








Permissioning and Personal Data Management in Alternate-Tenancy Smart Environments

Catarina Silva^{*}, Vitor Jesus^{†}, João P. Barraca^{*}, Antonio Nehme^{†},
Gilad Rosner^{§}, Mohammad Waqas^{¶}, Rui L. Aguiar^{*}

^{*} Instituto de Telecomunicações, Universidade de Aveiro, Aveiro, Portugal

[†] Aston University, Birmingham, UK, [‡] Birmingham City University, Birmingham, UK,

[§] Internet of Things Privacy Forum, UK, [¶] Privdash Ltd, UK

Abstract—We focus on a scenario that we call *SmartBnB*: smart homes rich in devices that are rented for a short period of time such as holidays. Such frequent rotation of different users raises two types of mutually dependent challenges. First, we have a *permissioning* challenge since devices are commonly designed with a single user in mind; yet, given our shared scenario, we need to grant the necessary, yet minimum, access to devices and just for the duration of the stay. Secondly, *privacy* challenges arise given the shared and externally managed nature of the devices (e.g., on the cloud). Other challenges exist such as the lack of user-interfaces that make difficult to serve a privacy notice.

This paper discusses how such a familiar scenario can bring stringent requirements. After characterising and discussing the emergent challenges, we present a conservative and bottom-up architecture (e.g., by reusing existing protocols) that we evaluate on a realistic scenario. We conclude that, while it is able to meet current functional requirements, a *SmartBnB* scenario raises technical gaps.¹

Index Terms—Privacy, smart devices, Internet of Things, Sensors, Transparency, Control, Consent Receipts

I. INTRODUCTION

We discuss a scenario that we term *SmartBnB*. Similar to the familiar online platform that allow booking of rooms and whole properties for holidays (e.g., AirBnB), a *SmartBnB* is a space that people can rent for a period of time and that takes full advantage of the Internet-of-Things (IoT) and Smart Homes paradigms. Such a smart space can dramatically increase the quality of the experience of both guests and hosts, while creating new ecosystems and business models. It follows the increasing commoditization of smart technology and communications. The key differentiating factor in *SmartBnB* is that the devices are shared over time. Whereas the platform, devices, the host, etc., are fairly static over time, guests come and go over short periods of time (e.g., over one single weekend). What was previously a space consisting of “dumb” and “single-feature” devices, we see now a proliferation of cheap internet-enabled devices that go far beyond the delivery of the basic functionality to deliver added-value services. For example, a smart light bulb can have different modes of operation, be controlled remotely with a mobile app, and be scheduled based on the time of the day. Whereas a simple lightbulb, by itself, raises (relatively) little risk for a temporary

stay, a smart lock recording habits and status on a front door or an indoor camera are much more problematic.

A *SmartBnB* is thus a problem of dynamic, in alternation, sharing of devices within a smart space. We are interested in two primary aspects. First, there is a dynamic process of selective delegation of access grants and permissions that we call *permissioning*; second, it is expected that most devices will collect some form of personal data thus raising privacy and data protection considerations. The work was done under Project CASSIOPEIA (Contextually-Appropriate Selective Sharing IoT Open-standard PErmissioning Architectures)² within a techno legal team. The baseline setting is a smart home containing a range of common devices for sensing and automation purposes: thermostats, video-enabled door locks, entertainment systems, robot vacuum cleaners, environmental sensors, security cameras, etc.

The core challenges in the *SmartBnB* scenario are two-fold. From an *Access Control, or Permissioning, perspective*, we (1) deal with how to give temporary access to the required devices for a guest, and, for a pre-defined period of time, (2) how to retain scoped access to devices and how to manage access delegation from a single, user-friendly control point. From a *privacy and data protection perspective*, other challenges exist such as how to isolate the personal data generated by all participants or how to comply with Data Protection regulations; furthermore, we look at how to assure users remain in control of the data collected. For example, in terms of Data Protection, a number of questions are unclear as the typical roles (e.g., Data Controller and Processor in EU’s GDPR) became unclear.

The project adopts a user-centric and use-case design approach. Furthermore, and as a strict requirement for our proof-of-concept, we used real-world, commercial devices, mature protocols, and open source software to build our testbed. This requirement was strictly enforced so we could test how far current technologies and products could support *SmartBnBs*. The project contemplates the full renting lifecycle: booking a property, accepting all privacy notices for the devices, controlling and accessing devices, checking-out, and requesting deletion of personal data. To the best of our knowledge, this specific angle to smart homes is novel.

¹The first three authors contributed equally as first authors. Email addresses: {c.alexandracorreia, jpbarraca, ruilaa}@ua.pt, v.jesus@aston.ac.uk, antonio.Nehme@bcu.ac.uk, gilad@iotprivacyforum.org, waqas@privdash.com

²Funded by EU Horizon 2020, NGI grants.

This paper is structured as follows. In Section II we present and explore the SmartBnB scenario and show the non-trivial challenges that a smart space with alternating tenants raise – technical, implementation, societal and regulator. In Section III we review related work but note that more work is needed across the different disciplines such as Law and Technology. In Section IV-B we present how we envision the user journey and interfaces: from booking a property to accepting Privacy Notices per device to managing thier personal data post-stay. In Section IV we propose an architecture that implements the user journey addresses while mitigation gaps we identified and noting limitations. As a design principle, we reuse as much as possible common off-the-shelf technologies. An implementation of this architecture on a testbed is then evaluated in Section V. From the design and implementation lessons, we raise gaps in Section VI which will also draw our final conclusions and identify future work. All the code used in this project is open-source and publicly available³.

II. THE SMARTBNB SCENARIO

Figure 1 shows our primary use-case. It reflects the familiar experience of booking a property for a short duration with the difference that there are a number of devices guests need access to and that there are data protection and privacy requirements. Ahead of the stay and soon after the end of the previous stay, the *Host* will roll-over any current access delegations and, connected, manage any personal data. Once

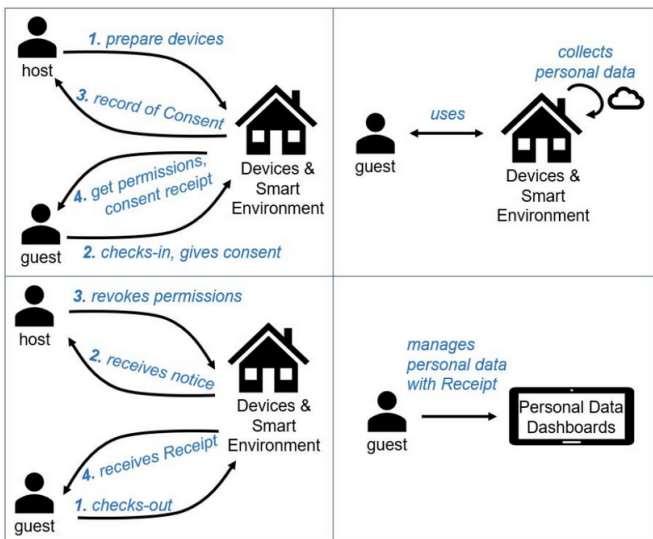


Fig. 1. SmartBnB use-case.

devices have been reset to a default state, the *Guest* checks-in. A crucial point is connected to Privacy for two reasons. First, the Guest should review the Privacy Notices of all devices – noting that there are many devices making this exercise time consuming. Secondly, guests must give consent for collection of personal data. The Host records consent against each device

and the overall SmartBnB. Only now the Host is able to allow Guests to use the devices. We further note that there are two broad types of devices depending on its role: (1) devices that are under full (temporary) control of the Guest and the Host cannot access (excluding emergencies), and (ii) devices that the Guest cannot access or has only limited permissions such as fire alarms.

The project also integrates *Consent Receipts* with extensions: upon giving consent, per device and against a particular Privacy Notice, the Guest receives a set of receipts which should be preserved as they will potentially be used later to request (e.g.) erasure of personal data – see Figure 3. CASSIOPEIA designed the Consent Receipt as an active digital object: the receipt is no more than a javascript object with authenticated data with a button to issue requests.

A. Challenges and Open Questions

Regardless of how familiar a SmartBnB scenario feels, a number of open challenges exist.

1) *multi-stakeholder*: Part of the complexity of this scenario stems from the intrinsic diversity of stakeholders which create a complex environment of access delegation and privacy. Direct stakeholders are the *property owner* (Host) and the renters (Guests). We then have the *Platform* where the booking happens and is, usually, a third-party. Note that, while the Guest can have full access and use of the devices, the Host will still have, very likely, administrative rights that may collide with Privacy directives. For example, there must always be an overriding administrative access for safety reasons (e.g., a fire emergency).

Another critical stakeholder is the *Device Manufacturer* which usually involves a *cloud provider*, often attached to a subscription that collects personal data – e.g., an indoor camera has an app to access the video stream remotely.

Other stakeholders we envision in our SmartBnB scenario are Authorities (Law Enforcement, Data Protection authorities, etc.) or sub-Actors, such as parents giving access to children.

2) *Integration of Devices and Identities*: IoT is notoriously difficult to integrate as there is no agreed overall standard, with solutions relying on Zigbee Home Automation, or custom protocols over WiFi. Existing standards such as OneM2M are uncommon, and custom solutions proliferate. ONVIF may be an exception for security cameras. However, low end devices tend to use custom solutions. MQTT provides the means for integration, but still not common in home products as commercial solutions seem to prefer the use of custom protocols. As such, integration frequently uses large cloud providers such as Google or Amazon with full stacks covering devices to services. The net result is a fragmented landscape with islands of compatibility around vendor infrastructure. In this view, several initiatives exist to add a shim integration layer – e.g., Domoticz, Home Assistant, SmartThings, OpenHab or Homekit. Integrations are made in an ad-hoc manner or as agreements are established.

3) *Identity and User Management in IoT*: The notion of identity in home IoT is usually weak, as often simply centred

³<https://github.com/catarinaacsilva/cassiopeia>

on a hub and mobile application supporting a single user. More than functional, this sits at the core of our alternating, multi-tenant SmartBnB. Data collected is often managed at device level, far unsuitable for multi-tenancy. Related, delegation of access is equally limited.

4) *Consent*: Consent, as modelled in EU’s GDPR, is only valid if explicit, informed, freely-given, unambiguous and specific [1]. For example, the data subject must be notified about the controller’s identity, what kind of personal data will be processed, how it will be used, or the purpose of the processing operations – among other requirements. When sharing devices, Consent gains an unexpected level of complexity. For example, devices need now to support multi-tenancy which, as said, is not the normal case.

Our project uses *Consent Receipts* [2] to this aim. A Consent Receipt is an artifact that records a data protection transaction similar to a conventional shopping receipt. If designed with strong auditability properties [1], it can empower individuals, make organisations compliant and more ethical, and allows authorities and watchdogs to effortlessly monitor the market and resolve disputes. The challenge is to embody auditability in the objects. Perhaps the starting point is that they need to be standardised – e.g., with the upcoming ISO/IEC 27560 [3].

5) *Regulatory questions*: SmartBnB raises important questions about privacy and data protection for the guest, including obtaining valid consent to store and process personal data when roles are either unclear or new [4]. When a Host buys a smart device and first sets up, they must give consent to the device manufacturer to collect personal data. However, they are not, strictly speaking, the Data Subjects and there is, thus, an implicit delegation of consent which most data protection laws forbid. Our project addresses this issue by obtaining explicit consent from a guest on behalf of device manufacturers prior to the guest arriving at a property. Nevertheless, unclear legal areas are equally raised.

6) *Device Integration and Hardware Access*: As discussed later, we use Home Assistant as the integration platform. One challenge we faced was to guarantee that, during the stay, the Host does not access devices during a stay. This is a limitation across common integration platform. To mitigate this problem, we embedded access tokens to every call to the platform. We note this is a weak approach that due to the lack of native access control. Access sub-delegation the delegation between the primary Guest and sub-Guests (e.g., from parents to children), is a further challenge. A similar problem consists of direct access to the hardware, often enabling a user to completely bypass a device (e.g., by resetting it) which can only be solved by integrating hardware security [5], very rarely seen in hom IoT.

III. RELATED WORK

In this section we review related work from the perspectives of data protection, privacy and consent, and the legal aspects in IoT and Smart Homes.

A. Permissions and Access Control in IoT

Access control mechanisms can monitor the access events to resources and restricting only to authorised users [6] depending on a set of pre-defined policies, particularly useful when regulatory questions are important [7]. The notion of role is also key, and supports a notion of data lifecycle.

Internet of Things (IoT) access control solutions can be roughly classified as centralized, hierarchical, federated or distributed [8]. More recently, decentralised approaches have been developed such as Distributed Ledgers Technologies (DLT) [9]–[12] or certificate-based ones [13] such as Lightweight Access Control and Key Agreement Scheme for IoT devices (LACKA-IoT) [14], [15].

Access delegation [16] is more challenging. Traditional access control models, such as Access Control Lists (ACL), Role-based Access Control (RBAC), or Attribute-based Access Control (ABAC), are not particularly fit for IoT environments, due to lack of centralised administration or simply complexity and unavailability of enforcement modules in constrained operating systems. Delegation architectures for IoT have, nevertheless, been proposed [17]. BlendCAC [18], in particular, was proposed as decentralized, federated capability-based access control mechanism to enable effective protection for devices, services and information in large-scale IoT systems. DLTs have also been considered for delegation [19].

B. Consent and Legal Aspects in IoT and Smart Homes

User-centric, intuitive architectures so data subjects control consents is a current research problem; see, e.g., ADVOCATE [20]. Specifically for IoT and Smart homes, consenting is an open topic [21] from which the usability/transparency problem is perhaps the most difficult to tackle [22] – e.g., there may not be a screen to show a privacy notice. It can be argued that, past the early stages of setup and collection of personal data, the problem is not significantly different, or more complex, than other scenarios such as the web [23] or mobile phones. Serving notices and seeking valid consent [24], [25] is thus the most challenging scenario, along with the distributed nature of the scenario, with numerous devices (and points of collection) available.

IV. ARCHITECTURAL APPROACH

This section presents our proposed architecture. We start by presenting the integration platform, taking a user-centric perspective by showing how a Guest navigates the permissioning and privacy controls, and then we elaborate on the technical architecture.

A. The Integration Platform

To facilitate a unified control platform for smart devices, we selected Home Assistant⁴ as it is free, open-source, extensible, actively developed, and supports a large number of devices. Devices, services and IoT technologies are supported using a range of protocols such as Queuing Telemetry Transport

⁴<https://www.home-assistant.io>

Invitations

Hello, Antonio

Welcome to **123, Road st, Shireham**

Ahead of your stay, please review

- The devices you will have access
- Each privacy notice

Period of stay: 14-May-2022, 11:30am to 18-May-2022, 10:00am

Lights

Device	Location	Privacy Notice
Light	Hall	View Consent
Light	Kitchen	View Consent
Light	Bedroom 1	View Consent

Fig. 2. Accepting Privacy Notices for each device.

(MQTT), Bluetooth, or ZigBee. The front-end dashboard ("Lovelace") offers different predefined layouts to display information and control devices, and has native web and mobile support. The interface is fully customizable using the integrated editor or by modifying the underlying configuration code (in YAML). The baseline goal of Home Assistant is to act as a central smart home controller hub. The provided rule-based system for automations allows creating custom routines based on a trigger event, conditions and actions, including scripts, which can be used programmatically. Community contributed add-ons get a security rating based on their access to system resources.

B. User Journey

The user interface is seen as critical as explained. This section focus on the user journey aspect and it will be complemented in Section V (Evaluation).

On logging in, and ahead of the stay, the first key action concerns accepting the Privacy Notices of all the devices involved. This is shown in Figure 2.

CASSIOPEIA	
active	erasure confirmed
When	08/05/2021 13:09:12
Who	NGI CASSIOPEIA Demonstrator
What	SmartBnB Device 7
Privacy Notice	at the time of the receipt (we keep copies)
Other Information	
request erasure	

Fig. 3. Consent receipt

At this point, it should be clear how cumbersome the process is: the user should open all Privacy Notices and accept all or some. For each Notice accepted a Consent Receipt is sent (Figure 3). After the stay, the user can utilise the Receipt to, among other actions, request deletion of personal data. A copy of each receipt is sent to the Data Controller, who often is the manufacturer of the device.



Fig. 4. Control panel for the user (floor plant view).

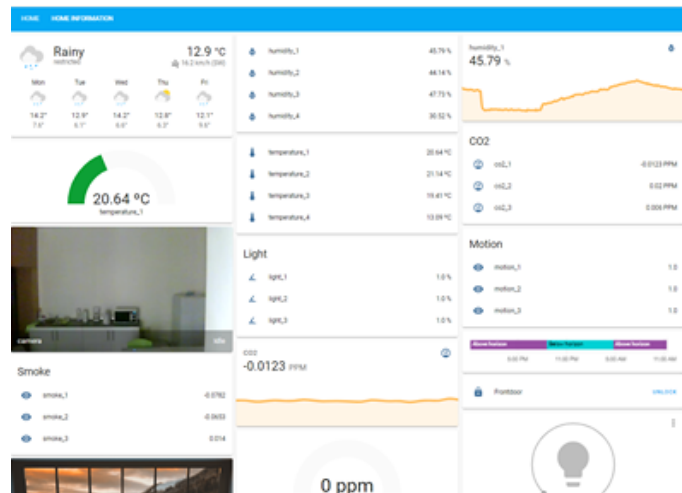


Fig. 5. Control panel for the user.

When the Renter checks-in to the property, the platform is ready to use and screens similar to Figures 4 (plant floor) and 5 (control panel) become available. The figures are taken from our testbed. Only devices whose associated Privacy Notices were accepted will be shown which ties IoT to legal requirements. On checking-out, permissions are revoked; the user can then request the erasure of their data through their consent receipt (Figure 3).

C. Architecture

Our proposed architecture has three different modules (Figure 6): Data Manager Service (DMS), Privacy Manager Service (PMS) and Receipt Manager Service (RMS). Home Assistant is used as the middleware and unified control platform. The user is in control of the complete data flow from moment of consent. At the initial stage, the Guest interacts with PMS to set up the stay. This service is responsible for requesting a consent receipt (see Figure 3) and provide it to the user (sent by email in the form of a link). The receipt contains all relevant information about the consent and its integrity is ensured by signing it and returning to PMS, which then forwards to the RMS and is stored.

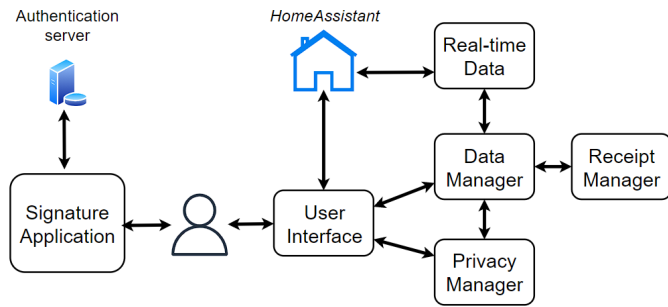


Fig. 6. Architecture.

The PMS is primarily a local service that is deployed on premises (ehte Smart Home) and is controlled by the Host. As a module, it has no (direct) access to Guest’s data but it is responsible for the initial setup and for communicating with the remaining components. In contrast, the DMS is a remote service that allows the Guest to have control over his personal data independently from the physical location. DMS manages the Guest’s personal data, supervises the first interactions (creating the stay), interacts with the RMS and provides the ability to the Host to decide to remove or anonymize the Guest’s personal data, should the Guest so request. Moreover, this component is responsible for interacting with the *Signature Application*.

In terms of implementation, we chose PostgreSQL⁵ as the relational database, InfluxDB⁶ for the time series database for IoT data, a Cassandra⁷ for the Distributed Hash Table (DHT) database (for scalability) which also stores consent receipts.

The three main services (Privacy, Data and Receipt Management) communicate over a REST API implemented in Python using the Django framework; the *Signature Application* was developed using Flask⁸ framework.

1) *Privacy Manager*: The PMS handles creation of new stays which is associated with a new guest; the email address is the identity. The PMS provides information about the dates of the stay, check-in and check-out dates (see Figure 2), as explained.

The final step of preparing the stay is the receipt signature which works as a proof of consent. The system creates a new receipt which the guest must sign and return. As a realistic demonstration and proof-of-concept, the RMS validates the signature using the certificate of the Portuguese Citizen Card, a smart card.

2) *Data Manager*: A key aim of CASSIOPEIA is to enhance privacy controls directly to Guests. DMS is the key module by offering features such as data transparency (by offering detailed reports of data flows), data portability (e.g., by exporting data in CSV format), anonymisation (in case of data retention); DMS further offers data isolation between the roles of Guest and Host. For example, the Host cannot

change the information displayed to the Guest. In addition, DMS directly manages raw data from the IoT platform helping to mitigate a malicious Host.

At the end of the stay, the Guest still has access to DMS to request data deletion, pending a decision from the Host (e.g., data needs to be kept for regulatory purposes). In both cases, the Guest is notified and can check using their own means the status of their personal data.

3) *Receipt Manager*: As mentioned, the RMS has the ability to generate and store signed receipts which is proof of consent. Since the receipt contains information that can be used to infer sensitive data about the guest, we implemented the following:

- The receipt should be sent to the service that requested it and should only be stored after the signature process;
- The identity of the data subject should be validated to ensure its integrity;
- The receipt should be signed and stored in a distributed way using a DHT structure to ensure availability.

The key design principle is that the service that requests the receipt should be the one providing the information to populate the fields in the receipt structure. In practice, the PMS should invoke the method to generate a new receipt (as a read-only GET request), and passing Host-populated receipt fields such as version, manufacturer, privacy notices, devices, etc. Receipts fields that are generic (e.g., language and jurisdiction) can have default values. The consent field should also be supplied but it only has two possible values: consent or not consent. The timestamp field uses the current time of the receipt generation using a trusted software library, thus offering some assurance. The receipt ID field is unique to high likelihood (UUID libraries).

Since Privacy notices are key to (in GDPR) meet the requirement of informed consent, it is integrity-protected so that no modifications post-consent are undetected. Since it is associated with a device, the receipt has a field "device" with a dictionary where the key is the name of the devices available at the smart home and the value is the privacy notice; a field "entity" with a dictionary where the key is the name of the entity and the value is the notice. In this way, we can decouple "entities" and "device" from privacy notices.

To protect the integrity of the whole receipt, a fingerprint of the receipt is created with SHA-256 and is stored in the *receiptFingerprint*. The Guest associates their identity to the receipt by signing the receipt with the *Signature Application* (e.g., using a smart card). The DMS only stores a reference to the receipt. The RMS stores the receipt and responds to all the requests by other services involving the receipts. Moreover, this service can verify the validity of the signature, which is verified by the PMS before storing in the RMS. At any moment, it is possible to request a verification using the signature validator in the RMS.

D. Home Assistant integration

The integration of the smart devices with Home Assistant is straightforward in the sense of following any other integration

⁵<https://www.postgresql.org/>

⁶<https://www.influxdata.com/>

⁷<https://cassandra.apache.org/>

⁸<https://flask.palletsprojects.com/en/2.0.x/>

with an IoT environment. However, CASSIOPEIA has further requirements that required custom extensions.

As previously mentioned, it is important to guarantee data isolation between the Host and the Guest: the Host should not access data about the Guest during the stay. Home Assistant does not provide any method to restrict the views depending on user roles. Our approach, rather pragmatic, was to create a new user group in *homeassistant/.storage/auth*.

This approach is not ideal and raises the limitations of existing platforms when handling multi-tenancy and roles. In particular, every time a new device was added the whole Home Assistant server, and all the modules we developed, had to be cold restarted so to load the new configuration. Our workaround, however, was functional.

We implemented three different roles. An Administrator can see all the devices and update the configurations; during Guest stays, Hosts can only access the disabled devices (not in use by the Guest); after the Guest leaves, the Host can access all available devices in the smart home again. Furthermore, during a stay, the Guest has access only to the consented smart devices. Afterwards, all Guest permissions are revoked automatically. The content for each role is configured in a *homeassistant/.storage/auth* file.

Both Guest and Host cannot change the configurations of Home Assistant by programatically removing a button. This workaround measure is only a soft control, ammountable to obfuscation, and is a further example of the limitations of current Smart Home platforms for multi-tenanted environments.

V. EVALUATION

In this section we show evaluation results of our implementation. Our testbed implemented the decribed services as follows:

- the PMS and the DMS are in the same virtual machine but in different virtual environments using Docker-Compose, emulating the fact that the PMS is deployed locally at the smart home and the DMS is publically available as a remote service;
- Home Assistant and the InfluxDB are in the same virtual machine;
- the RMS and the Cassandra database are in the same virtual machine due to the high required computational capacity to run the Cassandra database; and
- the PostgreSQL runs in the same virtual machine than the PMS and the DMS with two different instances for each service. The prototype setup assumes that the *Signature Application* runs on the user device to sign the receipt.

We report on two aspects: the user interface, complementary to Section IV-B, and the functional properties.

A. User interface

Whereas we previously described the user journey before, we detail now the process of signing a receipt at the end of the creation of the stay. The user identification (in this case, the user's email) is associated to a new stay and new consent receipt is generated and sent to the user (Figure 7). The final

step is to sign the receipt, by the user, to validate the receipt as proof of the given consent. DMS provides the methods to

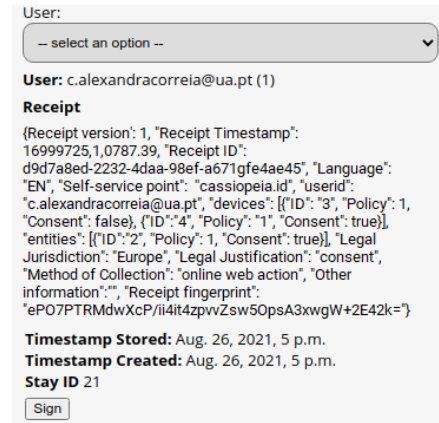


Fig. 7. Signing a new receipt.

user access to the application and check all stay and consent information such as receipt data containing the privacy notices, list of entities and consented devices, or dates of stay and stay identification (Figure 8).



Fig. 8. Retrieving a stay from the DMS.

B. Functionality

The exchanged messages between the different services to create a new stay are depicted in Figure 9. The Client is the device accessing the service; in this case, it is the Owner's device such as a browser. Initially, the Client requests the registration of a new stay. The PMS request a new Receipt from the RMS. After creating the Receipt, it is stored temporarily in the PMS, while it waits for the signature of the guest. The signature process uses a local plugin and involves no network communication (hence not visible in the figure). When signed, the receipt is removed from the temporary storage and sent to the RMS for permanent storage. Finally, PMS sends all the necessary stay's information to the DMS. During the stay, the guest's personal data are collected and stored in InfluxDB time-series database. Figure 10 depicts the data from a temperature sensor considering that the only sensor chosen by the guest was this sensor. The Guest uses the receipt to

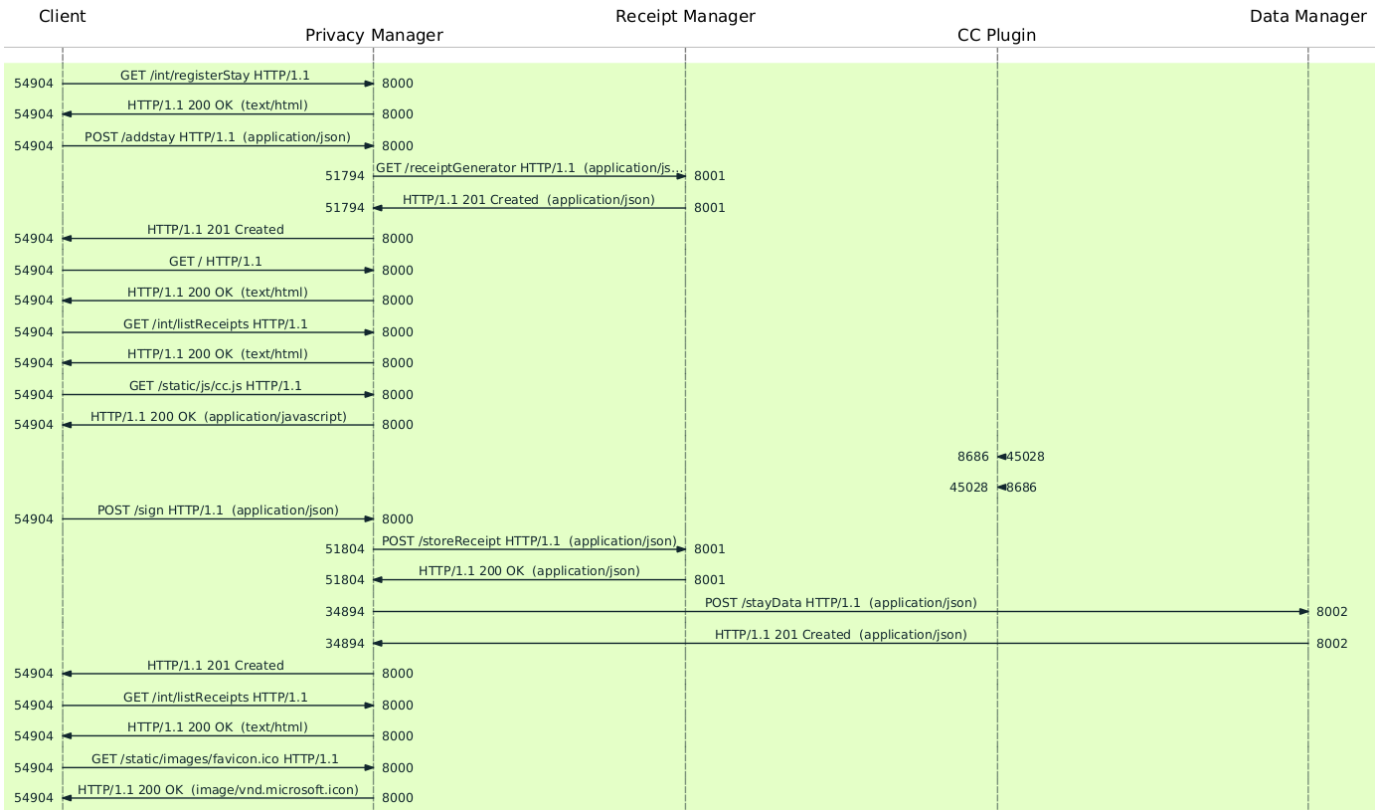


Fig. 9. Packet capture for a new registration.



Fig. 10. Readings of a temperature sensor.

have access to the DMS, as depicted in Figure 11. The receipt contains the endpoint of the DMS, by accessing that endpoint the guest can upload the receipt, the *Signature Application* validates the signature on it using the Portuguese Citizen Card. The guest is granted access to the stay information and data if the signature is valid. When the Guest requests a data

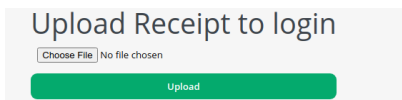


Fig. 11. Data Manager access using the receipt.

deletion through the DMS then the Owner is notified through the PMS (Figure 12). In order to ensure that only the guest can control his personal data, the DMS is the unique service that can access the database where the data from sensors are

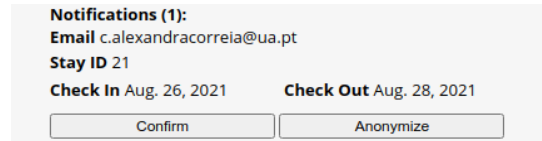


Fig. 12. Notification about data deletion request.

stored. The data deletion operation includes the interaction with personal data thus the DMS (data_manager) accesses the InfluxDB database (home.cassiopeia) to request data deletion and as we can see, is the unique service that interacts with the database where the personal data were stored. Moreover, the operation to export data to a CSV file also interacts directly between DMS and InfluxDB database.

VI. GAPS AND CONCLUSIONS

An approach to address the SmartBnB problem was proposed in this paper, focusing on developing a user-centric approach where the individuals can decide about what happens with their own personal data and manage it by controlling who can access it by using consent receipts. Three important components were developed with specific functionalities: stay management (create stays or associate users to stays); data management (distributed service to centralize the user's personal data control over their personal data); receipt management (generate and store the receipts also in a distributed

way). Moreover, additional components were developed and integrated to simulate a very complete SmartBnB scenario: Home Assistant was integrated as a unified control platform to manage the smart devices in the simulated smart home; and a signature application was developed to provide the ability to sign the receipt as proof of the consent ensuring its integrity.

Unfortunately, the current approach also considers a very restrictive scenario and only makes it possible to control the data inside the home. Controlling data inside the home is strictly important to restrict the content available across different individuals based on their roles. In the proposed solution, it is considered two different privacy breaches: the owner cannot access the renter's data; and different renters cannot access the previous renter's data. For the first case, the Home Assistant was adapted and we can ensure that the owner cannot access the renter's data by creating two different views (one for the renter with the consented devices and another one for the owner with the non-consented devices by the renter). For the second case, also through the Home Assistant configuration, we automatically change the password to the renter's view when the stay ends. In this way, the previous renter cannot access the future renter's data. However, this approach does not consider when the smart home has different renters at the same time. A clear definition of the permissions across different renters should be considered due to the possibility to have families with more than one member including children. In this way, a more efficient delegation service should be integrated to facilitate the management of different views and functionalities depending on the consented entities, consented devices, and user's identity.

REFERENCES

- [1] V. Jesus and H. J. Pandit, "Consent receipts for a usable and auditable web of personal data," *IEEE Access*, vol. 10, pp. 28 545–28 563, 2022.
- [2] V. Jesus, "Towards an accountable web of personal information: The web-of-receipts," *IEEE Access*, vol. 8, pp. 25 383–25 394, 2020.
- [3] ISO/IEC, "ISO/IEC WD TS 27560 Privacy technologies — Consent record information structure," 2021.
- [4] J. Chen, L. Edwards, L. Urquhart, and D. McAuley, "Who is responsible for data processing in smart homes? Reconsidering joint controllership and the household exemption," *International Data Privacy Law*, vol. 10, no. 4, pp. 279–293, 09 2020. [Online]. Available: <https://doi.org/10.1093/idpl/ipaa011>
- [5] V. Jesus, "Blockchain-enhanced roots-of-trust," in *2018 International Conference on Smart Communications and Networking (SmartNets)*, 2018, pp. 1–7.
- [6] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.
- [7] C. Bartolini, S. Daoudagh, G. Lenzini, and E. Marchetti, "Gdpr-based user stories in the access control perspective," in *International Conference on the Quality of Information and Communications Technology*. Springer, 2019, pp. 3–17.
- [8] S. Dramé-Maigné, M. Laurent, L. Castillo, and H. Ganem, "Centralized, distributed, and everything in between: Reviewing access control solutions for the iot," *ACM Computing Surveys (CSUR)*, vol. 54, no. 7, pp. 1–34, 2021.
- [9] P. Patil, M. Sangeetha, and V. Bhaskar, "Blockchain for iot access control, security and privacy: a review," *Wireless Personal Communications*, vol. 117, no. 3, pp. 1815–1834, 2021.
- [10] H. Liu, D. Han, and D. Li, "Fabric-iot: A blockchain-based access control system in iot," *IEEE Access*, vol. 8, pp. 18 207–18 218, 2020.
- [11] S. Ding, J. Cao, C. Li, K. Fan, and H. Li, "A novel attribute-based access control scheme using blockchain for iot," *IEEE Access*, vol. 7, pp. 38 431–38 441, 2019.
- [12] S. Pal, A. Dorri, and R. Jurdak, "Blockchain for iot access control: Recent trends and future research directions," *Journal of Network and Computer Applications*, p. 103371, 2022.
- [13] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K. R. Choo, and Y. Park, "Certificateless-signcryption-based three-factor user access control scheme for iot environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3184–3197, 2020.
- [14] A. K. Das, M. Wazid, A. R. Yannam, J. J. P. C. Rodrigues, and Y. Park, "Provably secure ecc-based device access control and key agreement protocol for iot environment," *IEEE Access*, vol. 7, pp. 55 382–55 397, 2019.
- [15] S. A. Chaudhry, K. Yahya, F. Al-Turjman, and M.-H. Yang, "A secure and reliable device access control scheme for iot based sensor cloud systems," *IEEE Access*, vol. 8, pp. 139 244–139 254, 2020.
- [16] Q. Wang, N. Li, and H. Chen, "On the security of delegation in access control systems," in *European Symposium on Research in Computer Security*. Springer, 2008, pp. 317–332.
- [17] N. Tapas, F. Longo, G. Merlino, and A. Puliafito, "Experimenting with smart contracts for access control and delegation in iot," *Future Generation Computer Systems*, vol. 111, pp. 324–338, 2020.
- [18] R. Xu, Y. Chen, E. Blasch, and G. Chen, "Blendcac: A smart contract enabled decentralized capability-based access control mechanism for the iot," *Computers*, vol. 7, no. 3, p. 39, 2018.
- [19] S. Pal, T. Rabehaja, A. Hill, M. Hitchens, and V. Varadharajan, "On the integration of blockchain to the internet of things for enabling access right delegation," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2630–2639, 2020.
- [20] K. Rantos, G. Drosatos, K. Demertzis, C. Ilioudis, A. Papanikolaou, and A. Kritsas, "Advocate: a consent management platform for personal data processing in the iot using blockchain technology," in *International Conference on Security for Information Technology and Communications*. Springer, 2018, pp. 300–313.
- [21] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013, including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X13000241>
- [22] V. Morel, M. Cunche, and D. Le Métayer, "A generic information and consent framework for the iot," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 366–373.
- [23] S. Human, H. J. Pandit, V. Morel, C. Santos, M. Degeling, A. Rossi, W. Botes, V. Jesus, and I. Kamara, "Data protection and consenting communication mechanisms: Current open proposals and challenges," in *2022 IEEE European Symposium on Security and Privacy Workshops (EuroSPW)*, 2022, pp. 231–239.
- [24] A. Das, M. Degeling, D. Smullen, and N. Sadeh, "Personalized privacy assistants for the internet of things: Providing users with notice and choice," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 35–46, 2018.
- [25] P. Pappachan, M. Degeling, R. Yus, A. Das, S. Bhagavatula, W. Melicher, P. E. Nacini, S. Zhang, L. Bauer, A. Kobsa, S. Mehrotra, N. Sadeh, and N. Venkatasubramanian, "Towards privacy-aware smart buildings: Capturing, communicating, and enforcing privacy policies and preferences," in *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2017, pp. 193–198.