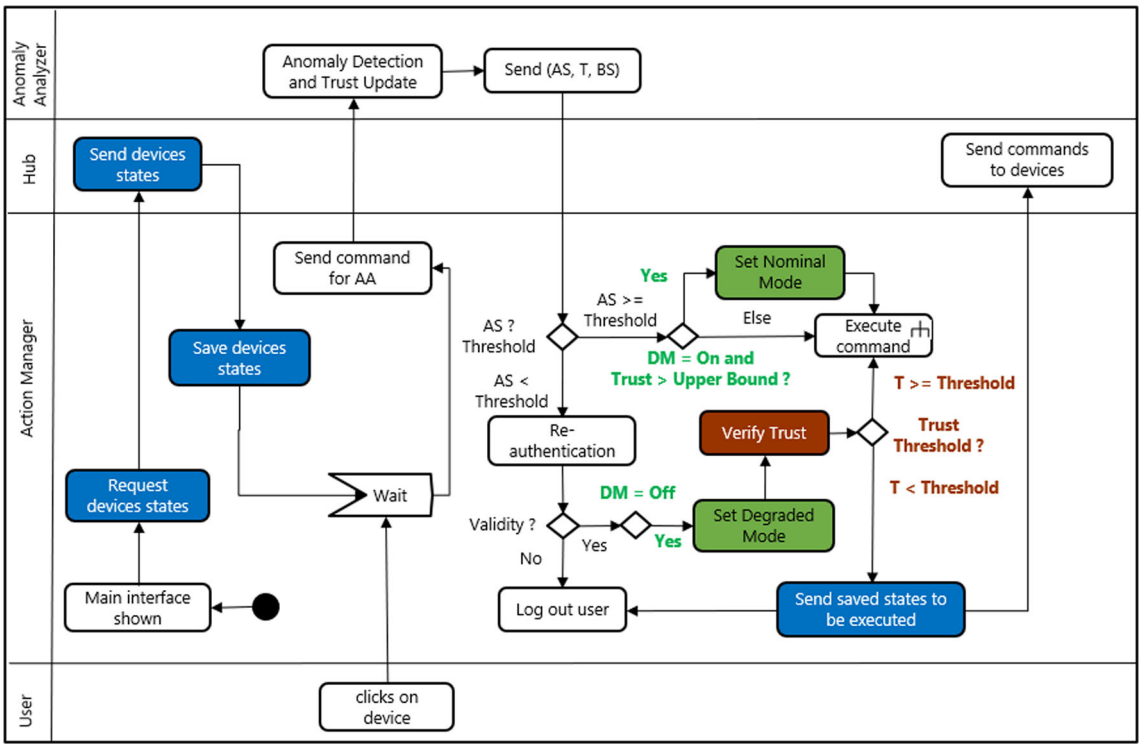

Fig. 11 Workflow of TRR-ICA-action manager (TRR-ICA-AM)

cation, Risk-awareness, and Recoverability, as presented in the following.

– Trust-based verification (in red): Previously, the ICA-action manager was executing a request command directly if the credentials provided by the user are valid. To prevent a user, that can successfully pass the re-authentication (i.e., a potential attacker compromising credentials) while continuously showing an anomalous behavior, from keep using the controlling app, a trust-based verification is proposed. The motivation is based on the idea that the trust value of an SHS user can be reflected by his/her behavior [44]. Basically, if the user succeeds in the re-authentication, the confidence toward this user is first verified before executing the requested command. If the trust value is still above the allowed level of trust (we call it lockout threshold), the command is executed. However, if it drops below the threshold, the requested command is rejected, and the user is logged out. It should be noted that operation commands executed after re-authentication are not used to update normal models since they belong to temporary trust-based verification.
Technically, to follow the confidence level related to user behavior, we leverage the trust-based verification first introduced in [45]. In particular, a trust value is calculated from the anomaly score (AS) outputted from analyzing the anomaly of the requested command. Hence, the confidence toward the current user may increase or decrease according to the anomaly result of the command. The formula (cf. Equation 1) that we adopt to calculate the change in user trust is the one described in [46], where the parameter AT represents the predefined anomaly threshold. If the anomaly score of a given command (denoted as ASi) is equal to this threshold then ΔTrust = 0. If ASi > AT then ΔTrust > 0 meaning that a reward is given, and if ASi < AT then ΔTrust < 0 meaning that the trust decreases because of a penalty. Furthermore, parameter B is the value of AS in which the maximum value of penalty/reward is given, whereas the parameters C and D are the upper bound value of the reward and the penalty, respectively.

$$\Delta_{Trust}\left(AS_i\right) = min\left(\frac{D\left(1+\frac{1}{C}\right)}{\frac{1}{C}+exp\left(-\frac{AS_i-AT}{B}\right)} - D, C\right) \tag{1}$$

The new value of the trust (i.e., Trusti) is calculated in terms of the current change of trust (i.e., ΔTrusti) and the last value of the trust (i.e., Trusti − 1) as described in the following equation (cf. Equation 2). In this equation, the minimum value is taken after calculating the change in trust. This choice allows preventing an adversary from taking advantage of the high system trust obtained by

the legitimate user before compromising the controlling device/app. Moreover, whenever a user is locked out, the trust value is reset to the reward upper bound C to simulate a new session starting after the set of actions that lead to this lockout.

$$Trust_i = min(max(Trust_{i-1} + \Delta_{Trust}(AS_i), 0), 100) \quad (2)$$

- Risk-awareness (in green): If the user can re-authenticate and shows an acceptable level of trust, the requested commands are executed, but his/her behavior is still considered suspicious (e.g., potential adversary compromising credentials). To reduce the impact of suspicious executed commands during the trust-based verification, we propose a degradation technique to a restricted operation mode. We inspired this idea from the fault-tolerance systems that usually downgrade to a specific mode (called Degraded Mode) in which they continue to operate but with minimal functionalities when abnormal operating conditions are detected. Once normal conditions are satisfied, the system switches back to its normal operation (called Nominal Mode) [47]. Since an SHS contains multiple IoT devices with different sensitivity degree, devices such as surveillance camera and front door-lock can be considered highly sensitive since they have a dangerous impact when are maliciously manipulated (e.g., a burglar unlocks the front door-lock to enter the house, raises the heating degree on the thermostat to cause a fire, etc.). On the other hand, other devices such as lights and TV can be considered as lowly sensitive since their malicious manipulation have no dangerous impact (e.g., switching on/off lights is disturbing but not dangerous). To this end, when switching to the Degraded Mode, the action manager disables highly sensitive devices and only allows the current user to operate lowly sensitive devices. This scheme makes the action manager aware of the risk that may come with false executed commands, as the legitimacy of the current user is not certain. The switch-back to the Nominal Mode of operation is done when the user's trust returns to the upper bound of reward (i.e., parameter C in Equation 1).
- SHS Recovery (in blue): When a user is logged out after failing in the trust-based verification, his/her previously requested commands have been executed since action manager was uncertain about their maliciousness. To prevent dangerous consequences of false executed commands, such as turning off the camera, raising the temperature, left the door unlocked, a recoverability technique is proposed wherein SHS devices are reset to their initial states saved before. In particular, devices' states (i.e., on/off, locked/unlocked, etc.) are saved in three points: 1) after successful login, 2) when the current session expires, and 3) after executing a requested command analyzed as legitimate.

TRR-ICA-Anomaly Analyzer (TRR-ICA-AA) To ensure the risk-awareness feature, TRR-ICA-anomaly analyzer (TRR-ICA-AA) needs to be adapted as shown in Fig. 12. For doing so, in addition to the conventional Starting AD (S-AD), we recognize two types of Activity AD models as depicted in Table 8. The Nominal Mode AD (NMA-AD) model is trained on all eight Activity Command analysis scores. However, for the Degraded Mode AD model (DMA-AD), three scores viz., device transition probability, current sequence probability, and state transition probability are excluded, and only five are used for the training.

3.3.3 SA-TRR-ICA: self-adaptive, trust-based, risk-aware, recoverable, implicit, and continuous authentication

Assuming that the legitimate user may change his/her regular behavior, Duenna ensures a self-adaptive trust-based, risk-aware, recoverable, implicit, and continuous authentication. In our previous work, it has been assumed that the behavior of an SHS user is static over the lifetime of the system and does not change after the enrollment stage [17]. In practice, however, a user may change his/her behavior for several reasons. For instance, during the quarantine caused by Covid-19, people were obliged to stay in their homes. As a consequence, their habits and daily routines have been changed; thus, the way they operate their SHS devices has been also changed. Such drift in the behavior of SHS users makes action manager unable to recognize legitimate operation commands since they are deviating from the learned normal behavior. Consequently, it is important for AA to quickly adapt to such drift.

To do so, we came up with the idea that when the trust-based verification starts, the user under such verification may be either a potential adversary or the legitimate user is behaving differently. As depicted in Fig. 13:

1. When a user fails in the trust-based verification, i.e., trust drops below the allowed threshold, the action manager displays an alert message to the user saying: "An abnormal behavior has been observed, please confirm your identity so the system can automatically take this behavior as legitimate in the future."
2. Since we are also assuming that adversaries may compromise the controlling device or the security credentials, the verification should be performed using multi-factor authentication.
3. If the user succeeds in such verification, a new AD is trained with the accumulated operation commands during the trust-based verification. The choice of keeping the old

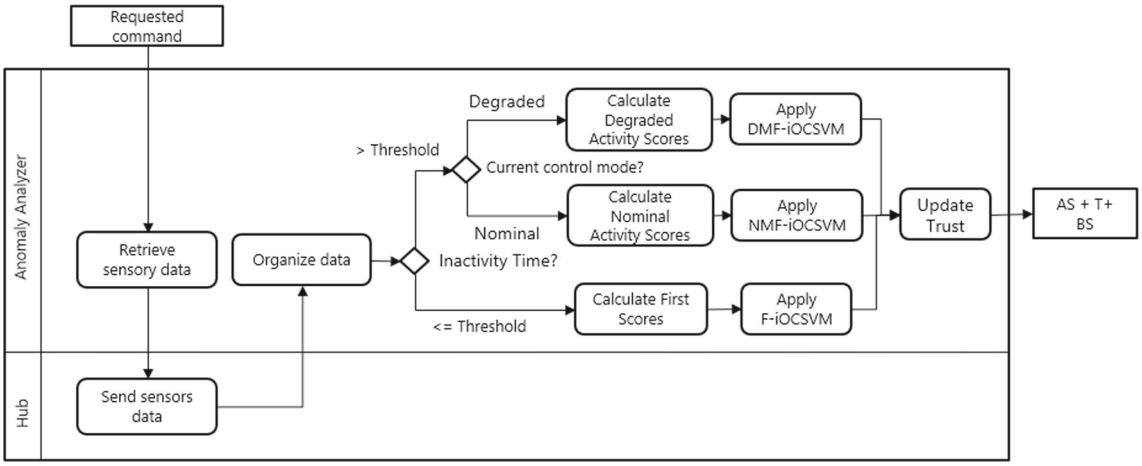

Fig. 12 Workflow of TRR-ICA-anomaly analyzer (TRR-ICA-AA)

Table 8 Types of AD models

Model	Training scores	Number of scores	Operation mode
S-AD	Starting scores	6	Nominal
NMA-AD	Activity scores	8	Nominal
DMA-AD	Activity scores	5	Degraded

AD model and training a new one allows preserving the old behavior of the legitimate user when he/she goes back to it, e.g., after the end of quarantine caused by covid-19.

4. Otherwise, if the user fails in multi-factor authentication, the other hand, the SHS recovery feature is performed (cf. Sect. 3.3.2) and the user is logged out.

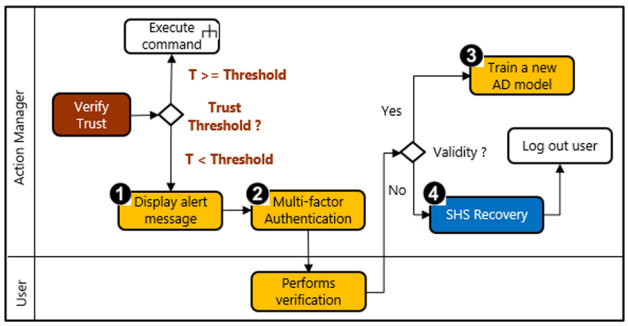

Fig. 13 Workflow of SA-TRR-ICA-action manager (SA-TRR-ICA-AM)

4 Empirical performance evaluation

In this section, we conduct an empirical evaluation to show the efficiency of our proposed framework through different anomaly detection and user authentication metrics. The choice of this validation methodology instead of others, such as formal verification [48,49], is mainly motivated by the adopted behavioral anomaly detection-based approach to detect malicious SHS devices operations. Indeed, anomaly detection uses a large amount of data collected and processed during both the enrollment and analysis stages, where AD models are trained and tested, respectively.

In this regard, we address the following research questions:

- RQ1: What is/are the best combination(s) of inactivity threshold for user session identification, and Day Segmentation to get a better authentication performance? (Sect. 4.3).
- RQ2: Using the obtained combination(s), what is the authentication performance of Duenna in different smart home layouts and devices? (Sect. 4.4).
- RQ3: How Duenna performs compared to similar frameworks? (Sect. 4.5).
- RQ4: What is the best value for the lockout threshold for an effective trust-based verification? (Sect. 4.6).
- RQ5: What is the computational complexity of incremental anomaly detection compared to the regular one? (Sect. 4.7).

4.1 Evaluation datasets

As described in the enrollment stage (Sect. 3.2.2), three types of data are used to train probabilistic and AD models viz., user

Table 9 Description of our adapted dataset

	House A	House B	House C
# of days	24	27	20
# of physical activities	10	13	16
# of control commands	2126	2580	1396
# of control commands per day (mean)	84	90	68

operation log, SHS states log, and user activity log. Although several datasets such as KDDCUP99 [50] and CICD2017 [51] have been created over the past two decades for evaluation purposes of BAD-based security systems, they do not include any specific characteristics of IoT-based systems as these datasets do not contain sensors' reading data or IoT network traffic [52]. Fortunately, to remedy the lack of such type of data, some IoT-specific datasets have been seen in the literature recently. In this section, we present the datasets which we use to evaluate our proposed framework. In particular, we propose an adaptation of a non-IoT-specific dataset to generate our SHS-specific dataset. Besides, we use another IoT-specific dataset that is publicly available.

4.1.1 Our adapted dataset

Since it has been extensively used in many ambient intelligence applications such as SHS recommender systems [53,54], the history data of user physical activities inside a smart environment can be easily found in public repositories. To generate an IoT-specific dataset from non-IoT-specific ones, we propose to adapt the history data of manual operation of appliances and objects by inhabitants in real-world home environments and assume that this data belongs to a controlling app manipulation.

The data we will be using for this purpose is the one collected by the University of Amsterdam [55] which contains three datasets (called houses A, B, and C). Each one of the houses was instrumented with wireless sensors (e.g., contact switches to measure open-close states of doors; pressure mats to measure lying in bed, etc.) to record the activities of one single inhabitant during several weeks. The activities include both manual operation of different house appliances (e.g., open door-lock, turn-on microwave) as well as daily living activities (e.g., cooking, sleeping, etc.). Inhabitants of three houses operate their appliances differently while performing different activities, as depicted in Table 9.

4.1.2 DS2OS dataset

Distributed Smart Space Orchestration System (DS2OS) is an open-source dataset publicly available on Kaggle [56]. It has been collected in a virtual IoT environment comprising a collection of micro-services during 24 hours. The simulated devices include eight types of devices viz., light controllers (22 instances), temperature controllers (20 instances), movement sensors (21 instances), washing machines (3 instances), battery sensors (6 instances), door lock sensors (5 instances), 4 heating control sensors (4 instances), and smartphones (3 instances). The size of the collected dataset is 357952 samples including 347935 samples representing normal behavior and 10017 samples representing anomalous data. Table 10 shows the frequency count and percentage of the considered attacks.

Table 10 Frequency distribution of considered attacks in DS2OS dataset

Attack	Frequency count	% of total data
Denial of service	5780	57.70
Scan	1547	15.44
Malicious control	889	08.87
Malicious operation	805	08.03
Spying	532	05.31
Data type probing	342	03.41
Wrong setup	122	01.21

4.2 Selection of anomaly detection model and evaluation methodologies

As explained in Sect. 3.2.2, Duenna considers incremental unsupervised AD models to be trained in the enrollment stage. While several models could be found in the literature [57], our selection is based on practical applications in the past and the attention in the scientific community. In particular, we use the One-Class Support Vector Machines (OCSVM) [58]. The advantages of using the OCSVM technique for anomaly detection are the theoretical support of the algorithm, being model-based, and fast in testing future instances. Besides, it has shown high performances in detecting anomalies in many other application domains compared to other AD models [59]. One incremental version of such model (i.e., iOCSVM) has been recently proposed and its efficiency has been shown compared to contemporary batch and incremental one-class classifiers [60]. In the following two sub-section, we present how iOCSVM is evaluated using each one of the described evaluation datasets.

4.2.1 Evaluation methodology using our adapted dataset

Our adapted dataset contains the data of three users each in one home environment; hence, three baselines (i.e., probabilistic models, operation commands' behavioral scores, and iOCSVM models) are built over the extracted logs of each user (i.e., user operation log, user activity log, and SHS states log).

To test the performance of the anomaly analyzer; thus, testing the iOCSVM, we would normally need labels that tell whether an operation command is malicious or not. Since our adapted dataset only contains legitimate users' data and does not include malicious operation commands, we propose to adopt a primary vs adversary strategy. Specifically, since the evaluation dataset contains the data of three users each in one home environment, three baselines (i.e., probabilistic models, operation commands behavioral scores and iOCSVM models) are built over the extracted logs of each user (i.e., user operation log, user activity log, and SHS states log). At each time, one inhabitant is designated as the primary user where his/her iOCSVM models are considered for the evaluation, whereas the two remaining inhabitants are considered as adversaries and their operation commands behavioral scores are considered as the testing data. After that, a part of the behavioral scores of the primary user is also included in the behavioral scores of the two adversaries. This process is then repeated, designating each of the other inhabitants as the legitimate user in turn.

To make sure that the performance evaluation results will not be biased or coincidental, cross-validation is employed. However, since we are dealing with a time-series data, the chronological order has to be maintained when splitting data into training and test datasets, i.e., instances included in the training set should be all earlier than the ones included in the test set. To do so, we employ a fivefold cross-validation on a rolling basis as depicted in Fig. 14.

In the first round:

1. We pick up the first 30% from the primary user's scores to train the AD models. Specifically, the scores are split into daily timeframes where the number of days equivalent to the percentage is rounded off. For instance, since the data of House A is recorded during 24 days, the first 30% of data denotes 7 days.
2. On one hand, the 30%-part (including starting and activity scores) is used to train S-iOCSVM and NMA-iOCSVM models, respectively.
3. On the other hand, the next 20%-part from the primary user's scores is picked up and then combined with the behavioral scores of each adversary to construct the final testing scores.

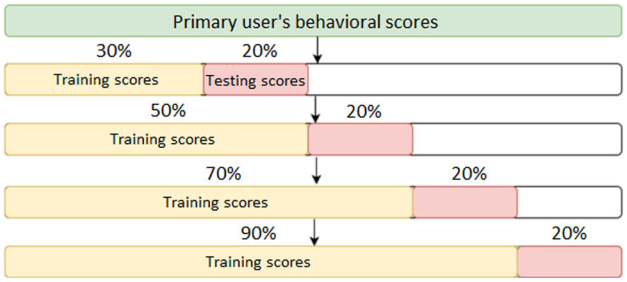

Fig. 14 Cross-validation on a rolling basis

However, in each one of the four following rounds:

1. We combine the 20%-part used for testing with the previous part. For instance, in the second round, we combine the 20%-part used for testing with the 30%-part used for training to get 50%-part of all data.
2. Then, we pick up the next 20%-part as the new testing scores.

In each one of all five rounds:

1. The previously trained AD models are exported to be tested on the corresponding final scores.
2. The anomaly detection results of both models are combined to be evaluated.

4.2.2 Evaluation methodology using DS2OS dataset

Since the collected DS2OS dataset refers only to one user, a single regular baseline (i.e., probabilistic models, operation commands behavioral scores, and iOCSVM models) is built over the extracted logs of this user (i.e., user operation log, user activity log, and SHS states log). The advantage of this dataset is that multiple attack scenarios are considered in the dataset and thus malicious operation commands are available. The behavioral malicious of anomalous commands from each considered attack are calculated using the legitimate user baseline.

To test the performance of the anomaly analyzer; thus, testing the iOCSVM, we also employ a similar fivefold cross-validation on a rolling basis (cf. Fig. 14).

4.3 Inactivity threshold and day segmentation tuning

During the enrollment stage, two variable parameters are used when preparing the history logs for the construction of the probabilistic models. Specifically, since we are following an inactivity-based identification strategy in extracting user session logs, choosing the best inactivity threshold to identify

Table 11 Day segmentation candidates

Candidate	# of periods	Segmentation
1	2	Night: ['00:00:00','07:59:59'],Day: ['08:00:00','23:59:59']
2	3	Night: ['00:00:00','07:59:59'], Day: ['08:00:00','17:59:59'], Evening: ['18:00:00','23:59:59']
3	4	Night: ['00:00:00','07:59:59'], Morning: ['08:00:00','11:59:59'],Afternoon: ['12:00:00','17:59:59'], Evening: ['18:00:00','23:59:59']

a session is challenging. Besides, in log segmentation where the corresponding period of the day is added to each record in the logs, choosing the best day division is also challenging. Since the behavioral scores used to train the iOCSVM models are calculated from the probabilistic models, different combinations of the two parameters would give different authentication results.

To find the combination (or the set of combinations) that better fits different history logs of the users from both evaluation datasets, different candidates of inactivity threshold and day segmentation are first applied to calculate behavioral scores. Then, the trained iOCSVM on the set of scores is evaluated using the Area Under the ROC Curve (AUC) metric. AUC measures the degree of separability of the iOCSVM between primary and adversary classes using various anomaly threshold settings. Finally, the optimal combinations are selected for training the final iOCSVM models to be evaluated. For the inactivity threshold, a range of values varying from 1 minute to 1 hour is employed to extract the sessions' logs. While for day segmentation, three candidates wherein the 24-hours-a-day are divided into two (2) or more periods with unequal lengths are used for logs segmentation, as depicted in Table 11.

Figures 15 and 16 show the AUC trend with different day segmentation and inactivity threshold candidates using our adapted dataset and DS2OS datasets, respectively. As one can see in these figures, manipulating different day segmentation and inactivity threshold candidates gives different AUC values. We can see that the AUC value certainly gets improved with small IT values. In particular, IT values smaller than 900 seconds and smaller than 100 seconds are the best values that we can use for extracting user sessions log regarding our adapted dataset and DS2OS datasets, respectively. However, the three segmentation candidates give almost similar AUC values for both datasets and do not have a considerable impact on the user behavioral pattern.

4.4 Authentication performance evaluation

After determining the best values of the two parameters for both evaluation datasets, we use this combination to train and test the performance of iOCSVM models following the same training and testing cross-validation process (cf. Fig. 14). This is called nested cross-validation (CV) as we are using two CV procedures. Since the goal of anomaly analyzer is to identify adversaries without incorrectly rejecting the legitimate user, we calculate the fraction of adversary operation commands that are incorrectly accepted, better known as False Acceptance Rate (FAR), whereas, to measure the legitimate user convenience level, we calculate the fraction of authorized operation commands that are incorrectly rejected, better known as False Rejection Rate (FRR). It should be noted that the chosen anomaly threshold (often called cutoff in ML terms) is 0 since the anomaly score varies in [-1, 1].

Tables 12 and 13 show the mean value of authentication metrics among the obtained fivefold values using our adapted dataset and DS2OS dataset, respectively. In particular, we have chosen a small value of inactivity threshold equal to 400 seconds and 100 seconds for the former and the latter dataset, respectively.

We can see that the FRR reaches in the worst case a value of 4.84% and 3.23% for our adapted dataset and DS2OS dataset, respectively. Such a low rate ensures a high level of user convenience since a legitimate user is rarely prompted to re-authenticate. On the other hand, the FAR successfully reaches the optimal value with 0.00% for our adapted dataset while reaching in the worst cases a value of 2.30% for the DS2OS dataset. Such a low rate makes sure that Duenna is

Table 12 Authentication performance results using our adapted dataset (N/A means that metric could not be calculated for primary vs itself)

	Primary								
	User 1			User 2			User 3		
Adversary	Accuracy	FAR	FRR	Accuracy	FAR	FRR	Accuracy	FAR	FRR
User 1	N/A			1.0000	0.0000	0.0206	0.9720	0.0000	0.0206
User 2	0.9831	0.0000	0.0476	N/A			1.0000	0.0000	0.0484
User 3	1.0000	0.0000	0.0452	0.9912	0.0000	0.0452	N/A		

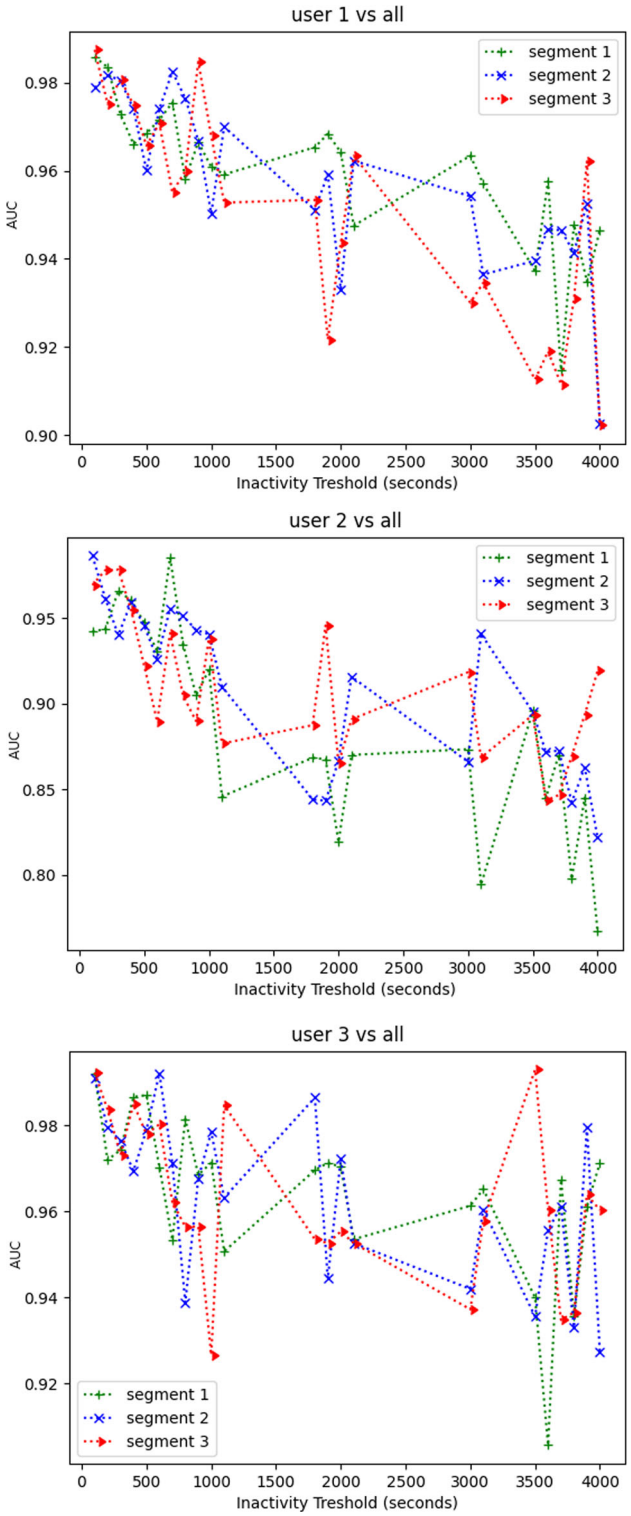

Fig. 15 AUC trend with different day segmentation and inactivity threshold candidates using our adapted dataset

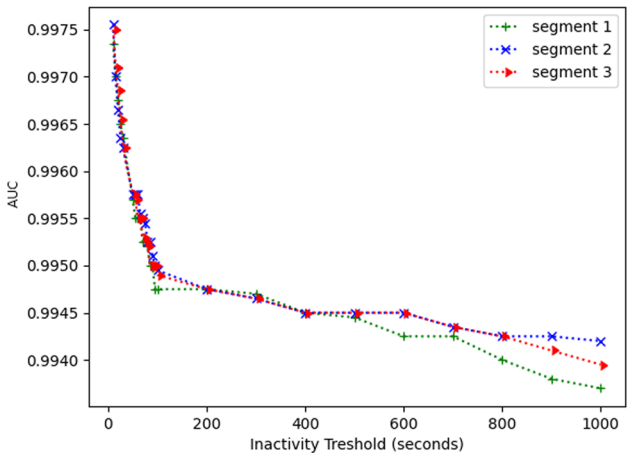

Fig. 16 AUC trend with different day segmentation and inactivity threshold candidates using DS2OS dataset

Table 13 Authentication performance results using DS2OS dataset

Attack	Accuracy	FAR	FRR
Denial of service	0.9776	0.0110	0.0074
Scan	0.9896	0.0000	0.0260
Malicious control	0.9926	0.0050	0.0010
Malicious operation	0.9960	0.0000	0.0104
Spying	0.9896	0.0026	0.0000
Data type probing	0.9963	0.0107	0.0007
Wrong setup	0.9995	0.0230	0.0323
Mean	**0.9944**	**0.0074**	**0.0109**

efficiently not accepting any adversary thus ensuring a high-security level. In particular, the low FAR value, obtained from DS2OS dataset which includes real attacks examples, has successfully shown that Duenna is resistant to them.

4.5 Comparison of Duenna with similar frameworks

In addition to the previous version of Duenna [17], some other existing frameworks have been also leveraging BAD-based security approach to secure IoT-based systems. For those specifically designed to protect SHS devices from unauthorized operation, we can find SoftAuthZ [15], [18], and [13] (cf. Sect. 2.3.2), whereas other frameworks have been generally designed for IoT-based systems viz., [61], [62], and [63].

Table 14 summarizes the AD techniques used by these frameworks as well as the datasets and the performance metrics used for their evaluation. In particular, the previous version of Duenna [17], SoftAuthZ [15], [18], and [13] has only conducted simulation-driven experiments for performance evaluation, whereas [61], [62], and [63] have been evaluated using the real-world DS2OS dataset. In contrast, our proposed framework has been evaluated using both our real-world adapted dataset as well as DS2OS dataset.

Figures 17 and 18 show the comparison of Duenna with the presented similar frameworks by anomaly detection accuracy and FRR, respectively. In terms of accuracy, evaluated on our adapted dataset, Duenna outperforms similar works that are evaluated on their own synthetic datasets but not those evaluated on DS2OS. However, applied on this latter dataset, Duenna outperforms similar work evaluated on such dataset. In particular, in terms of False Rejection Rate (FRR), it outperforms other frameworks when evaluated on both our adapted dataset as well as on DS2OS dataset.

We explain this outperforming that Duenna uses a multitude of behavioral scores in contrast to SoftAuthZ [15] and [13] that only use a single user behavioral feature viz., variability between operation commands and operation sequence, respectively. Besides, incremental OCSVM adopted by Duenna gives higher performance than other AD techniques such as Linear Regression [15], the batch OCSVM [17], Clustering [61], and Random Forest [63].

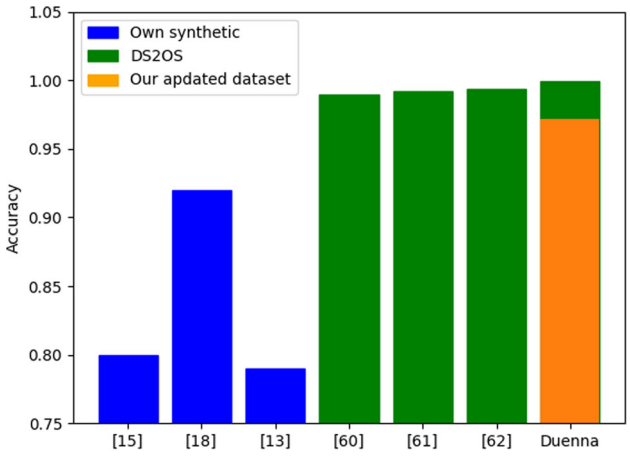

Fig. 17 Comparison of Duenna with similar frameworks by accuracy

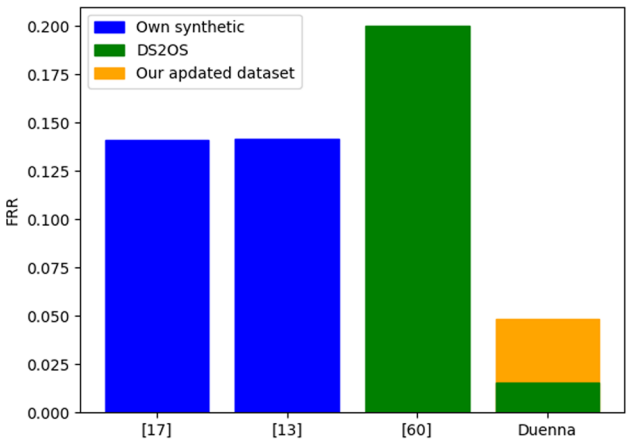

Fig. 18 Comparison of Duenna with similar frameworks by false rejection rate

4.6 Lockout threshold tuning

As discussed in the TRR-ICA action manager (cf. Sect. 3.3.2), a user is logged out when his/her trust reaches a specific value called lockout threshold. It is challenging to choose such a threshold since a value that is near the reward upper bound (parameter C in Equation 1) is likely to lock the legitimate user very quickly. This choice makes iOCSVM very sensitive to FRR. On the other hand, a threshold that is distant from the reward upper bound is likely to let adversaries manipulating devices for too long before getting locked making iOCSVM models very sensitive to FAR. Consequently, it is important to choose the best threshold assuring that an impostor user can do as little as possible while not locking legitimate too quickly. For doing so, we measure two metrics [46] to follow the impact of different threshold values: 1) Average Number

Table 14 Comparison of Duenna with similar frameworks

Work	AD technique	Dataset used	Performance metrics
[17]	OCSVM	Own synthetic	FRR=0.0412
[15]	Linear regression		Accuracy=0.80
[18]	Association rules		Accuracy=0.92
[13]	Trees		Accuracy=0.79 FRR=0.1447
[61]	BIRCH Clustering	DS2OS	Accuracy=0.99 FRR=0.2
[62]	Random Neural Network (RaNN)		Accuracy=0.9920
[63]	Random Forest		Accuracy=0.994
Duenna	iOCSVM	Our adapted dataset	Accuracy=0.9720 FRR=0.0484
		DS2OS	Accuracy=0.9944 FRR=0.0153

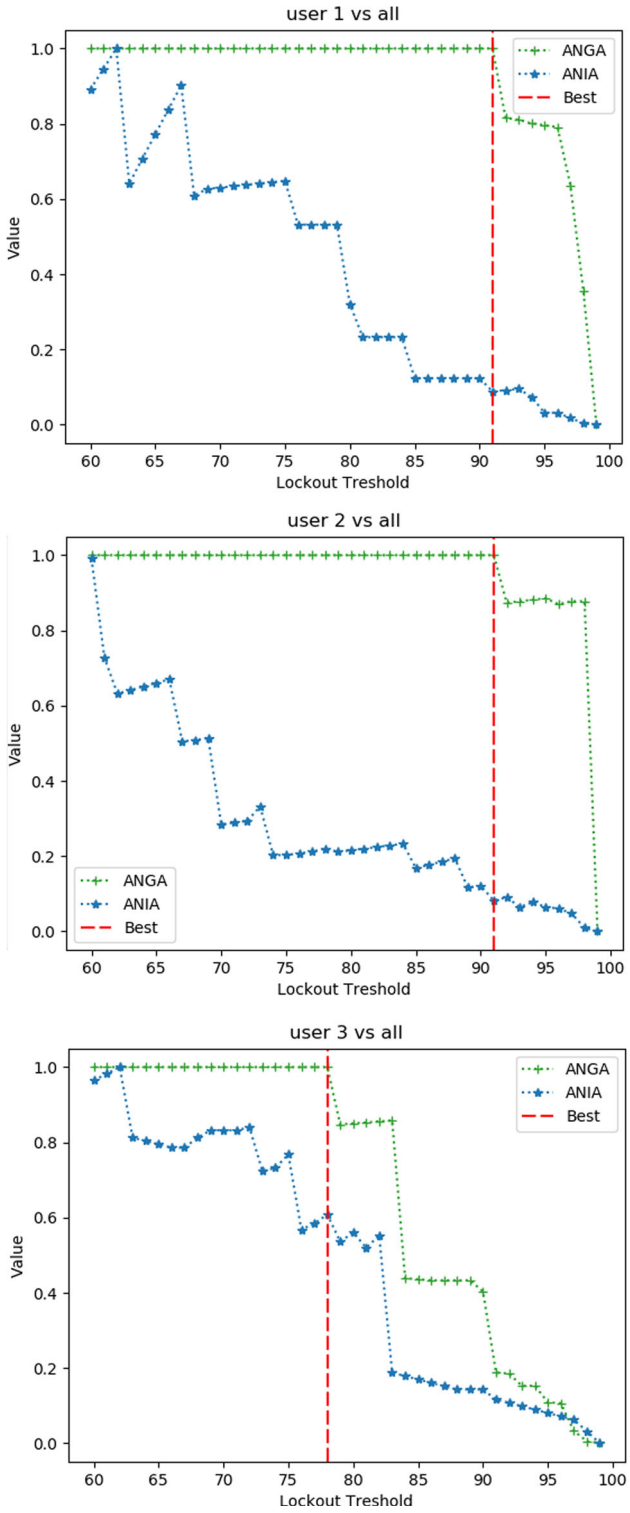

Fig. 19 ANGA and ANIA trend with different lockout threshold values using our adapted dataset

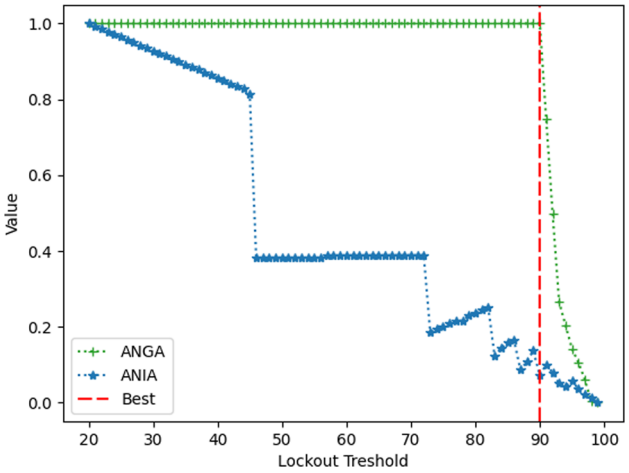

Fig. 20 ANGA and ANIA trend with different lockout threshold values using DS2OS dataset

of Genuine Actions (ANGA) which is the average number of operation commands requested by the genuine user before locked by the system, and 2) Average Number of Impostor Actions (ANIA) which is the average number of operation commands requested by the adversarial user before locked by the system. The goal is obviously to have ANGA as high as possible, while at the same time the ANIA value must be as low as possible.

Figures 19 and 20 show the trend of ANGA and ANIA using different values of lockout threshold for our adapted dataset and DS2OS dataset, respectively. It should be noted that the values of ANGA and ANIA are scaled in the range of [0, 1] for good visualization. In the beginning of both figures, we can see that as long as the lockout threshold increases, ANIA value decreases while ANGA stays steady. This is a good indicator since our objective is to maximize ANGA and minimize ANIA as possible. However, ANGA begins to decrease starting from lockout threshold = 91, 91, 78 for our adapted dataset and starting from lockout threshold = 90 for DS2OS dataset. Since the decrease in AGNA is; we pick the minimum threshold among the three values for the first dataset (i.e., 78) and 90 for the second one. We consider the chosen values as the best thresholds that gives a good combination of both ANGA and ANIA values.

Using the selected lockout threshold for our adapted dataset, Table 15 shows the mean value of AGNA and ANIA for all possible primary vs adversaries (NA means that ANGA could not be calculated as it only concerns the primary user against itself), whereas, for DS2OS datasets, we have found that the mean value of AGNA and ANIA is 1245 and 11, respectively.

Table 15 AGNA and ANIA results using our adapted dataset

	Primary User 1		User 2		User 3	
Adv.	**AGNA**	**ANIA**	**AGNA**	**ANIA**	**AGNA**	**ANIA**
U1	745	N/A	N/A	5.75	N/A	9.08
U2	N/A	5.88	569	N/A	N/A	7.18
U3	N/A	5.88	N/A	7.97	860	N/A

From the obtained results from both datasets, we can see that the big values of ANGA show that the legitimate user is never locked out, whereas, for ANIA we can see that an adversary can perform at most nine (9) actions before getting logged out. We recall that the action manager recovers devices to their initial states although these nine operation commands have been executed (cf. SHS Recovery, Sect. 3.3.2).

4.7 Evaluation of computational complexity

In our previous work [17], the regular batch OCSVM has been adopted as the AD model, whereas, in this paper, we adopt the iOCSVM which can be updated incrementally. Yet, this interesting feature needs more computational time and resources. Figure 21 shows the required training time for regular OCSVM and incremental OCSVM for different sizes of training data. We can see that iOCSVM exponentially needs more time to be trained when data size increases compared to the regular one.

However, we argue that besides the advantage of updating iOCSVM without being re-trained from scratch in contrast to regular OCSVM, the added delay in training time does not pose a problem. On one hand, since the training of iOCSVM models in the first stage is done offline before user operation commands are being analyzed, the time needed for the training does not affect the user experience and can be accepted. On the other hand, during the interactive operation, iOCSVM models are only updated with new executed commands (i.e., after the user session has finished). Using the optimal values of inactivity threshold and day segmentation, we have found that the maximum length of user sessions does not surpass 50 operation commands. As shown in Fig. 22, with such a small number of training instances, iOCSVM does not need a remarkable training time compared to the regular OCSVM.

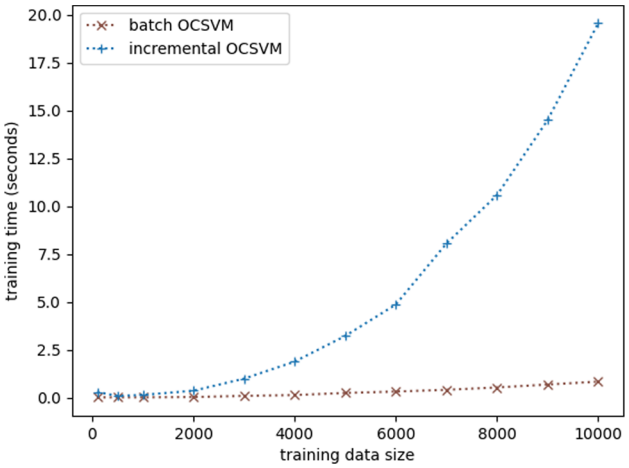

Fig. 21 Log of training time (per model) in seconds for regular OCSVM and incremental OCSVM for different sizes of training data

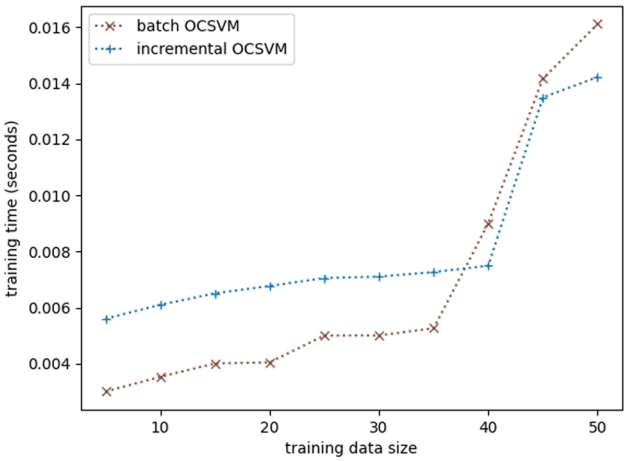

Fig. 22 Log of training time in seconds for regular OCSVM and incremental OCSVM for different length of user sessions

5 Conclusion

Compromising security credentials used to authenticate to the controlling app, hijacking the controlling device itself (e.g., smartphone), or being manipulated by informed adversaries such as family members are some of the serious security concerns for smart home systems (SHS) consumers. Indeed, such concerns are because adversaries may exploit them to maliciously operate SHS devices. Without integrating the behavior analysis of both SHS and its users, current techniques cannot efficiently prevent the unauthorized operation of SHS devices.

To address this problem, this paper presents a new anomalous behavior detection-based approach. In particular, we hypothesized that SHS users tend to follow typical and distinctive behavioral patterns when operating their IoT devices. Therefore, users that attempt to operate devices differently from such a regular behavior are considered malicious. Such an approach was our basis to build Duenna: an authentication framework for SHS users. Duenna secures SHS IoT devices from unauthorized operation by continuously authenticating operation commands and only allowing those that have been seen in the previously built behavioral patterns of the legitimate user.

We have conducted an extensive empirical evaluation on real-world SHS datasets including our adapted dataset and DS2OS dataset. Performance results obtained from experiments validate the ability of Duenna to differentiate between different SHS users as well as its resistance to real attacks. Specifically, Duenna exhibits high authentication rates in terms of anomaly detection accuracy, false acceptance rate, and false rejection rate; thus, ensuring both security and user experience. These results also show that such a user behavior-based approach is a promising security scheme that could be integrated into existing SHSs.

As a part of future work, we aim to investigate how the anomalous users' patterns that become available from rejected commands can be used to enhance authentication rates. Besides, we also aim to leverage BAD-based security approach to secure other IoT-based systems.

Funding Not applicable.

Availability of data and material Raw data is publicly available for download from [55], whereas our own datasets adapted from such raw data could be provided after the acceptance and publication of this paper.

Declarations

Conflicts of interest/Competing interests Not applicable.

Code availability Not applicable.

Disclosure of potential conflicts of interest Noureddine Amraoui declares that he has no conflict of interest. Belhassen Zouari declares that he has no conflict of interest.

Research involving human participants and/or animals This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Not applicable.

References

1. Guth, J., Breitenbücher, U., Falkenthal, M., Fremantle, P., Kopp, O., Leymann, F., Reinfurt, L.: A detailed analysis of IoT platform architectures: concepts, similarities, and differences. In: Internet of everything (Springer, 2018), pp. 81–101
2. Fernandes, E., Rahmati, A., Feamster, N.: New Problems and Solutions in IoT Security and Privacy. arXiv preprint arXiv:1910.03686 (2019)
3. Miessler, D.: Securing the internet of things: Mapping attack surface areas using the OWASP IoT top 10. In: RSA Conference (2015)
4. Gamundani, A.M., Phillips, A., Muyingi, H.N.: An Overview of Potential Authentication Threats and Attacks on Internet of Things (IoT): A Focus on Smart Home Applications. In: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (IEEE, 2018), pp. 50–57
5. Ling, Z., Luo, J., Xu, Y., Gao, C., Wu, K., Fu, X.: Security vulnerabilities of internet of things: a case study of the smart plug system. IEEE Internet Things J. **4**(6), 1899 (2017)
6. Martin, V., Cao, Q., Benson, T.: Fending off IoT-hunting attacks at home networks. In: Proceedings of the 2nd Workshop on Cloud-Assisted Networking (2017), pp. 67–72
7. Zarpelão, B.B., Miani, R.S., Kawakani, C.T., de Alvarenga, S.C.: A survey of intrusion detection in Internet of Things. J. Netw. Computer Appl. **84**, 25 (2017)
8. Liang, Y., Samtani, S., Guo, B., Yu, Z.: Behavioral biometrics for continuous authentication in the Internet-of-Things Era: an artificial intelligence perspective. IEEE Internet Things J. **7**(9), 9128 (2020)
9. Shah, S.W., Kanhere, S.S.: Recent trends in user authentication-a survey. IEEE Access **7**, 112505 (2019)
10. Dutta, S., Chukkapalli, S.S.L., Sulgekar, M., Krithivasan, S., Das, P.K., Joshi, A., et al. Context Sensitive Access Control in Smart Home Environments. In: 6th IEEE International Conference on Big Data Security on Cloud (BigDataSecurity 2020) (2020)
11. Omolola, O., More, S., Fasllija, E., Wagner, G., Alber, L.: Policy-based access control for the IoT and Smart Cities. Open Identity Summit **2019**,(2019)
12. Singh, M.P., Sural, S., Atluri, V., Vaidya, J.: Security Analysis of Unified Access Control Policies. In: International Conference On Secure Knowledge Management In Artificial Intelligence Era (Springer, 2019), pp. 126–146
13. Yamauchi, M., Ohsita, Y., Murata, M., Ueda, K., Kato, Y.: Anomaly detection in smart home operation from user behaviors and home conditions. IEEE Transactions Consumer Electron. **66**(2), 183 (2020)
14. Wazid, M., Das, A.K., Odelu, V., Kumar, N., Conti, M., Jo, M.: Design of secure user authenticated key management protocol for generic IoT networks. IEEE Internet Things J. **5**(1), 269 (2017)
15. Ghosh, N., Chandra, S., Sachidananda, V., Elovici, Y.: SoftAuthZ: a context-aware, behavior-based authorization framework for home IoT. IEEE Internet Things J. **6**(6), 10773 (2019)
16. McCarthy, J., Powell, M., Stouffer, K., Tang, C.Y., Zimmerman, T., Barker, W., Ogunyale, T., Wynne, D., Wiltberger, J.: Securing Manufacturing Industrial Control Systems: Behavioral Anomaly Detection. National Institute of Standards and Technology (NIST), Gaithersburg (2018)
17. Amraoui, N., Besrour, A., Ksantini, R., Zouari, B.: Implicit and continuous authentication of smart home users. In: International Conference on Advanced Information Networking and Applications (Springer, 2019), pp. 1228–1239
18. Rath, A.T., Colin, J.N.: Strengthening access control in case of compromised accounts in smart home. In: 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (IEEE, 2017), pp. 1–8
19. Brian, B.: Sustainability hacks: automatic window control. Sustainability hacks: automatic window control. https://hackaday.com/2011/09/29/sustainability-hacks-automatic-window-control/ (2011). [Online; accessed 15-April-2021]
20. Wang, C., Yang, B.: Composite behavioral modeling for identity theft detection in online social networks. arXiv preprint arXiv:1801.06825 (2018)
21. Xiao, Y., Jia, Y., Liu, C., Alrawais, A., Rekik, M., Shan, Z.: HomeShield: a credential-less authentication framework for smart home systems. IEEE Internet Things J. **7**(9), 7903 (2020)
22. Zhang, Z., Guan, Y., Ma, X., Yu, T., Zhang, L.: Sovereign: User-Controlled Smart Homes. arXiv preprint arXiv:2006.06131 (2020)
23. Samsung. Smartthings, one simple home system. w world of possibilities. https://www.smartthings.com/ (2021). [Online; accessed 15-April-2021]
24. Sikder, A.K., Babun, L., Aksu, H., Uluagac, A.S.: Aegis: a context-aware security framework for smart home systems. In: Proceedings

of the 35th Annual Computer Security Applications Conference (2019), pp. 28–41
25. Román-Castro, R., López, J., Gritzalis, S.: Evolution and trends in iot security. Computer **51**(7), 16 (2018)
26. Megan, W.: Have a smart lock? Yeah, it can probably be hacked. https://www.cnet.com/news/have-a-smart-lock-yeah-it-can-probably-be-hacked/ (2016). [Online; accessed 15-April-2021]
27. Jaikumar, V.: Target attack shows danger of remotely accessible HVAC systems. https://www.computerworld.com/article/2487452/target-attack-shows-danger-of-remotely-accessible-hvac-systems.html (2014). [Online; accessed 15-April-2021]
28. Jack, L.: Half baked IoT stove could be used as a remote controlled arson device. https://hackaday.com/2017/04/20/half-baked-iot-stove-could-be-used-as-a-remote-controlled-arson-device/ (2017). [Online; accessed 15-April-2021]
29. Darlene, S.: Researchers hack Philips Hue lights via a drone; IoT worm could cause city blackout. https://www.computerworld.com/article/3139860/researchers-hack-philips-hue-lights-via-a-drone-iot-worm-could-cause-city-blackout.html (2016). [Online; accessed 15-April-2021]
30. Brad, R.: The ultimate nightmare: Researchers learn how to hack connected 'smart home' toilets. https://bgr.com/2014/06/12/smart-home-toilets-hacked/ (2014). [Online; accessed 15-April-2021]
31. Dmitry, D., Elena, P., Anna, C., Tatiana, Z., Elena, P.: Approaches to Anomaly Detection in Web Application Intrusion Detection Systems. In: 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT) (IEEE, 2020), pp. 532–535
32. Liao, Q., Li, H., Kang, S., Liu, C.: Application layer DDoS attack detection using cluster with label based on sparse vector decomposition and rhythm matching. Secur. Commun. Netw. **8**(17), 3111 (2015)
33. Najafabadi, M.M., Khoshgoftaar, T.M., Calvert, C., Kemp, C.: User behavior anomaly detection for application layer DDoS attacks. In: 2017 IEEE International Conference on Information Reuse and Integration (IRI) (IEEE, 2017), pp. 154–161
34. S. Mathew, M. Petropoulos, H.Q. Ngo, S. Upadhyaya. A data-centric approach to insider attack detection in database systems. In: International Workshop on Recent Advances in Intrusion Detection (Springer, 2010), pp. 382–401
35. Mazzawi, H., Dalal, G., Rozenblatz, D., Ein-Dorx, L., Niniox, M., Lavi, O.: Anomaly detection in large databases using behavioral patterning. In: 2017 IEEE 33rd International Conference on Data Engineering (ICDE) (IEEE, 2017), pp. 1140–1149
36. Viswanath, B., Bashir, M.A., Crovella, M., Guha, S., Gummadi, K.P., Krishnamurthy, B., Mislove, A.: Towards detecting anomalous user behavior in online social networks. In: 23rd {USENIX} Security Symposium ({USENIX} Security 14) (2014), pp. 223–238
37. Ruan, X., Wu, Z., Wang, H., Jajodia, S.: Profiling online social behaviors for compromised account detection. IEEE Transactions Information Forensic. Secur. **11**(1), 176 (2015)
38. Sánchez, P.M.S., Valero, J.M.J., Celdrán, A.H., Bovet, G., Pérez, M.G., Pérez, G.M.: A Survey on Device Behavior Fingerprinting: Data Sources, Techniques, Application Scenarios, and Datasets. arXiv preprint arXiv:2008.03343 (2020)
39. Kayacik, H.G., Just, M., Baillie, L., Aspinall, D., Micallef, N.: Data driven authentication: On the effectiveness of user behaviour modelling with mobile device sensors. arXiv preprint arXiv:1410.7743 (2014)
40. Birnbach, S., Eberz, S.: Peeves: Physical Event Verification in Smart Homes. (2019)
41. He, W., Golla, M., Padhi, R., Ofek, J., Dürmuth, M., Fernandes, E., Ur, B.: Rethinking access control and authentication for the home internet of things (IoT). In: 27th {USENIX} Security Symposium ({USENIX} Security 18) (2018), pp. 255–272
42. Goldstein, M., Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one **11**(4), e0152173 (2016)
43. Bisong, E.: Batch vs. online learning. In: Building Machine Learning and Deep Learning Models on Google Cloud Platform, pp. 199–201. Apress, Berkeley, CA (2019)
44. Chen, Z., Tian, L., Lin, C.: Trust evaluation model of cloud user based on behavior data. Int. J. Distributed Sensor Netw. **14**(5), 1550147718776924 (2018)
45. Bours, P.: Continuous keystroke dynamics: a different perspective towards biometric evaluation. Information Secur. Tech. Rep. **17**(1–2), 36 (2012)
46. Mondal, S., Bours, P.: A continuous combination of security & forensics for mobile devices. J. Information Secur. Appl. **40**, 63 (2018)
47. Robert, T., Fabre, J.-C., Roy, M.: Application of early error detection for handling degraded modes of operation. In: 12th European Workshop on Dependable Computing, EWDC 2009 (2009)
48. Abdalla, M., Fouque, P.A., Pointcheval, D.: Password-based authenticated key exchange in the three-party setting. In: International Workshop on Public Key Cryptography (Springer, 2005), pp. 65–84
49. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuéllar, J., Drielsma, P.H., Héam, P.-C., Kouchnarenko, O., Mantovani, J., et al. The AVISPA tool for the automated validation of internet security protocols and applications. In: Etessami, K., Rajamani, S.K. (eds.) Computer Aided Verification. CAV 2005. Lecture Notes in Computer Science, vol 3576. Springer, Berlin, Heidelberg
50. Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (1999). [Online; accessed 15-April-2021]
51. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
52. Essop, I., Ribeiro, J.C., Papaioannou, M., Zachos, G., Mantas, G., Rodriguez, J.: Generating datasets for anomaly-based intrusion detection systems in IoT and industrial IoT networks. Sensors **21**(4), 1528 (2021)
53. Rasch, K.: An unsupervised recommender system for smart homes. J. Ambient Intell. Smart Environ. **6**(1), 21 (2014)
54. van Kasteren, T.L.M., Englebienne, G., Kröse, B.J.A.: Human activity recognition from wireless sensor network data: benchmark and software. In: Activity Recognition in Pervasive Intelligent Environments, pp. 165–186. Atlantis Press (2011)
55. van Kasteren, T.L., Englebienne, G., Kröse, B.J.: University of Amsterdam activity recognition dataset. http://casas.wsu.edu/datasets/ (2011). [Online; accessed 15-April-2021]
56. Francois-Xavier, A.: IoT traffic traces gathered in a the DS2OS IoT environment. https://www.kaggle.com/francoisxa/ds2ostraffictraces (2018). [Online; accessed 15-April-2021]
57. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: a survey. IEEE Transactions Knowl. Data Eng. **24**(5), 823 (2010)
58. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443 (2001)
59. Garcia-Font, V., Garrigues, C., Rifà-Pous, H.: A comparative study of anomaly detection techniques for smart city wireless sensor networks. Sensors **16**(6), 868 (2016)
60. Kefi-Fatteh, T., Ksantini, R., Kaâniche, M.B., Bouhoula, A.: A novel incremental one-class support vector machine based on low variance direction. Pattern Recognition **91**, 308 (2019)
61. Pahl, M.O., Aubet, F.X.: All eyes on you: Distributed Multi-Dimensional IoT microservice anomaly detection. In: 2018 14th

International Conference on Network and Service Management (CNSM) (IEEE, 2018), pp. 72–80
62. Latif, S., Zou, Z., Idrees, Z., Ahmad, J.: A novel attack detection scheme for the industrial internet of things using a lightweight random neural network. IEEE Access **8**, 89337 (2020)
63. Hasan, M., Islam, M.M., Zarif, M.I.I., Hashem, M.: Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things **7**, 100059 (2019)