Preserving Privacy in the Globalized Smart Home:The SIFIS-Home Project

Preserving Privacy in the Globalized Smart Home: The SIFIS-Home Project

Luca Ardito, *Member, IEEE* Luca Barbato, Paolo Mori and Andrea Saracino

***Abstract*—This paper introduces the SIFIS-Home project, which defines and enforces privacy and security policies in a Smart Home environment. The project allows users to control data privacy and developers to provide privacy-aware services to mitigate the privacy risk of a globalized Smart Home by leveraging the concept of the Smart Home cyber-perimeter.**

***Index Terms*—Data Privacy, IoT, Security Policy, Distributed Systems, Smart Home**

I. Introduction and Background

Smart Home is an application paradigm that has been gaining popularity in the last few years. The Internet of Things (IoT) has recently fostered a vision of Smart Home systems, where users can automatically install connectable (smart) devices and appliances that cooperate to manage home services and functionalities. This emerging market is rapidly attracting software developers to produce novel applications and services to provide additional Smart Home services and functionalities. While this integration has been a relevant driver for the increased distribution of the Smart Home paradigm, it intrinsically introduced a set of security and privacy concerns. In fact, a Smart Home environment is bound to produce a large set of extremely sensitive data. Voice recordings, requested services, daily habits, biometrics, behavioral patterns can all be used to profile users, inferring their preferences. In fact, service providers can use these sensitive data to provide targeted advertisement, perform market-basket analysis, and even profit on the raw value of acquired data because the majority of these integrated Smart Home services are cloud-based. This kind of system implies that any information, piece of data or provided command needs to be processed outside the Smart Home premises. Thus, the users (home tenants) lose control of their data, how they are used, and their redistribution. Moreover, these integration services push IoT device producers and application developers to provide devices and applications integrated with these cloud services, increasing thus the Smart Home dependency on these globalized integrators. Furthermore, it is worth noting that third party applications might also imply safety issues in a Smart Home. In fact, Smart Homes are critical cyberphysical environments [1], involving both sensors and actuators which can potentially cause physical damage to both objects and home tenants. At the same time, users are becoming more aware of cybersecurity related risks. In fact, according to recent research, privacy and security concerns is considered one of the main barriers toward global Smart Home adoption [2]. For all these reasons, mechanisms to verify the quality and security of third party applications, paired with proactive security mechanisms for data privacy and security policy enforcement, are becoming a requirement for Smart Home environments.

This paper presents the software components of the *SIFIS-Home* (Secure Interoperable Full-stack IoT for Smart Home) project, namely the SIFIS-Home framework, for the management of Smart Home security and privacy, and the development tools which enable third-party developers to implement SIFIS-Home aware applications (depicted in Figure 1). The concept at the basis of the SIFIS-Home framework is that the Smart Home user should have full control over the data produced by the Smart Home devices. The SIFIS-Home framework aims to design, implement, and validate a secure software framework for guaranteeing privacy and resiliency in Smart Home systems, where a (large) number of different devices (both from the point of view of functionality and computational capabilities) cooperate in implementing complex functionalities.

The SIFIS-Home framework leverages secure communication and management protocols suitable for the IoT, full-lifecycle evaluation and management of software security, machine learning-based distributed intrusion detection mechanisms, and privacy-preserving data management and analysis techniques. A relevant feature characterizing the SIFIS-Home framework is that it operates at all Smart Home system stack levels, from the device kernel, through network and application, to the user level, thus guaranteeing a complete integration with the system it protects.

SIFIS-Home considers third party application developers as relevant stakeholders, providing them with APIs, guidelines and automated self-assessment tools to develop certifiable security policy-compliant and privacy-aware applications. Both IoT devices and the applications running on them can be restricted on managing data on the Smart Home cyber-perimeter, i.e., the virtual perimeter separating those services and applications where data are not shared with external entities, from the ones potentially implying a loss of control on processed data. In particular, only specific

This work is supported by the SIFIS-Home H2020 funded project, GA n. 952652

L. Ardito is with the Dept. of Control and Computer Engineering, Politecnico di Torino, Turin, Italy (e-mail: {luca.ardito}@polito.it), L. Barbato is with Luminem, Turin, Italy, {luca.barbato}@luminem.it), P. Mori and A. Saracino are with Istituto di Informatica e Telematica - Consiglio Nazionale delle Ricerche, Pisa, Italy (e-mail: {first.last}@iit.cnr.it).

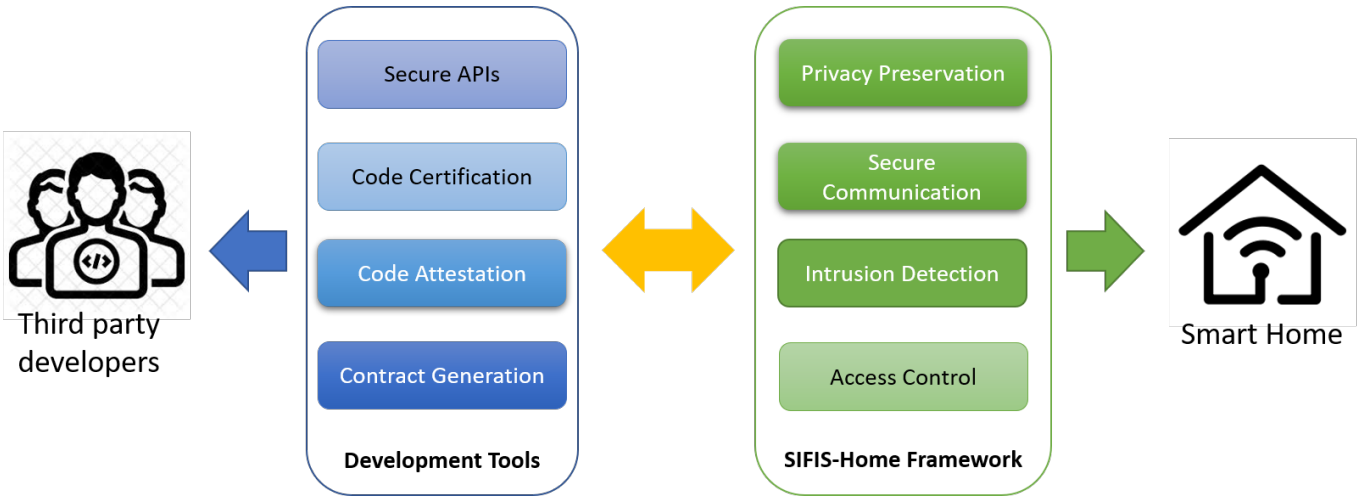

Fig. 1: The SIFIS-Home software components.

actions are allowed on data inside the cyber-perimeter, and unexpected behaviour are flagged and tackled according to the user-specified policies, aimed at increasing data privacy and user control.

Finally, SIFIS-Home provides a user-friendly interface to Smart Home users and administrators to efficiently exploit the provided functionalities according to their roles. In particular, the SIFIS-Home framework allows its users to assess the security and reliability of the services and applications they choose to install in their Smart Home system, enabling the definition of simple and easy readable security and management policies to regulate the operations that such services and applications can perform. These policies are enforced by the SIFIS-Home framework at all time, thus ensuring users' safety, security and privacy. The enhanced security and privacy support provided by the SIFIS-Home framework would improve users' trust toward the acquisition of new Smart Home services to be integrated into their Smart Home systems.

As described by Lin and Bergmann in [3], and by Touqeer et al. in [4], security and privacy challenge Smart Home environments. In the literature, it is possible to find specific frameworks that deal with those issues. For example, Varghese and Hayajneh in [5] provide a framework which determines if a Smart Home device is safe for the user to use at home. Hussain and Qui in [6] identify IoT devices based on their communication behaviours and study Security and Privacy Schemes for Smart Devices to prevent intruder men in the middle attack. These frameworks are specific and aim to validate devices and counteract wrong and malicious actions that can be done on them.

Nowadays, there is a rising interest in this topic, with the major players in internet services and IoT software actively involved. There are several projects like:

- Web of Things[1] that describes a set of standards to counter the fragmentation of the IoT devices.
- OpenHAB[2] that is a vendor and technology agnostic open source automation software for the Smart Home.
- MATTER [3] that is a unified connectivity protocol for IoT devices, which promises reliable, secure connectivity between IoT devices.

The SIFIS-Home project can coexist with those frameworks, standards, and technologies because, from an architectural point of view, SIFIS-Home is above the devices communication level and introduces privacy issues not covered at this level in the literature.

Summarizing, the contributions of the paper are the following:

- We introduce SIFIS-Home a distributed P2P framework for managing security, privacy and safety in Smart Home environments;
- We present the concept of Smart Home cyber-perimeter and its implications to security and privacy policies;
- We present the SIFIS-Home ACS (Attestation, Certification and Security Evaluation) toolbox for evaluating quality and security of third-party Smart Home applications;
- We introduce a high level view of the SIFIS-Home Architecture and the mechanisms used for proactive security enforcement.

The remainder of the paper is structured as follows: Section II introduces the concept of cyber-perimeter, Section III describes the process of certifying the app behavior, Section IV introduces the SIFIS-Home architecture, Section V provides a simple use case, and finally, Section VI concludes the paper.

[1]https://www.w3.org/WoT

[2]https://www.openhab.org/

[3]https://buildwithmatter.com/

II. The Smart Home Cyber-Perimeter

Smart Home globalization implies that many devices and services running in a Smart Home rely on external or cloud services. The data flow between the external service providers and the Smart Home devices is necessary to provide effective smart services. However, this might imply the disclosure of sensitive information since there is limited control from the Smart Home users on the amount, type and sensitivity of data sent to these services.

Protecting the Smart Home and its users from unintended disclosure of sensitive information requires defining a logical distinction between the outside and inside of the Smart Home. The separation between these two domains is what we define as the *Smart Home cyber-perimeter*. In particular, the cyber-perimeter is a logical barrier, which identifies the elements (i.e., devices and application) of the Smart Home, which can be used to receive and send data toward entities that are not part of the Smart Home (i.e., external entities). For example, a smart speaker with an embedded voice assistant exploiting a cloud service to process voice commands is both an access and exit point of the Smart Home cyber-perimeter. Though even inside the Smart Home cyber-perimeter, there might be specific privacy constraints, a violation of the Smart Home privacy is performed when sensitive information leaves the Smart Home cyber-perimeter. We base this definition of privacy violation on the worst possible case: once a piece of data leaves the Smart Home cyber-perimeter, the users potentially lose control of that data piece, which can thus be re-used and redistributed indefinitely. We derive that data can be safely exchanged among devices and services inside the Smart Home cyber-perimeter. The rationale behind this distinction is in the trade-off between ensured privacy and needed accuracy for data analysis algorithms, which are essential to provide smart services. Thus, inside the Smart Home cyber-perimeter, data can be exchanged and processed without applying privacy-enhancing techniques to maximize data analysis accuracy, providing the best service level. In fact, inside the cyber-perimeter (Figure 2), data cannot be shared with external entities and remains only available to the data owner, i.e., the Smart Home residents. The validity of this assumption depends on the devices and applications behaviour on the cyber-perimeter, which act as access points to the Smart Home. Their behaviours should be monitored and certified, when possible, to ensure that when they have access to sensitive information, they are not going to send them outside out of the cyber-perimeter. When this cannot be ensured, or it is known that a data piece is bound to leave the perimeter, it should be processed through specific privacy-enhancing techniques to avoid disclosing sensitive information. Trading-off data privacy and analytics accuracy is a hot research topic that we are also exploring in the SIFIS-Home project, still it is out of the scope of this work.

In SIFIS-Home, the cyber-perimeter is an essential element for defining data policies and regulating the access to specific functionalities with possible privacy implications, such as reading the video streaming of an indoor surveillance camera. The data flow toward devices and applications on the cyber-perimeter is subject to specific privacy policies that anonymize sensitive information before sending them out of the cyber-perimeter, for example, by blurring faces in video streams. To this end, data tainting techniques are also exploited to know beforehand the possible data paths and take actions on those data that are likely bound to leave the cyber-perimeter. As shown in Figure 2 it is possible to label both devices and communications according to the possibility of sending data to external entities.

We easily derive that any device connected to the Internet is potentially an access point of the cyber-perimeter. Thus to have complete control of the Smart Home cyber-perimeter, it is necessary to have a complete control of the Internet-capable devices and the applications running on them. SIFIS-Home aims at empowering the user with complete control of the Smart Home cyber-perimeter by logically bringing any controllable device inside the perimeter. This control is done by verifying the applications installed in the Smart Home, certifying their behaviours and the data usage, and proactively protecting the communication, network and devices. How SIFIS-Home handles these aspects will be discussed in the following sections.

III. Controlling and Certifying App Behavior

Security certification of IoT devices and applications is a crucial element to support the development and deployment of reliable IoT systems and applications [7]. End users need a simple way to assess the level of trust of the applications they install in their Smart Home. To this end, SIFIS-Home provides a labeling schema to describe the potential threats to safety, security and privacy which might stem from misuses of Smart Home devices' APIs. More in detail, the SIFIS-Home framework enforces this kind of assessment where labels are formally assigned to callable APIs after the execution of a certification process which statically analyzes the application source code. By composing the labels related to all the APIs invoked by each application, SIFIS-Home builds an application contract describing quality, trust and potential hazards related to the application usage. In this way, the user has an immediate perception of the level of trust in IoT applications.

The enhanced security and privacy support provided by the SIFIS-Home framework is based on the following key concepts:

- **Certifiable Secure Coding**: the SIFIS-Home framework provides tools and APIs to implement secure and resilient Smart Home applications, also performing privacy-aware data management.
- **Software Verification**: the SIFIS-Home framework provides a methodology and a tool to verify the compliance of applications with user-defined policy, also computing their overall quality, safety, and security

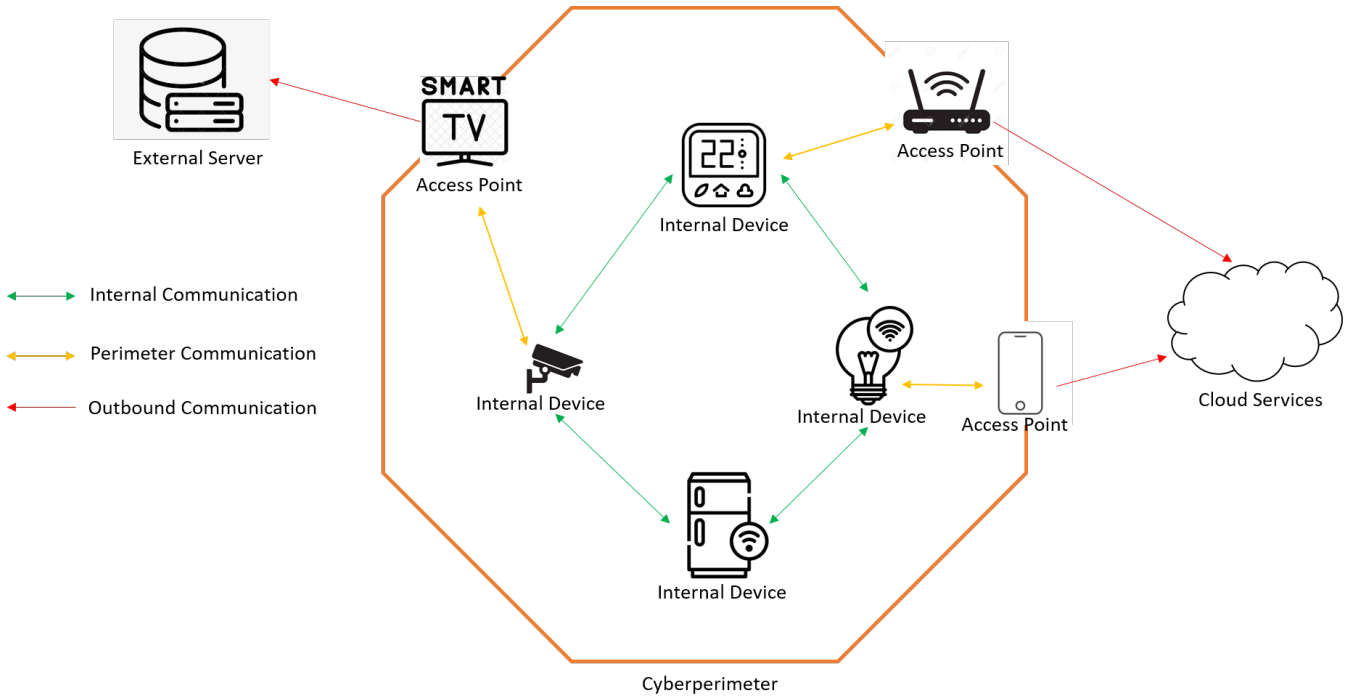

Fig. 2: Logical view of the Smart Home cyber-perimeter

risk, which is represented using labels defining the overall quality and security level of applications.

- **Dynamic Security Enforcement**: the SIFIS-Home framework provides several services aimed at continuously guaranteeing its security, i.e., for protecting the framework from malicious application in order to timely react with proper countermeasures to prevent or mitigate malicious and dangerous effects.

This section describes how the previous key concepts are implemented in the SIFIS-Home framework, while the following section describes the architecture that has been designed to support the dynamic security enforcement in the Smart Home.

A. Certifiable Secure Code

Software Engineering has dealt extensively with finding applicable models to measure the maintainability of software source code during its lifecycle. Through these models, it is possible to measure the source code maintainability after any change to the code, checking whether the maintainability improves or worsens. Measuring and improving code maintainability is very useful for managing *technical debt* (i.e., the cost of additional rework caused by choosing an easy and limited solution instead of using a better approach that would require more work); a definition used to describe all the complications that arise during the development of a software project. Besides, a recent study has shown that analysis and measurement of source code maintainability are still the main methods used for the management of technical debt [8]. Software quality management is becoming a topic of absolute necessity as systems evolve in complexity and size over the years. Using effective programs or tools to maintain them is critical for developers during the software lifecycle. There are several types of tools in the literature that can be used to improve software quality [9]:

- Static Analysis Tools: are useful for examining problems based on code analysis, such as the use of uninitialized variables, the possibility of memory leaks, dereferencing of null pointers;
- UT Tools: allows performing Unit Testing of the source code;
- Memory Hazard Detection Tools: detect possible memory leaks and invalid memory access at runtime;
- Code Browsing/Reverse Engineering Tools: help with code understanding so that improvements and troubleshooting can be applied appropriately;
- Profiling Tools: help understand and monitor performance aspects of the code;
- Coverage Tools: highlight which test cases cover parts of the code run to ensure test quality.

Software Quality is an aspect that has fundamental importance within the SIFIS-Home project, together with Security and Privacy.

B. The SIFIS-Home Development Kit

The SIFIS-Home development kit is a set of software libraries and development tools to develop SIFIS-Home compliant applications. The libraries provide security-aware access to the functionalities of the Smart Home platform by natively including security management functionalities embedded in standard operations for IoT and Smart

Home environments. Two operative blocks compose the development kit:

- SIFIS-Home Development APIs;
- SIFIS-Home ACS (Attestation, Certification and Security Evaluation) Toolkit.

The SIFIS-Home Development APIs support privacy management for secure data storage, innovative IoT specific mechanisms for secure inter-device communication, privacy-aware inter-application data exchange and usage, safe resource management and event logging. The development APIs aim to make the security management transparent to the developer, who will invoke them, reducing the likelihood of making security-relevant mistakes that might introduce vulnerabilities in the Smart Home system. Besides, the APIs will use and handle data analysis service, to interact and exploit data already stored in the interconnected Smart Home system or collected at runtime through sensors. Collected data also include user behaviours, biometric data, voice commands, and the voice assistant's interaction. Thus, the SIFIS-Home Development APIs usage will ensure that managed data will comply with a privacy-preserving approach at all steps of their life span, including collection, use, storage, and transmission. It is worth noting that the labels are assigned to known available APIs through a certification process. A label will thus be assigned to an application depending on the labeled APIs it invokes in its source code. Such label will be used to inform, on one hand, the developer of the risks related to misuse of an API, together with avoiding invocation of the wrong API. On the other hand, the label is directed to the end user, to make them aware of potential risks on privacy (such as data which can be accessed by the developer/service provider), security and safety, making them able to decide beforehand whether to install or not such application in the Smart Home devices.

The SIFIS-Home ACS (Attestation, Certification and Security Evaluation) Toolbox helps the developer to evaluate the quality and the reliability of the produced application as described in Section III. The ACS toolbox returns feedback on the source code to spot code vulnerabilities, data misuse, and poorly handled dangerous operations. Depending on the security and quality of the source code, the ACS toolkit computes an aggregated Reliability Score shown both to the developer and to the end user, who can choose whether to deploy or not an application if the reliability score is too low. The toolbox also performs a match among the invoked APIs and the resource type that they are handling, enabling a mapping to the hazard type that an application or service might bring (e.g., money loss due to energy consumption, privacy-sensitive data access, data transmission outside the Smart Home, physical hazard, etc.). Thus, the ACS toolbox shapes a manifest of the application, reporting all the safety and security-critical operations that the application/service is potentially able to perform. The Manifest is then matched with a user-defined policy to assess if the application is deployable or not in the Smart Home. Manifest and Reliability Score are bundled with the application executable in a single file, whose integrity is protected via digital signature. Hence, in the hypothesis that it is possible to certify the developer's identity, the executable cannot be decoupled from the Reliability Score and Manifest. Depending on the functionality, it is also possible to deploy applications that do not match the policy, monitor their behaviour, and prevent actions that are not in line with the user policy. Unlike other systems like Android, the SIFIS-Home Manifest contains more than the list of critical operations and desired resource access (permissions). The SIFIS-Home Manifest also includes a description of the application behaviour, represented as either a control flow graph or a probabilistic model, which can be matched with a process algebra-based policy describing more complex conditions on the desired behaviour for an application or service.

C. Application Contract

By knowing the APIs invoked by an application, it is possible to infer the offered functionalities, together with the set of resources to which the application accesses. The SIFIS-Home development APIs make explicit the list of critical resources and functionalities accessed by each API. It is performed by labelling the various APIs, with information reporting the potential risks for data privacy or physical security (fire hazard, energy consumption, electrocution risk, etc.), which might stem from misuses of each API call. The labels are thus implemented through JSON-based records, acting as metadata for the various API calls, similar to those presented in [10].

Making explicit the behaviour of an API, knowing the resources it intends to access (access request) is the base element for the generation of an application *contract* [11]. More in general, a contract is the representation of an application behaviour [12] and can be represented using textual documents, markup languages (e.g. the Android Manifest), control flow graphs, formal methods models or formula. In particular, the SIFIS-Home contract brings information about the accessed resources and critical operation, focusing on data privacy, with the contract specifying if processed and generated data are bound to stay in the cyber-perimeter or leave it. Together with giving a representation of the application behaviour, contracts are designed to be matched with usage or security policies, to verify at deploy time if an application is in line with preferences and security regulations provided by the Smart Home administrator. More details on policies will be discussed in Section IV.

IV. Architecture for Security Enforcement

A. Components

A composition makes the SIFIS-Home framework of two device types, which can be present in any number and with different interconnection patterns. Namely, the two device types are Smart Devices and Not So Smart Devices.

- **Smart Devices** have a decent or medium-to-high computational power, one or more connectivity interfaces and can be customized by installing third-party applications. A good example of Smart Devices is Raspberry Pi, Android or iOS devices, including smartphones and tablets, general-purpose embedded systems and even desktop or laptop devices. In the SIFIS-Home framework, Smart Devices are interconnected, using a logical connection pattern in a P2P model.
- **Not So Smart Devices (NSSD)** are generally low powered devices with one or more connectivity interface and they are typically used to read environmental values (i.e., sensors) or to interact actively with the physical environment itself (i.e., actuators). Differently from smart devices, not so smart devices cannot be customized. They generally have simplified operative systems, closed or with limited possibilities of configuration. In particular, it is not possible to install third-party applications on Not So Smart Devices. NSSDs are configured to communicate with a specific smart device, which will send commands to perform actions or read values.

Thus, the SIFIS-Home architecture is made of the interconnection of Smart Devices in a P2P fashion, using mesh-based communication protocols at the data-link level and Distributed Hash Tables (DHT) at the application level, to exchange control messages, enable services and applications intercommunication, transfer and store data, and exploit them to provide services. The distributed network of Smart Devices is the core element of the SIFIS-Home Security Architecture, and intrinsically it ensures an improved architecture resilience since they do not constitute a single point of failure. If a Smart Device fails, i.e., it is corrupted, broken, or switched-off, the remaining Smart Devices are still able to communicate and, thanks to the replication factor introduced by the DHT, the whole system functionality can be maintained until a sufficient number of devices remains active and not corrupted. Smart Devices can be either dedicated embedded systems (e.g. Raspberry Pi, Intel Shield, Mind Cubes), or Smart Home appliances that have connectivity capabilities (Wi-Fi) and have the possibility of installing third-party apps (AndroidTVs, smart fridges, smart stoves, alarm managers, etc.). Tablets and smartphones can also be considered Smart Devices. Also, Smart Devices might have their sensors and actuators to interact with the physical world. However, their functionalities are empowered by connecting with the Not So Smart Devices. The NSSDs, such as smart thermostats, generally expose a limited number of functionalities and, even if they are interconnected, they are generally controlled through other devices (e.g. via a smartphone app). In the SIFIS-Home security architecture, NSSDs are directly connected to one or more Smart Devices, performing requested actions or collecting and sharing sensor data.

Through this architecture, data control is enforced at the application level. In particular, NSSDs are not supposed to connect directly to the Internet. Instead, they are configured to expose a single connection interface only toward one or more Smart Devices. On the other hand, one or more Smart Devices can be connected to the Internet, being thus on the cyber-perimeter. Since every Smart Device is controlled by the SIFIS-Home framework, which also enforces control on the network traffic at a packet level, it is possible to have full control on data crossing the cyber-perimeter. As anticipated, each SIFIS-Home Smart Device runs the SIFIS-Home Security Architecture software, which exploits a set of services for ensuring resilience, security, privacy and safety of the Smart Home users, which will be described in the following.

B. Dynamic Security Enforcement

The SIFIS-Home framework provides several services for guaranteeing its security, taking into account several security aspects. One of the security services provided by the SIFIS-Home framework for Dynamic Security Enforcement concerns **secure, robust and resilient communication**. This service enables exchanging messages over the network secured end-to-end at the application layer, guaranteeing confidentiality, integrity, source authentication and freshness of exchanged network messages. This service primarily supports exchanges of messages in group communication setups that rely on, e.g., the IETF Constrained Application Protocol (CoAP) over IP multicast. Furthermore, this service will enhance the robustness and resilience of the networked system and its communications by providing effective and efficient methods for preventing and dynamically reacting against (Distributed) Denial of Service (DDoS) attacks.

The *System Secure Lifecycle Management* service provides methods, extensions and protocols for handling security-management tasks in the Smart Home system. This will ensure that such tasks are handled and carried out in a correct, efficient and secure way throughout the lifecycle of individual devices and the system as a whole. Particularly relevant tasks related to this service include: securing the bootstrapping, registration and management of (IoT) devices in the Smart Home system; flexible enforcement of fine-grained access control policies in order to grant service and resource utilization to both (IoT) devices and users; establishment, distribution and renewal of security credentials and keying material, in a secure, efficient, scalable, lightweight and authenticated way, as core support to secure communication among (IoT) devices.

The Dynamic Security Enforcement is also provided through the *privacy aware Data Storage and Management* service, which ensures data integrity, confidentiality and availability during the whole data lifecycle, from collection to storage, usage and transmission. In particular, the SIFIS-Home framework stores data in the DHT implemented by the Smart Devices, which ensures a

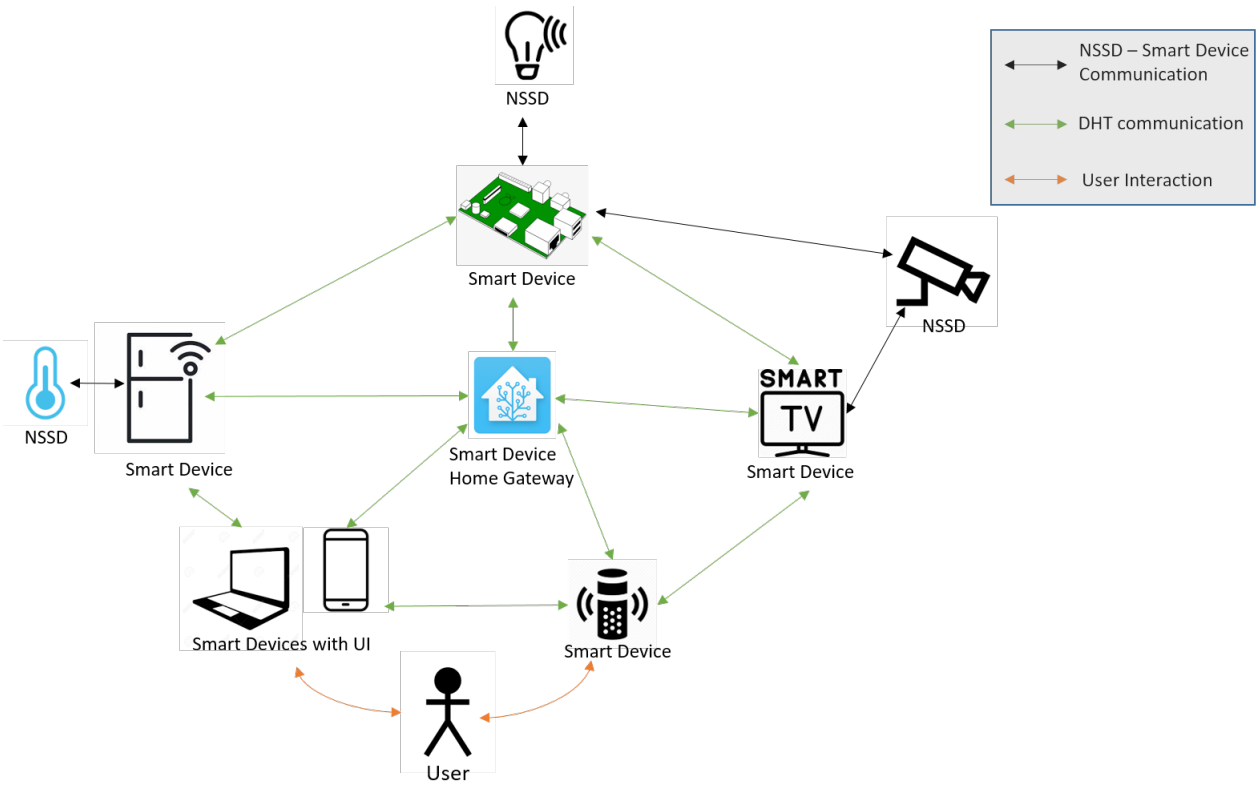

Fig. 3: SIFIS-Home physical architecture

configurable level of replication and distributes the data among the existing devices according to their storage capacities. Security and privacy policies are defined for each data type and, according to such policies, data could be stored encrypted or anonymized. Furthermore, specific authorizations are needed to send data outside of the home cyber-perimeter and, depending on the recipient, data should be anonymized, i.e., stripped of all privacy sensitive information to ensure their privacy.

The *Multi-Level Intrusion Detection* service is the last component of the Dynamic Security Enforcement. This service collects features from several levels of the SIFIS-Home framework, i.e., Kernel, Network, API, DHT, Application and User level, such as:

- number of invoked system calls over time;
- type and number of outgoing/incoming packets;
- number of DHT operations (read/write);
- number of running services;
- presence of active user interaction;
- number and type of user issued commands.

These features are continuously monitored over time, generating multi-level feature vectors, which are constantly fed to a barrier of machine learning-based classifiers trained to discern between standard behaviours and anomalies. If an anomaly is identified, the extracted features are passed to an expert system, which mixes rule-based approaches (heuristics) to classifiers and predictors to automatically understand the specific anomaly type and identify the responsible and the countermeasure to mitigate the anomalous behaviour. By exploiting a multi-level classification approach, it is possible to achieve a global view of the Smart Home system, thus increasing the detection accuracy and specificity in understanding the type of attack and the most effective countermeasure to be taken.

C. Privacy Policies Management

The SIFIS-Home framework enforces control at application and network levels by combining the secure APIs offered to developers of SIFIS-Home-aware applications with proactive and dynamic control daemons working at application and network levels as well. The enforcement behaviour is regulated by user-defined policies. User-defined policies regulate the right to access a specific resource (e.g., data) or functionality from human home users (i.e., tenants, children, guests), devices and applications. To this end, the SIFIS-Home framework includes an engine for the definition and evaluation of Attribute-Based Access Control (ABAC) policies. The engine leverages the Usage Control (UCON) paradigm [13] to define and enforce dynamic policies based on mutable attributes, i.e., access decisions (PERMIT or DENY) are taken by using measures whose value can change over time, thus altering the engine's decision. This dynamicity is a requirement for the Smart Home environment, where the attributes relevant for deciding on an authorization can be physical measures read from devices' sensors, with values changing over time [14] (e.g., room temperature, number of people in a room, services

currently using a data stream, etc.). Through the policy engine, it is possible to specify both general and specific authorizations for data management. To this end, the cyber-perimeter becomes an essential parameter in policy definitions. It becomes possible to specify which data category can be sent out of the cyber-perimeter as-is, which category requires anonymization[4] and which category cannot leave the perimeter. Categories can be based on data type, data labels, application-specific data, or any other identifier characterizing a relevant piece of data on which the administrator might be willing to set up a policy.

By design, policies also act as counterpart for applications contract. The contract, as anticipated, is used to describe the behaviour of an application, while a policy specifies which behaviour is accepted. Thus, it is possible to define usage control policies such as: "It is not allowed to install applications sending video stream data out of the cyber-perimeter ". Hence, by matching the contract of an application with this policy, its enforcement will block possible installation attempts. It is worth recalling that the representation of data flows, useful to model the possibility of data leaving the cyber-perimeter, can be either made explicit by the developer, or they can be extracted afterwards by using data tainting techniques [15].

V. Use Cases and Performance

In this section we report two use cases to present two relevant applications of the SIFIS-Home framework respectively on managing rights of Smart TV applications, and for enforcement of energy saving policies. Some considerations on performance will also be reported.

A. Labeling and Policies

The most common smart devices present in Smart Homes are Smart TVs. Most of the Smart TVs currently on the market have networking capabilities, a large screen, and support the installation of applications. Some of them also have embedded microphones and cameras, while others allow users to install external ones. The primary purpose of network connections in Smart TVs is to connect to a remote content provider service such as Netflix, Amazon Prime Video, Google, and present movies to the users. The camera and the microphone provide data to the smart assistant, replace the remote, and, sometimes, work as access control by recognizing who is sitting in front of the Smart TV.

It is currently reasonably difficult to properly restrict its reach since the operating system is less accessible to the user than other devices due to the DRM requirements. Moreover, its network access requirements make it relatively unwieldy to segregate it into a separate network with no access to the internet.

[4]Obligations can be used to provide authorizations under conditions.

The SIFIS-Home approach to make the system more trustworthy requires proper labelling at the application level.

Thus, the privacy risks would be apparent to the user buying the device:

- A camera and a microphone may record the environment
- The network connectivity may let a third party access such recordings

The user would know beforehand what remote resources a SIFIS-compliant application would need to access and for which purpose.

- The user would be aware of the risks clearly and succinctly, with the option to further detail what is doing what, down to the single resource access. For example, the Agent application would need to access the camera and microphone and potentially send data to a remote node for further computation.
- The user may allow only the resource access he deems acceptable, clarifying the trade-off between privacy, functionality and comfort. E.g., the playback application might want to access the camera to pause the playback optionally and automatically, but it is not required for its core functionality.
- By having a list of network resources, it is easier to detect earlier rogue behaviour and prevent it accordingly.

Listing 1 shows a possible SIFIS-Home manifest file for the *Player* application running on the Smart TV, focusing on the privacy risk related to camera, microphone and for the network access.

Listing 1: Example of manifest.json

```
{
    "name": "Player",
    "capabilities": {
        "devices": {
            "camera": {
                "path": "/dev/video*",
                "risk_type": "privacy",
                "risk_level": "medium",
                "info": "Gesture␣tracker"
            },
            "microphone": {
                "path": "/dev/snd/*"
                "risk_type": "privacy",
                "risk_level": "high",
                "info": "Voice␣Agent"
            }
        },
        "network_access": [{
            "url": "https://mediaserver.lan",
            "risk_type": "privacy",
            "risk_level": "low",
            "info": "Default␣local␣media␣server"
        }, {
            "url":
                ↪ "https://privacy.invasion.com",
            "risk_type": privacy,
            "risk_level": "high"
```

```
            "info": "Share your preference with
                ↪ EPI"
        }],
    },
}
```

Listing 2 shows an example of access control policy including four rules, written in a human readable language, which have been derived from the information taken from the manifest of the Player application shown in listing 1.

Listing 2: Human readable example of access control policy rules

```
{
  ruleid = "disable the camera",
  effect = Deny,
  target:
    subject.name = "Player",
    resource.name = "camera",
  conditions:
    resource.path = "/dev/video*",
},
{
  ruleid = "disable the mic",
  effect = Deny,
  target:
    subject.name = "Player",
    resource.name = "microphone",
  conditions:
    resource.path = "/dev/snd*",
},
{
  ruleid = "forbidden network
            connections",
  effect = Deny,
  target:
    subject.name = "Player",
    resource.name = "network",
  conditions:
    resource.protocol = "https",
    resource.address = "privacy.invasion.com"
},
{
  ruleid = "allowed network
            connections",
  effect = Permit,
  target:
    subject.name = "Player",
    resource.name = "network",
  conditions:
    resource.protocol = "https",
    resource.address = "mediaserver.lan"
}
```

The first three rules of the policy prevent the subject specified in the field subject.name, i.e., the application "Player", from using the resources "camera" and "microphone" and to open network connections with the site "privacy.invasion.com" because the risk levels paired to such resources in the manifest file is not low. The forth rule, instead, allows the aforementioned application to use the network resource when the protocol is "http" and the site address is "privacy.invasion.com", because the risk paired with this network connection in the manifest file is low.

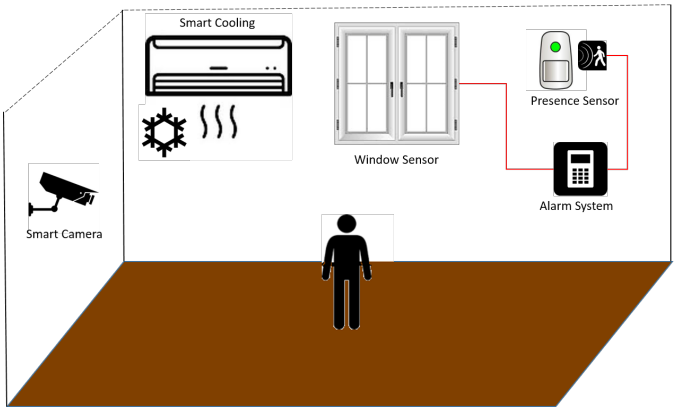

Fig. 4: Deployment example for energy saving.

B. Deployment and Usage Policies

Figure 4 provides an example of deployment of the SIFIS-Home framework for enforcing an energy saving policy.

In this example we consider as Smart Devices the smart cooling manager and the alarm system of the house, while we consider as NSSD the smart camera, the air conditioning split, the presence sensor and the window sensor. The envisioned system is able to implement, among others, the following usage policy for energy saving: “The air-cooling can be switched on if there are people in the room and there are no open windows”. Though simple, the example is interesting as it shows how decisions can be taken by a Smart Device by evaluating conditions based on attributes observable by other devices. The presence of people in the room can be observed either through the camera or the presence sensor, which are both NSSD connected to the Smart Device alarm system. The windows sensor is connected to the Smart Device alarm system as well. In this setting, the policy evaluation can be performed on the Smart Device alarm system, while the access decision enforcement is performed by the smart cooling manager through the air conditioning split.

Concerning system performance, the evaluation of a policy using a well known access control engine[5] requires a time which depends on the computational capability of the Smart Device where the evaluation is performed, on the number of attributes taken into account in the policy, and on the time required to collect those attributes in case they must be retrieved from other smart devices. A first set of experiments has been performed, where the devices have been emulated through Raspberry-PI3s,

[5]https://github.com/wso2/balana

interconnected through WiFi using the AODV[6] routing protocol. In particular, in the case of the previous policy implemented using 2 attributes local to the evaluating smart device (i.e., the boolean values provided by the presence sensor and by the window sensor), the evaluation time is about 252 ms [7]. More complete experiments will be performed at later stage of the project and are in the scope of future work. Though we are aware of other solutions able to implement the presented policy, it is worth noting that SIFIS-Home brings the following advantages: (i) the policy is not hard-coded in a home-automation system, instead it is fully configurable, able to express complex conditions and based on a human readable language; (ii) the framework is able to operate in an heterogeneous environment, with devices of different brands with a fully software integration; (iii)the analysis is not performed in the cloud but locally, hence without having potentially sensitive data to leave the cyber-perimeter.

VI. Conclusions

An increasing number of services is offered to the Smart Home tenant by the third-party global providers, who process user data and commands in the cloud. For this reason, data privacy is a relevant concern brought about by the globalized Smart Home paradigm. The SIFIS-Home project proposes an alternate paradigm centred on giving back control to the Smart Home user by capillary enforcement of security and privacy policies. The *SIFIS-Home framework* enables a paradigm that is still compliant with the globalized Smart Home model. It still leaves the user the possibility of choosing which data can be shared out of the home cyber-perimeter and how they are shared to avoid disclosing private information. On the other hand, the *SIFIS-Home* architecture implements security-by-design and proactive security mechanisms to protect devices, users and data from malicious applications and other intrusion attempts.

Though the principles of SIFIS-Home can be adapted to other IoT environments, the SIFIS-Home framework has been designed specifically to match the requirements of a Smart Home environment. The SIFIS-Home framework relies on the concept of a P2P architecture with devices part of the same Wi-Fi network and limited mobility. Also, SIFIS-Home is based on a user-centric paradigm, where the user defines the usage, security and safety policies for his own environment.

This paper gave a high-level view of the SIFIS-Home framework and architecture, presenting the involved actors and discussing a use case. In-depth technical discussions of the technological enablers, algorithms, protocols and specific implementations are material for ongoing and future research work in the scope of the SIFIS-Home research project.

[6]https://bit.ly/3mbfi6I

[7]for a more complete evaluation please refer to [14]

References

[1] G. Mantas, D. Lymberopoulos, and N. Komninos, "Security in smart home environment," in *Wireless Technologies for Ambient Assisted Living and Healthcare: Systems and Applications*, A. Lazakidou, K. Siassiakos, and K. Ioannou, Eds. Hershey, Pennsylvania, USA: IGI Global, 2011, ch. 10, pp. 170–191.

[2] W. Li, T. Yigitcanlar, I. Erol, and A. Liu, "Motivations, barriers and risks of smart home adoption: From systematic literature review to conceptual framework," *Energy Research & Social Science*, vol. 80, p. 102211, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214629621003042

[3] H. Lin and N. W. Bergmann, "Iot privacy and security challenges for smart home environments," *Information*, vol. 7, no. 3, 2016. [Online]. Available: https://www.mdpi.com/2078-2489/7/3/44

[4] H. Touqeer, S. Zaman, R. Amin, M. Hussain, F. Al-Turjman, and M. Bilal, "Smart home security: challenges, issues and solutions at different iot layers," *The Journal of Supercomputing*, 2021. [Online]. Available: https://doi.org/10.1007/s11227-021-03825-1

[5] J. Varghese and T. Hayajneh, "A framework to identify security and privacy issues of smart home devices," in *2018 9th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, 2018, pp. 135–143.

[6] F. Hussain and M. Qi, "Integrated privacy preserving framework for smart home," in *2018 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, 2018, pp. 1246–1253.

[7] G. Baldini, A. Skarmeta, E. Fourneret, R. Neisse, B. Legeard, and F. Le Gall, "Security certification and labelling in internet of things," in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, 2016, pp. 627–632.

[8] N. A. Ernst, S. Bellomo, I. Ozkaya, R. L. Nord, and I. Gorton, "Measure it? manage it? ignore it? software practitioners and technical debt," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ser. ESEC/FSE 2015. Association for Computing Machinery, 2015, pp. 50—60.

[9] R. Krishnan, S. M. Krishna, and N. Bharill, "Code quality tools: Learning from our experience," *SIGSOFT Softw. Eng. Notes*, vol. 32, no. 4, p. 5–es, 2007.

[10] P. E. Naeini, Y. Agarwal, L. F. Cranor, and H. Hibshi, "Ask the experts: What should be on an iot privacy and security label?" in *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 2020, pp. 447–464. [Online]. Available: https://doi.org/10.1109/SP40000.2020.00043

[11] N. Dragoni and F. Massacci, "Security-by-contract for web services," in *Proceedings of the 4th ACM Workshop On Secure Web Services, SWS 2007, Fairfax, VA, USA, November 2, 2007*, 2007, pp. 90–98. [Online]. Available: https://doi.org/10.1145/1314418.1314433

[12] A. Aldini, A. L. Marra, F. Martinelli, and A. Saracino, "Ask a(n)droid to tell you the odds: probabilistic security-by-contract for mobile devices," *Soft Comput.*, vol. 25, no. 3, pp. 2295–2314, 2021. [Online]. Available: https://doi.org/10.1007/s00500-020-05299-4

[13] J. Park and R. S. Sandhu, "The ucon_{abc} usage control model," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 1, pp. 128–174, 2004. [Online]. Available: https://doi.org/10.1145/984334.984339

[14] A. L. Marra, F. Martinelli, P. Mori, and A. Saracino, "Implementing usage control in internet of things: A smart home use case," in *2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, August 1-4, 2017*. IEEE Computer Society, 2017, pp. 1056–1063. [Online]. Available: https://doi.org/10.1109/Trustcom/BigDataSE/ICESS.2017.352

[15] J. Newsome and D. X. Song, "Dynamic taint analysis for automatic detection, analysis, and signaturegeneration of exploits on commodity software," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2005, San Diego, California, USA*. The Internet Society, 2005. [Online]. Available: https://www.ndss-symposium.org/ndss2005/dynamic-taint-analysis-automatic-detection-analysis-and-signaturegeneration-exploits-commodity/

Luca Ardito is an Assistant Professor at the Department of Control and Computer Engineering at Politecnico di Torino. His current research interests are mobile development and testing, green software, new programming language analysis, IoT, and empirical software engineering methodologies.

Luca Barbato is the founder of Luminem and contributes to multiple high profile open-source projects, like Gentoo, VideoLan, and FFmpeg. He is now involved in the development of rust-based multimedia components, among those rav1e, a fast AV1 encoder.

Paolo Mori is a researcher at IIT-CNR. His research interests include trust, security and privacy in distributed systems and IoT, focusing on access/usage control, and Blockchain technology.

Andrea Saracino is a researcher at IIT-CNR. His research interests include usage control, IoT, distributed system security, and mobile malware analysis.