CARPOOL: Secure And Reliable Proof of Location

Sayon Duttagupta
COSIC, KU Leuven
Leuven, Belgium
sayon.duttagupta@esat.kuleuven.be

Dave Singelée
COSIC, KU Leuven
Leuven, Belgium
dave.singelee@esat.kuleuven.be

Xavier Carpent
University of Nottingham
Nottingham, UK
xavier.carpent@nottingham.ac.uk

Volkan Guler
KBC Bank
Leuven, Belgium
guler.volkan@yahoo.com

Takahito Yoshizawa
COSIC, KU Leuven
Leuven, Belgium
takahito.yoshizawa@esat.kuleuven.be

Seyed Farhad Aghili
Sirris
Leuven, Belgium
seyedfarhad.aghili@sirris.be

Aysajan Abidin
COSIC, KU Leuven
Leuven, Belgium
Aysajan.Abidin@esat.kuleuven.be

Bart Preneel
COSIC, KU Leuven
Leuven, Belgium
bart.preneel@esat.kuleuven.be

Abstract

Multiple authentication solutions are widely deployed, such as OTP/TOTP/HOTP codes, hardware tokens, PINs, or biometrics. However, in practice, one sometimes needs to authenticate not only the user but also their location. The current state-of-the-art secure localisation schemes are either unreliable or insecure, or require additional hardware to reliably prove the user's location. This paper proposes CARPOOL, a novel, secure, and reliable approach to affirm the location of the user by solely relying on location-bounded interactions with commercial off-the-shelf devices. Our solution does not require any additional hardware, leverages devices already present in a given environment, and can be integrated effortlessly with existing security components, such as identity and access control systems. To demonstrate the feasibility of our work and to show that it can be deployed in a realistic closed environment setting, we implemented a proof of concept realisation of CARPOOL on an Android phone and multiple Raspberry Pi boards and integrated CARPOOL with Amazon Web Services (AWS) Cognito.

CCS Concepts

• **Security and privacy** → **Authentication**; **Multi-factor authentication**; **Graphical / visual passwords**; *Access control*; Cryptography; Authorization.

Keywords

Location-based authentication, IoT security, CyberPhysical systems

1 Introduction

With the emergence of Internet-of-Things (IoT) devices, users' quality of life has significantly improved, facilitated by smart devices in home and industrial automation, and security services, enhancing convenience. User authentication remains a fundamental aspect of any information and communications technology system's security architecture. Despite the consensus that passwords are both insecure and cumbersome, they continue to proliferate, prompting the exploration of various alternative authentication methods, each with its own set of advantages and disadvantages.

Currently, the username/password scheme remains the predominant method for end-user authentication, despite its well-documented security flaws and lack of user-friendliness [8]. Common authentication schemes are based on one or more of the following factors: (i) Proof-of-Possession (e.g., a token), indicating something the user possesses; (ii) Proof-of-Knowledge (e.g., a password), signifying something the user knows; or (iii) Proof-of-Characteristics (e.g., biometrics), representing something the user is. However, these methods fall short in providing Proof-of-Location, crucial for verifying a user's physical location securely, such as in access control scenarios. Although there has been a shift in the academic literature to propose new contextually aware authentication solutions [20, 28, 39], none of them are *usable* enough to be physically deployed for commercial use. They either rely on some additional hardware (UWB, distance bounding) or use a probabilistic key generation/agreement solution, which results in some degree of false positives and a much slower execution time.

1.1 Location-based access control

There are many access control scenarios where it is worth considering authenticating the location of the user. Let us assume a location where only a specific set of users can get physical access, due to the physical or social security measures in place. For example, to access one's private house, one either needs to have the key to open the house door or break into the house (with all the risks involved). Similarly, many office buildings have physical access control means, such as badge systems, to prevent outsiders from entering the premises. Moreover, when these outsiders arrive at a specific location, there is a high likelihood that they will get detected. Therefore, if one can implicitly assume that only authorised users can be present at a protected location, then applying proof-of-location to validate claims of being at that location would increase the barrier for an adversary to successfully spoof the authentication system.

One practical use case of location-based access control is an official meeting where only the participants of the meeting should have access to the documents being shown (that is, if and only if they are physically present inside the meeting room) and only during that meeting. For the sake of uniformity and presentation, in the rest of this paper, we will use this specific use case to explain our solution. However, there are various other location verification use cases where one can rely on the interaction with devices tied to a specific location. For example, when used at home, CARPOOL allows the owner of the house to make (a selection of multimedia) content available to guests on their personal devices only when they are physically present inside the house, without worrying about strangers also accessing the content. Yet another example is a hotel use case. Using CARPOOL, hotels can offer location-specific services or make specific content available to hotel guests when they are physically present at specific locations in the hotel, while restricting access to people outside or non-guests. In the healthcare industry, CARPOOL can be used in instances where a practitioner can get access to the resource/data *iff* they are physically present inside a given hospital room, or in a governmental facility, users are allowed access to confidential data *iff* they are physically located inside a specific perimeter.

1.2 Problem statement

Realising secure location verification is a challenging problem. Besides accuracy and reliability, one of the main research challenges to be solved is security. *How can one ensure that a user is actually located where they claim to be?* Multiple technical approaches could be taken, we refer to Sect. 8 for a more detailed overview. Unfortunately, all these solutions have important drawbacks, either they require additional hardware, rely only on a single device, do not work indoors or only offer limited security guarantees. Moreover, none of the existing work can be easily and effortlessly deployed in the real world at this moment. We argue that none of the state-of-the-art solutions provide a perfect balance between security, usability and deployability, and we wanted to bridge this research gap by providing a secure solution to verify a user's location, without relying on additional hardware and components, and computationally light enough to be effortlessly run on low-powered embedded devices.

In this paper, we want to explore an alternative approach and propose CARPOOL: seCure And Reliable PrOof Of Location without any additional hardware devices. CARPOOL is based on the concept of *location proofs*. Instead of buying new and additional hardware to assist in authentication, we rely on existing devices which are *already* present in that given room and performing their primary functions. We leverage their communication channels and any additional properties these devices may have, to communicate with the user's device and provide secure verification of the user's location on top of conventional user authentication. It starts from the observation that a typical room has multiple devices installed. For example, in the case of an office room, there are telephone or teleconferencing systems, a screen and/or projector, smart light bulbs, air quality and/or temperature sensors, etc. Any user physically located inside the room can use and interact with these devices via their personal device (e.g., smartphone). Therefore, we want to explore how to leverage the presence of the devices installed in the room to allow a user to prove to a third party that she is located in that room. Important design constraints are that (i) we do not want to install extra hardware in the room, (ii) we only want to rely on interactions that a conventional smartphone can support, and (iii) we want our solution to be compatible with existing authentication and access control schemes.

Location-based Authentication* versus *Location-based Access Control. Although often used interchangeably in the literature, these concepts differ subtly. Most existing location-based authentication schemes described in the literature, which rely on location as the *sole* criterion for authentication, are found to be insecure, unreliable, or both (see Sect. 8). In contrast, we propose a scheme that integrates standard authentication and authorisation solutions with secure verification of the user's location – aiming for scalability, efficiency, deployability, reliability, and security. This research aims to bridge the gap by designing a solution where access to a resource is conditioned not only on security policies, resource specifics, and the user's identity but also on the user's location. For simplicity, we will refer to our approach as location-based access control in the rest of this paper.

Manufacturer's incentives. For our solution, office devices must support the CARPOOL protocol. We envisage that device vendors or manufacturers could either produce CARPOOL-compatible devices or issue firmware updates for compatibility. Since CARPOOL doesn't necessitate adding hardware-specific features, manufacturers can easily adapt their existing devices for compliance through a straightforward software *over-the-air* update. This requires minimal firmware modification—merely 20-80 lines of code (LoC) as shown in Sect. 9 – without altering the hardware. This approach offers manufacturers the potential for enhanced services and additional revenue. Drawing from previous commercial success stories, like Apple AirTag [21] and Matter [14] where the vendors adjusted their business roadmaps and priorities based on the growing usage and popularity of the function, we envision the same for CARPOOL as well. An almost ready-to-deploy code is made available, and the manufacturer needs to make a simple function call and some adjustments to make their device CARPOOL compliant. Furthermore, given our solution's flexibility regarding device usage, many existing devices could easily become CARPOOL-compliant, broadening the scope for protocol adoption.

Another advantage of our solution is its practical back-end integration with off-the-shelf components, such as authentication and access control software. It's possible to deploy necessary software modules as wrapper functions that interact with the unmodified back-end modules. This integration and the feasibility of our solution are demonstrated in our proof-of-concept implementation, detailed in Sect. 6.

1.3 Contributions of our paper

This paper proposes CARPOOL, a novel location-based access control scheme that leverages interactions with local commercial off-the-shelf devices and integrates smoothly with traditional security services like identity and access management systems. The salient motivation of our work is to bridge the gap between usable and deployable solutions, with a particular focus on designing a solution that is ready to be deployed now, while providing security. In Sect. 8, we claim and show that none of the existing solutions provide a balance between security, usability and deployability, and we are the first to introduce a balance between them. The key contributions of this paper include:

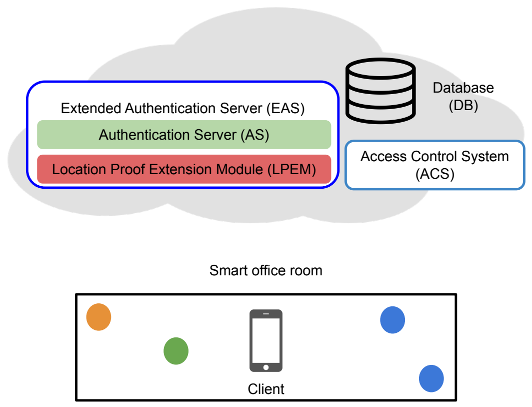

Extended Authentication Server (EAS)
Authentication Server (AS)
Location Proof Extension Module (LPEM)
Database (DB)
Access Control System (ACS)
Smart office room
Client

Figure 1: High-level architecture of the CARPOOL system. The dots represent various different types of smart devices.

- Design of the CARPOOL protocol incorporating an additional authentication check based on the user's location.
- Evaluation of viable location-bounded communication interfaces.
- Solution does not rely on purchasing any additional hardware.
- Proof of concept implementation using an Android smartphone and three Raspberry Pi devices, and integration with the cloud-based back-end system.

Availability Statement. The entire source code and artefact underlying this paper can be found at https://github.com/KULeuven-COSIC/CARPOOL.

2 Main concept of CARPOOL

2.1 System architecture

Figure 1 illustrates the generic system architecture of CARPOOL within a smart office room context. The user, equipped with a mobile device (referred to as *client*), aims to authenticate their presence in a specific room. The process begins with the user logging in using their credentials (*username* and *password*, or any other MFA scheme), followed by interactions with various devices located in the room, such as projectors, telephone systems, and light switches. The core aspect of our approach is that successful authentication requires (i) successful login, (ii) interaction with the devices at the time of authentication, and (iii) the condition that this interaction can solely be performed by users physically present in the same room as the devices. Upon authentication, the user gains the ability to access designated resources, for example, viewing a specific document on a server. In our system architecture, the main components include:

Smart office room: In our scenario, we have a smart office building with several rooms. Among these, certain rooms, such as designated meeting rooms, enable the user to authenticate their presence.

Extended Authentication Server (EAS): This server component verifies the authentication factors of the user and her location. It consists of two subcomponents:

Authentication Server (AS): The AS verifies all of the user's authentication factors, excluding those related to location. In our architecture, the AS functions as a commercial off-the-shelf (COTS) software component, meaning no modifications are needed. It operates under the assumption that, upon successful authentication, the AS issues a token (signed JSON tokens) as its output.

Location Proof Extension Module (LPEM): Since the AS, being a COTS component, does not recognise our security concept and protocols, we augment it with an additional software component, the LPEM. It enriches the AS by supporting the necessary APIs and functionalities required for CARPOOL. It is aware of all smart office rooms within the building that are CARPOOL-compatible and verifies the user's location factor.

Database (DB): The database contains the resource the user wants to access. We do not require any changes to this component, hence it is unaware of our proposed security concept and its protocols.

Access Control System (ACS): To ensure only authorised users can access specific resources in the database, we presume an access control system is in place to enforce this. The decision to grant access to a resource depends on the token issued by the AS, security policies, and the specific resource the user wishes to access. The ACS is considered a COTS component as well, thus not requiring any modifications for our purposes. Consequently, it remains unaware of the proposed security concept and its protocols.

Mobile client: The user's mobile device, such as a smartphone, is referred to as the client in our architecture. This mobile client is utilised for accessing resources in the database. Due to the enforcement of location-based access control, the mobile client must execute the CARPOOL protocol to obtain a token for presentation to the ACS. We assume the mobile client is a standard smartphone equipped with conventional I/O interfaces (screen, microphone, etc.) and supports common wireless technologies.

Local devices: Rooms supporting location proofs are outfitted with various devices suitable for CARPOOL, provided they are (i) fixed within the room, making unnoticed relocation challenging, and (ii) equipped with at least one *location-bounded* output interface for interacting with users or mobile clients. Such interfaces necessitate close proximity for effective interaction, with screens, loudspeakers, and short-range RF communication channels being common examples. More on these interfaces in Sect. 5. We differentiate between *online devices*, which are network-connected (e.g., via WiFi or cable) and can thus interface with the LPEM, and *offline devices*, which lack network connections and direct communication to the LPEM. Our goal is to utilise pre-existing devices within office spaces. Once made CARPOOL compliant through manufacturing adjustments or firmware updates, these devices can participate in the CARPOOL protocol by facilitating location-bounded interactions.

2.2 Actors

We briefly clarify the responsibilities of the different parties in a CARPOOL system below.

Device Manufacturer: Devices must support the CARPOOL location authentication protocol and be administrator-configurable (Sect. 4). Manufacturers need to integrate this feature into new devices or apply a lightweight firmware update to existing ones. The incentives are detailed in Sect. 1.

System Administrator: The system administrator performs the onboard/initial configuration of devices in a CARPOOL system.

User: The user interacts with the CARPOOL system for authentication using her personal device (e.g., smartphone).

2.3 Security policy

CARPOOL allows a user to successfully authenticate themselves and access a specific resource *iff* they are physically located in a given location, by interacting with various devices present in their vicinity. The number of devices the user has to interact with depends on the security policy. This policy is set up dynamically by the administrator and depends on the sensitivity of the resource to protect. For highly confidential resources, the access control will involve a larger number of devices. For less critical resources, the policy can be moderate and require fewer device interactions.

2.4 High-level protocol description

The core concept of our CARPOOL protocol involves a multi-step authentication process enabling a mobile client to access a specific database resource. This process includes:

- **Initial Authentication:** The mobile client connects to the Extended Authentication Server (EAS), providing authentication details (username and password or another authentication method supported by the Authentication Server (AS)) along with its location to the Location Proof Extension Module (LPEM).
- **Token Issuance and Encryption:** Following successful authentication, the AS issues a freshly signed token. The LPEM then encrypts this token with a newly generated random key S and sends the encrypted token back to the mobile client.
- **Decryption Key Acquisition:** To decrypt the token, the mobile client needs to compute the decryption key S by executing the CARPOOL protocol.
- **Selective Interaction with Local Devices:** According to predefined security policies, the mobile client interacts with select local devices (online or offline) to prove its location. Importantly, multiple combinations of device interactions are permitted, allowing the mobile client flexibility in choosing which devices to engage with to authenticate its presence. It's not necessary to interact with all devices, but rather any combination that satisfies the selected security policy.
- **Share Collection and Token Decryption:** The mobile client collects shares from the interactions with devices, along with a share from the LPEM. Combining these shares reconstructs the decryption key S, enabling the client to decrypt the token and forward it to the Access Control System (ACS).
- **Resource Access:** Once authorised, the mobile client can access the desired resource in the database.

3 Threat model and security assumptions

In our CARPOOL scheme, the mobile client must authenticate with the Extended Authentication Server (EAS) to obtain a valid token issued by the Authentication Server (AS).

Security Assumptions. The Location Proof Extension Module (LPEM) is assumed to have full knowledge of the network topology of all CARPOOL-enabled rooms, including the devices present and their location-bounded interfaces. A loose time synchronisation between the LPEM and offline devices is assumed, with minor deviations tolerated. Security policies specify valid device interaction combinations for location verification, are readable by all entities in the system, and can only be modified by authorised personnel. Online communications between the back-end system, mobile clients, and online devices are assumed to be secured via TLS. Persistent or stealth tampering with devices is assumed to be noticeable due to the regular physical presence of individuals in meeting rooms, prompting updates to security policies when necessary.

Insider versus External Attackers. The primary focus is on external attackers attempting to authenticate remotely without being physically present in the designated room. Insider threats, including malicious authorised users capturing screen contents or distributing retrieved resources externally, are not in scope. Addressing such cases requires additional security mechanisms, such as Digital Rights Management (DRM) or Data Loss Prevention (DLP), which can be integrated separately if required. Consequently, a notable class of attacks that falls outside our scope is the terrorist fraud attack, where an accomplice inside the office room forwards shares from smart devices to a remote adversary attempting authentication. This scenario has been well studied in the literature [6, 16, 29, 30], but effective countermeasures often require additional hardware or distance-bounding techniques. These solutions introduce significant complexity and overhead, making them impractical for large-scale commercial deployment at present. Since terrorist fraud attacks rely on collusion between an external adversary and an insider, they are distinct from the standalone external adversary model that CARPOOL primarily defends against.

Attack Scenarios. An adversary who successfully logs in and obtains an encrypted token at time T may attempt to compromise location verification remotely. The following attack strategies are considered:

(1) Remote Attacker at Authentication Time (T). The adversary is outside the room at time T but attempts to obtain the necessary secret shares remotely. This could involve using specialised equipment to interact with devices over a distance, such as long-range cameras for visual channels, directional microphones for audio, or advanced antennas for BLE signals. CARPOOL mitigates this risk by enforcing multi-device interaction thresholds, requiring simultaneous compromise of multiple independent location-bounded channels, significantly raising attack complexity.

(2) Adversary with Prior Physical Access ($P < T$). The attacker previously accessed the room and left behind a device to later capture shares. Some channels, such as Bluetooth or audio, may be susceptible to such attacks, whereas others, like visual interfaces, require precise calibration, making undetected exploitation significantly harder. CARPOOL mitigates this by requiring a combination of channels, increasing the likelihood of detection for hidden devices attempting unauthorised interactions. Alternatively, the adversary may attempt to tamper with existing devices to extract cryptographic material or alter their functionality. Such tampering is constrained by physical security measures, such as secure storage of cryptographic keys, hardware protections, and detection mechanisms. The longer a compromised device remains in the room, the higher the probability of detection, leading to policy updates that render the attack ineffective.

(3) Adversary Relocating Devices ($P < T$). An attacker with prior room access may move a device to another location to facilitate

remote authentication at time T. This is mitigated by physical security measures and policy updates upon detection. Additionally, CARPOOL's reliance on multiple devices ensures that relocating a single device is insufficient to bypass the location proof.

CARPOOL increases the cost and complexity of attacks by enforcing a minimum number of required device interactions, defined by a threshold l. The adversary's practical budget is assumed to allow compromise of at most y devices, ensuring that $y < l$ holds. This enforces a security threshold that makes large-scale, simultaneous device compromises infeasible.

4 CARPOOL scheme

4.1 Protocol flow

Our CARPOOL protocol, depicted in Figure 2, unfolds across six distinct phases: (1) authentication; (2) authorisation; (3) encryption and generation of secret shares; (4) collection of online shares; (5) reconstructing the key and decryption; and (6) verification and access to the resource. Prior to delving into each phase in detail, we initially outline the one-time setup process required for the local devices.

4.1.1 Setup phase. Before implementing the CARPOOL protocol, each local device within the system must undergo initialisation. Starting with local devices, as previously noted, vendors or manufacturers are responsible for installing firmware to render these devices CARPOOL-compliant, achievable either during the manufacturing process or via a firmware update. For newly produced devices, it's advisable for vendors to also embed an initialisation key at the point of manufacture, subsequently providing this key with the device (e.g., detailed in the manual or encoded in a QR code). System administrators can utilise this initialization key to customise the local device and input keying material, which may consist of a unique secret offline key shared with the LPEM for offline share generation or a TLS key-pair accompanied by a certificate. For pre-existing devices, secure configuration relies on the available interfaces.

Adding a new local device to the system necessitates updating the LPEM with details such as the device's IP and MAC address (for online devices) or offline key (for offline devices), its specific location, and the supported location-bounded interface.

4.1.2 Authentication phase. The authentication phase encompasses user login and verification. In the login step, a user wishing to authenticate must send four pieces of information to the EAS server: (i) username, (ii) password (or another standard authentication factor), (iii) room number (RN), and (iv) policy number (PN).

The AS employs the username and password for user authentication. The remaining two items, RN and PN, are passed on to the LPEM. RN indicates the user's claimed location, specifically the smart office room they are in. As previously mentioned, each room is associated with one or more security policies, defining acceptable device interaction combinations for proving location. Users have the option to choose their preferred interaction combination from those available for a room, as indicated by PN.

4.1.3 Authorisation phase. In this step, a signed JSON access token is generated for the authenticated user. This token contains the identity information of the user, an expiration time and her access privileges, and is signed by the AS.

4.1.4 Encryption and generation of secret shares. During this phase, the following three steps are executed.

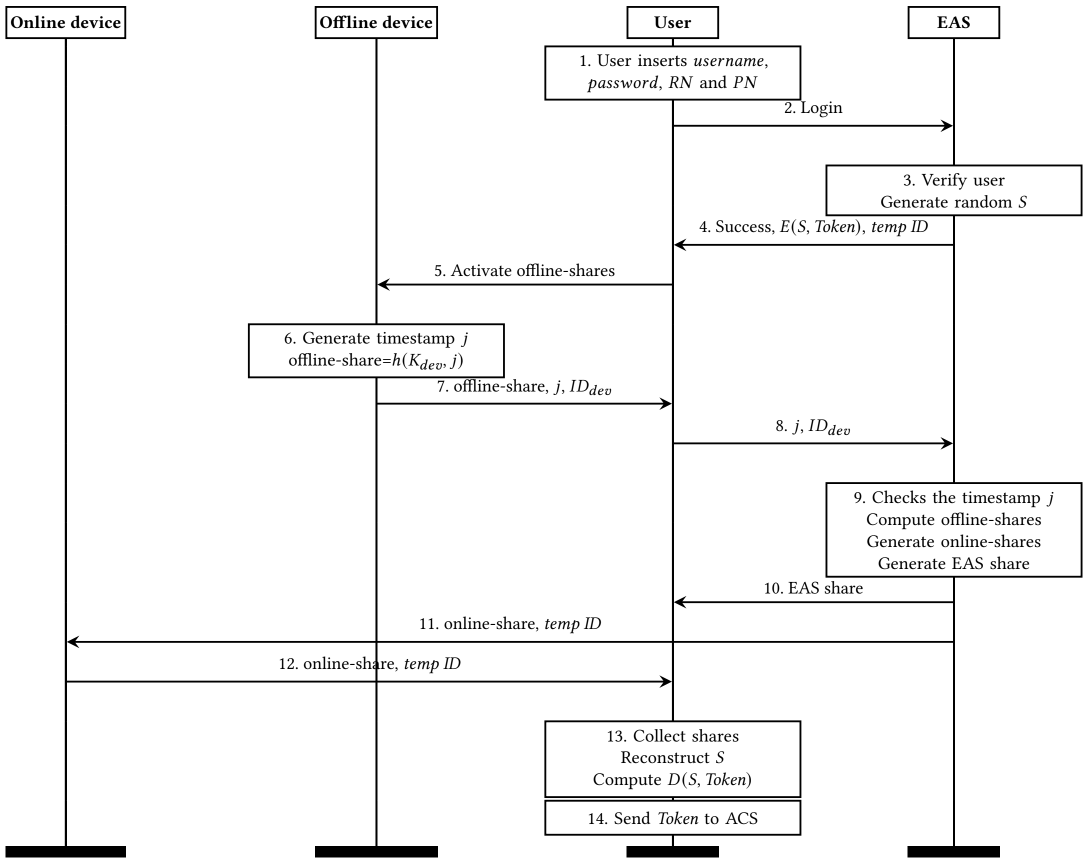
- Upon receiving the token, the EAS generates a random symmetric encryption key S and uses this key to encrypt the token. Next, the EAS sends $E(S, \mathit{Token})$ to the user, together with a temporary identity *temp ID* (e.g., a rotating counter). We use *temp ID* to indicate which user is trying to authenticate during multiple concurrent sessions and not the *username* of the user to avoid any privacy leakage to protect against an attacker who listens on a communication channel.
- The key S is divided into shares using additive secret sharing, with each share recommended to be at least 128-bits in length. More details on the secret-sharing process will be discussed in Sect. 4.2. The shares are categorised into online and offline types. For generating the offline shares, the EAS first requires the offline shares. Upon receiving the encrypted token, the user initiates the offline device(s) by sending an "Activate offline-shares" message through the device's location-bounded interface. The offline device then generates a timestamp j and, using its unique secret key K_{dev}, inputs these into a cryptographic pseudo-random function h to produce its offline share. This share, along with the timestamp j and the device's identity ID_{dev}, is sent to the user via the location-bounded interface. The user then forwards timestamp j and ID_{dev} to the EAS, which will discard this share if the timestamp's deviation from the current time is too significant. The maximum tolerable deviation, set by the system administrator, is argued to be reasonably up to a few minutes for most practical applications. Finally, the EAS computes the offline shares using the accepted tuples (j, ID_{dev}), the secret keys K_{dev}, and the pseudo-random function h.
- Once the offline shares have been computed, the EAS generates the other shares (i.e., online shares and its own share) using the additive secret sharing algorithm and the security policy PN. Once all these shares are generated, the EAS sends its own share to the user and the online shares to the respective online devices, together with the temporary identity *temp ID* of the client. In Sect. 4.2, we will explain why we exactly need this share from the EAS (in addition to the online and offline shares).

At the end of this step, the user has the encrypted token, the offline shares of the offline devices, the EAS's share and a temporary identity *temp ID*.

4.1.5 Collection of online shares. In this phase, the user engages with the online devices through their designated location-bounded interfaces, obtaining the online share from each interaction. Upon receiving each online share, the first step is to verify the temporary identity, confirming the share genuinely pertains to that user.

4.1.6 Reconstructing the key and decryption. After gathering offline shares, online shares, and the EAS's share, the user reconstructs the key S, decrypts $E(S, \mathit{Token})$, and presents the decrypted token to the ACS for database access.

4.1.7 Verification and access to the resource. When the back-end server (ACS) receives the token, it will evaluate it. If the token is valid and not yet expired, the data will be given to the user when



Online device
Offline device
User
EAS
1. User inserts *username*, *password*, *RN* and *PN*
2. Login
3. Verify user
Generate random S
4. Success, $E(S, Token)$, *temp ID*
5. Activate offline-shares
6. Generate timestamp j
offline-share=$h(K_{dev}, j)$
7. offline-share, j, ID_{dev}
8. j, ID_{dev}
9. Checks the timestamp j
Compute offline-shares
Generate online-shares
Generate EAS share
10. EAS share
11. online-share, *temp ID*
12. online-share, *temp ID*
13. Collect shares
Reconstruct S
Compute $D(S, Token)$
14. Send *Token* to ACS

Figure 2: Message diagram of CARPOOL

the user is authorised (not shown in the figure). In all other cases, access to the data will be denied.

4.2 Additive secret sharing

4.2.1 Background. In our protocol, we employ an additive secret-sharing based threshold secret-sharing scheme, utilizing the disjunctive normal form (DNF). DNF is a logic structure made up of a disjunction of conjunctions, essentially an OR of ANDs. Consider $\mathcal{P}$ as the set of n players, $\mathcal{P} := \{P_1, P_2, \cdots, P_n\}$, which in our scenario includes both online and offline devices. Our objective is to distribute a secret key S among $\mathcal{P}$'s players so that only specific, authorised combinations of these players can reconstruct S. The access structure, denoted as Γ, defines these authorised sets. The DNF expression allows us to articulate the conditions under which the secret key S can be reconstructed by the involved players.

$$\bigvee_{X \in \Gamma} \Big(\bigwedge_{P_i \in X} P_i \Big) .$$

4.2.2 Additive secret sharing in CARPOOL. Recall that for each room, there are one or more combinations of interactions with devices that serve as valid location proof, each identified by a policy number (PN). Assuming there are m combinations for a specific room, the DNF expressing the access structure for this room can be represented as follows:[*]

$$\Big(\bigwedge_{1} P_i \Big) \vee \Big(\bigwedge_{2} P_j \Big) \vee \ldots \vee \Big(\bigwedge_{m} P_l \Big) .$$

Additive secret sharing is applied by dividing the secret S into random shares for each device P_i within the authorised sets for the m combinations, expressed as $(\bigwedge P_i)$. For example, consider a scenario in a specific room where two combinations are permitted: (1) interaction with devices A, B, and C; and (2) interaction with devices C and D. In this case, the secret S would be split into additive shares s_i as follows:

$$\begin{aligned} S &= s_{1,A} + s_{1,B} + s_{1,C} \\ S &= s_{2,C} + s_{2,D} \end{aligned} \tag{1}$$

Note that in this example, device C gets shares $s_{1,C}$ and $s_{2,C}$, and all other devices can get only one share. The secret S can be reconstructed by either collecting all the shares $s_{1,i}$ or the shares $s_{2,j}$.

[*]This approach may become impractical for a large number of players due to exponential growth in the access structure's size, it remains viable for our smart office scenario, which involves only a few online/offline devices

In our system, there's no specific assumption about the quantity of online and offline devices, which implies a scenario where potentially all devices could be offline. Each offline device, during a protocol run, generates a timestamp j and computes a fresh offline share using its secret key and j. This scenario poses a challenge, as illustrated in our example, where the equation Eq. 1 might not hold if all devices are offline. To address this, we introduce an extra constraint ensuring every allowed combination of interactions includes interaction with the EAS, which is inherently met since the mobile client must engage with the EAS, specifically the LPEM. This adjustment modifies Eq. 1 as follows:

$$\begin{aligned} S &= s_{1,A} + s_{1,B} + s_{1,C} + s_{1,EAS} \\ S &= s_{2,C} + s_{2,D} + s_{2,EAS} \end{aligned} \tag{2}$$

Now, these equations will always hold, as the EAS can compute its own local share ($s_{1,EAS}$ or $s_{2,EAS}$ in Eq. 2) such that the sum of the shares in each line is always equal to the key S.

5 Location-bounded communication channels

The security of our solution strongly relies on the local interaction and communication between the mobile client and nearby devices located in the smart office room. Each device has its specific location-bounded communication channel. For our solution to be practical and deployable, we need to use communication channels that are practical and easy to use, which are supported by most mobile clients and which offer some degree of security (i.e., it should not be trivial for an adversary to use this communication channel when not located in the same room). Moreover, the channel should be suitable for key transport (i.e., the secret shares).

5.1 Design requirements

Before selecting suitable communication channels, it's essential to define the design requirements for location-bounded communication channels, drawing from CARPOOL's needs. Inspired by Bonneau et al.'s authentication framework [8], these requirements fall into five categories: Security (S), Usability (U), Reliability (R), Communication (C), and Deployability (D). A summary of these requirements is presented in Table 1, with further details available in Appendix A.

5.2 Candidates for location-bounded communication channels

The attributes previously outlined also apply to location-bounded communication channels, each characterised by distinct SURCD (Security, Usability, Reliability, Communication, Deployability) properties. This paper focuses on four primary types of location-bounded communication channels feasible for deployment in an office environment: (i) visual channel, (ii) audio channel, (iii) light intensity, and (iv) short-range RF, like Bluetooth Low Energy (BLE).

However, no single channel perfectly meets all five macro-design requirements. For instance, while audio or light intensity channels may offer greater security, they might lack the reliability of channels like BLE, which, though easier to deploy, may offer reduced security. We will now delve deeper into these four categories of location-bounded communication channels.

Table 1: CARPOOL design requirements

Category	Requirements
Security	Strong cryptographic protection (S1)
	Revocation (S2)
	Isolation (S3)
	Availability (S4)
	Resilient-to-Physical-Observation (S5)
	Resilient-to-Internal-Observation (S6)
	Resilient-to-Guessing (S7)
	Requiring-Explicit-Consent (S8)
Usability	Limited user participation (U1)
	Easy-to-Learn (U2)
	Easy-to-use (U3)
	Efficient-to-Use (U4)
	Memory-wise effortless (U5)
	Nothing to carry (U6)
Reliability	Low Rejection (R1)
	Infrequent-Errors (R2)
Communication	Efficient to transmit (C1)
Deployability	Support for personal devices (D1)
	Scalability (D2)
	Lightweight (D3)

Visual. This channel facilitates embedding large shares within QR codes or barcodes displayed on screens, making it easily accessible for users to capture codes through a smartphone camera. Deploying visual devices is straightforward, thanks to the widespread availability of devices supporting visual channels (e.g., projectors, TV screens). The deterministic nature of displaying QR codes or barcodes ensures high reliability and a low rejection rate for this channel. However, a potential drawback is the possibility of capturing screen content from afar using a telescopic camera lens.

Acoustic: Utilising an audio channel, data is modulated on frequencies receivable by smartphones, including the voice communication spectrum. Audio channels, offering wide bandwidth, allow for transmitting substantial data volumes with minimal user interaction—merely activating the smartphone's microphone. They operate without additional hardware, compatible with any device having speakers (e.g., conference systems). Their deployment ease and use of ultrasonic (or near-ultrasonic) frequencies (≥18 KHz), inaudible to most, reduce bystander disturbance. Nonetheless, the probabilistic nature of acoustic channels may lead to significant error rates without precise encoding. Balancing error correction with bit transmission speed is essential. Security-wise, audio's penetrative capacity raises interception risks, and its vibrations might be remotely detected by advanced eavesdropping techniques [11].

Light. Like acoustic channels, light can transmit secret shares by modulating intensity. It offers faster bandwidth with minimal user interaction, using devices like smart light bulbs or smartphone cameras, without needing extra hardware. However, light shares acoustic channels' drawbacks: its probabilistic nature allows remote signal capture, and internal lighting may cause interference. Additionally, fluctuating light intensity could discomfort some users, such as those with epilepsy.

Bluetooth. Short-range RF communication through Bluetooth or Bluetooth Low Energy (BLE) acts as a location-bounded communication channel, offering easy deployment due to extensive device compatibility and requiring minimal user interaction. The channel is divided into beacon messages, which allow for passive reception without user involvement, and regular communication that necessitates device pairing, demanding some user effort for setup. Security varies between these types; paired devices can encrypt communication, thwarting remote snooping, while unpaired beacon signals are vulnerable to long-distance interception by sophisticated receivers.

Although we show that none of the location-bounded channels, which can be used to distribute the shared key, possesses all the required design properties; we claim that as long as the user selects the combination of such channels which in total supports all the properties, the scheme is said to be secure and CARPOOL can thus guarantee the working of the protocol. For example, the user selects a visual channel along with a paired-BLE, or a light source with a beacon BLE.

6 Proof of concept implementation

We made a working prototype of our CARPOOL solution to demonstrate its feasibility and show that it can be easily integrated into existing security systems such as commercial identity and access management systems.

6.1 Our demonstration setup

Our demonstration setup, shown in Figure 3, aims to validate CARPOOL's concept and its practical applicability for location-based authentication. As discussed in Sect. 2.1, the system consists of a cloud-hosted back-end, a user's mobile client, and local devices, encompassing both online and offline devices with distinct communication interfaces. Online devices are connected to the back-end through the meeting room's WiFi, with details in Table 2. Let us now zoom in on each of the entities in our demonstration setup:

Back-end server: The back-end server consists of the EAS, ACS and a database, with the former consisting of two sub-components (LPEM and AS). To implement these four back-end entities, we used different Amazon Web Services (AWS) services.

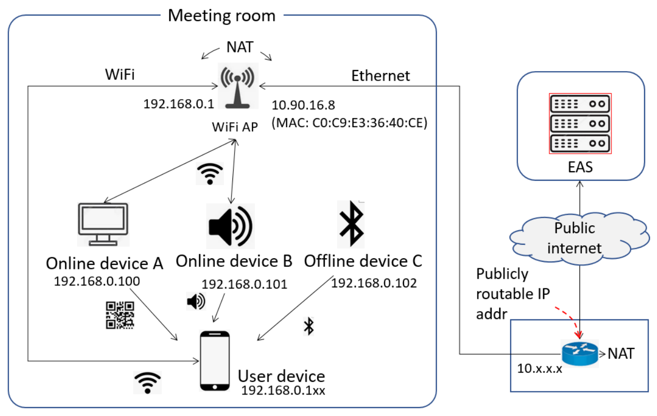


Figure 3: Network topology of demonstration setup

Table 2: Entities in our proof of concept

Entity	Sub-entities
Back-end (cloud) server	EAS={LPEM, AS}, ACS, DB
Mobile client	Android smartphone
Online device	Visual, Audio channel
Offline device	Bluetooth channel

- LPEM: We utilised an AWS EC2 instance with Ubuntu 18.04 for flexibility and scalability, leveraging Amazon EC2 for simplified setup and capacity management [43]. The LPEM's key generation and share distribution were efficiently programmed in Python3.
- AS: We leveraged AWS Cognito for identity and access management (IAM), utilising its Identity Pools service to manage system users, such as meeting room attendees [41].
- ACS: We employed AWS Cognito, particularly the User Pools service, for the ACS. Upon confirming an authorised user, AWS Cognito issues an ID token and access token after successful authentication and authorisation [42].
- DB: For ease of implementation, we used an off-the-shelf AWS RDS MySQL database instance, an isolated database environment running in the cloud [44].

Mobile client: In our implementation, the mobile client is a smartphone running on Android. The client application, which executes the CARPOOL protocol and interacts with the cloud-based back-end system, is implemented in Java (Android API version 8.1).

Online and offline devices: Our setup includes two online devices and one offline device, each simulated on a separate Raspberry Pi 3 Model A+. For visual communication, one Raspberry Pi is equipped with a mounted screen, while the audio channel is enabled by attaching external loudspeakers. Both are connected to the back-end server through WiFi. The third Raspberry Pi, representing the offline device, also features a touch screen, which, though not essential, facilitates user interaction for beacon message activation.

6.2 Implementation details back-end server and mobile client

We'll delve into the CARPOOL protocol's implementation, starting with the back-end server and mobile client, then addressing online and offline devices and their role in location-bounded communication.

During the CARPOOL protocol, users are navigated through login and subsequent device interactions via specific location-bounded interfaces, as depicted in Fig. 4. These steps aim to (i) enhance usability by clarifying necessary actions and (ii) minimising user errors during authentication.

6.2.1 *Authentication phase.*

As outlined in Sect. 4, the client initiates the connection by logging into the EAS and specifying the room and policy numbers. After inputting the room number, the mobile client accesses that room's specific policies stored in an Amazon Simple Storage Service (Amazon S3) bucket,[†] enabling straightforward retrieval of security policies for any office room.

[†]https://docs.aws.amazon.com/s3/index.html#amazon-s3

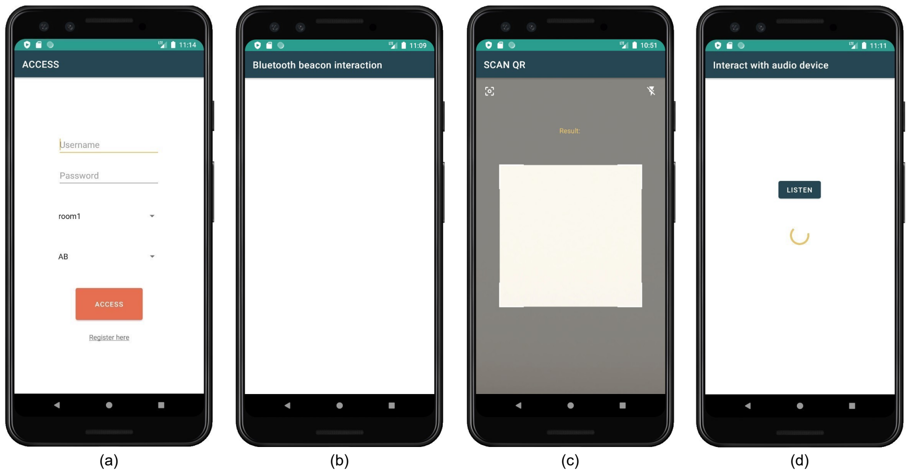


Figure 4: The CARPOOL Android Application life-cycle. (a) the main home screen, (b) the mobile client actively listens for an incoming BLE packet and automatically proceeds to the next step when it receives one (no user action required), (c) the user is instructed to scan a QR code, and (d) the user activates the mobile client to listen for the near-ultrasonic audio transmission.

Only administrators can update these policies. For user-friendly policy selection, devices in the room are represented by letters:

- A: Online Visual Device
- B: Online Audio Device
- C: Offline Bluetooth Device
- D: EAS server

6.2.2 *Authorisation phase.* In our setup, authenticated users receive an access token indicating their access level, linked to the ID token from the AS, with identical validity periods. AWS Cognito was chosen for AS and ACS due to its seamless feature integration, supported by the "boto3" Python library for easy implementation. The LPEM intercepts all mobile client to EAS communications, verifying room and policy numbers, and rejects invalid policy numbers. After authentication, AWS Cognito issues ID and access tokens.

6.2.3 *Encryption and generation of secret shares.* After generating the user's ID and access tokens, the LPEM intercepts them, creates a 128-bit key S, and encrypts the tokens using AES-CBC with a fixed salt, utilising the Fernet library. It dispatches $E(S, Token)$ and a $temp\ ID$ to the mobile client. The LPEM then awaits the mobile client's transmission of offline shares, identified by ID_{dev} and timestamps, verifies these timestamps, computes the corresponding offline shares, and discerns active online devices for the CARPOOL protocol according to the selected security policy. For each, it computes a random 128-bit share, alongside its own. For example, with one offline and two online devices in our setup, S is divided into four shares: three for the devices and one for the EAS.

6.2.4 *Reconstructing the key and decryption.* This phase is exclusively handled by the mobile device. Our Android implementation made use of necessary software to interact with the location-bounded communication interfaces for online/offline devices. Leveraging Android's native audio (`MediaRecorder` and `AudioRecord`) and Bluetooth libraries, along with the Budiyev-QR scanner for device data acquisition, the smartphone's camera, microphone, and Bluetooth functionality are essential in our prototype.

Upon collecting all shares pertinent to the selected security policy and identifying those specific to the user via the temporary identity, the mobile client aggregates them to reconstruct the secret key S. Utilising S and a hard-coded salt in our Android app, the token is decrypted. Subsequently, the decrypted tokens are transmitted to the EAS for accessing database resources.

6.2.5 *Verification and access to the resource.* In this phase, the Extended Authentication Server (EAS) receives both the ID and access tokens from the mobile client. AWS Cognito verifies and validates these tokens. If the verification is successful, access to the database is granted. To facilitate this process, we utilised the `warrant` and `pycognito` libraries in Python, which did not require any modifications for our implementation.

6.3 Implementation details location-bounded communication channels

6.3.1 *Online device with the visual communication channel.* In our demonstrator, the device with a visual communication channel functions online, obtaining a 128-bit share from the LPEM. The system is configured with the visual device as a UDP server and the LPEM as a UDP client, a setup crucial for enabling the LPEM to trigger share calculation and transfer. Consequently, the online device is prepared to receive shares from the LPEM whenever necessary. Upon receiving a share, the device translates it into a QR code displayed on an attached screen. The user captures this QR code using her smartphone to secure the online share.

6.3.2 *Online device with audio communication channel.* The second online device in our demonstrator employs an audio communication channel, based on a Raspberry Pi 3 model. For our protocol, a 128-bit share is transmitted from this audio-enabled device to the mobile client using near-ultrasonic frequencies ($f_c \geq 18$ kHz) via the acoustic channel. This frequency band is chosen because it's inaudible to most humans, reproducible by most speakers, and

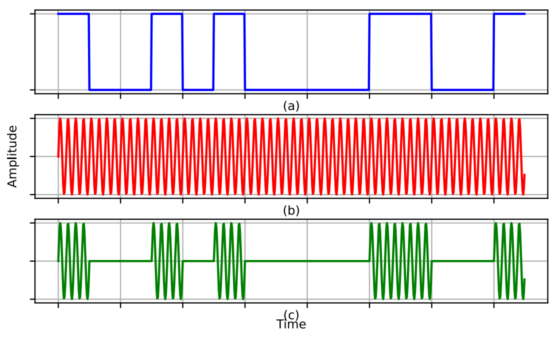


Figure 5: The audio encoding process – (a) the binary data (digital wave) to be encoded, (b) the carrier (analogue) wave, and (c) the modulated OOK wave (analogue signal) which is being played by the speakers.

capturable by smartphones. Employing on-off keying (OOK) modulation, the amplitude of the carrier wave is modulated with the secret share, illustrated in Fig 5. Opting not to develop a full signal processing algorithm, we utilised Android's audio processing libraries. With a constant carrier frequency (f_c) of 18 kHz and a sampling frequency (f_s) of 44.1 kHz within a 10-second window, we modulate the digital data (secret share s) into an analogue sound wave ($s.sin(2\pi f_c t)$, where $t = 1/f_s$). This high sampling frequency allows the transmission of many bits, expanding each bit of the secret share by 3330 times for modulation. This $(1, 3330)$ repetition code enhances the audio channel's reliability extremely, and produces no error or false positives. The smartphone's microphone captures this sound, demodulating it into digital format to recover the original secret share s.

6.3.3 Offline Bluetooth beacon device. The third device, an offline Bluetooth interface on the same Raspberry Pi model as online devices, doesn't connect to the LPEM for share reception. It generates an HMAC-SHA-256 value of the timestamp as its offline share, preventing replay attacks, and requiring loose synchronization with the LPEM's clocks. The share includes the timestamp and its HMAC, with the key pre-shared with the LPEM. Unique device IDs in the QR code differentiate shares from multiple offline devices.

We utilise the *Eddystone URL* message [51] to transmit offline shares, embedding the lowest 10 octets of the HMAC output due to the message's short length. The mobile client extracts and forwards the offline share to the LPEM upon receiving the message.

Unlike online devices triggered by the LPEM, the offline device lacks automatic interaction cues with clients. Activation via a Raspberry Pi touchscreen is required to generate and send a new offline share through BLE beacon. This method, while secure, offers less protection than online devices due to share truncation. A standard Bluetooth connection could enhance security but would increase the need for user interaction.

7 Evaluation and Discussion

Our CARPOOL solution meets the design requirements presented in Sect. 5.1. It clearly meets the **deployability** requirements because, by design, it relies on the use of a personal device (e.g., smartphone) and commercial off-the-shelf devices present in a room. A typical smartphone has all the necessary capabilities and interfaces to interact via different location-bounded communication channels, such as visual, light, audio or short-range RF. We argue that a fully deployable solution has not been studied in the literature, and moreover, we claim to achieve a desired balance between security, deployability and usability, as shown in Appendix A. Moreover, we do not provide any restrictions about the placement of the devices in the CARPOOL protocol, the devices can be moved around, positions interchanged, and also simultaneously with other dynamic objects or persons in the room, and it will still work accurately.

Efficiency is maintained with minimal overhead; online devices display or transmit minimal data, and offline devices compute a single hash, resulting in simple operations under 100 lines of code per device. The mobile device has negligible computational overhead, as it only needs to compute the secret key by addition and then perform one decryption operation. The computational costs of the operations are extremely minimal and negligible for even a battery-powered microcontroller, although we envision our model to be used with more commercially available devices, which are (generally) AC-powered. Almost all of the major overheads are outsourced to the server, which runs on Amazon AWS in our instance. By running our real-world experiments multiple times, our average authentication time was around **3.4 seconds** (depending on the policy), with a maximum of 15 seconds (if audio was chosen). The audio channel is the only probabilistic channel we use, and due to the $(1, 3330)$ repetition code, we claim our solution has absolutely **no error or false positives**, even after running the experiments multiple times in a noisy environment with sufficient entropy – hence extremely **reliable**.

Usability is enhanced in CARPOOL, with most interactions requiring minimal or no user interaction. Post-authentication, the mobile phone guides the user through easy tasks (e.g., scanning the QR code, pressing a button to start recording an audio signal, etc.), making the process intuitive for even non-technical users without additional memorisation or hardware. The only usability property that is debatable is *efficient-to-use (U4)*. CARPOOL clearly increases the time to perform the entire authentication procedure. In our practical experiments, the entire procedure typically took between **2 and 15 seconds** (*maximum*) to perform, depending on the chosen policy level. **Communication and reliability** aspects are detailed in Sect. 5 and in the Appendix A. Discussing **security**, CARPOOL incorporates measures for secure authentication, elaborated further below.

Strong cryptographic protection (S1): The key S and its shares are each 128 bits. We assume a remote adversary might get up to $y < l$ shares, where l is the minimum needed for security policies. Thus, an adversary must brute-force at least one 128-bit share. Moreover, AS's tokens are signed, preventing spoofing.
Revocation (S2): Secret shares are single-use, becoming invalid after successful authentication. Additionally, the AS can revoke users, preventing them from receiving valid tokens.

Isolation (S3): CARPOOL ensures isolation as all keys and shares are randomly generated without knowledge of keys among different actors, including past or future protocol shares.
Availability (S4): In CARPOOL, the LPEM maintains continuous connections with online devices and the mobile client. The system allows various interaction combinations with devices, letting users select alternative security policies if a device fails. Administrators can update policies to reflect changes in device availability, and there is no single point of failure.
Resilient-to-Physical-Observation (S5): This attack doesn't apply because physical observation by a remote adversary is impossible, and shares from different protocol instances are independent.
Resilient-to-Internal-Observation (S6): In CARPOOL, shares are randomly generated per protocol instance, preventing one client from using another's shares. Each instance produces a new set of shares, rendering old or previously recorded shares invalid, making CARPOOL resilient to replay attacks.
Resilient-to-Guessing (S7): CARPOOL is resistant to guessing attacks as an adversary must guess undetected shares or the 128-bit key S. CARPOOL does not reveal incorrect shares upon protocol failure, forcing an adversary to attempt breaking all shares without gaining an advantage from breaking just one, as it remains indistinguishable from the rest.
Requiring-Explicit-Consent (S8): This property is realised by CARPOOL since the user initiates the authentication process and the shares cannot be generated unless the primary authentication step is completed.

Composite security and residual device compromise. Each CARPOOL share is encrypted under a fresh 128-bit key, so off-line brute-forcing one ciphertext already costs 2^{128} trials. Reconstructing the session secret, however, demands that an adversary compromise *all* l independent, location-bounded channels *within the same protocol run*: scrape the on-screen QR code with a long-range camera, relay the BLE advertisement using a directional antenna, and eavesdrop the ultrasonic audio with a high-gain microphone — all before the per-run timestamp window (Sect. 3) expires. This simultaneity requirement inflates the effective attack space to $2^{128\times l}$, rendering a practical relay or guessing attack economically infeasible even for a well-funded adversary. Dormant ("offline") devices could in principle suffer invasive tampering such as clock manipulation, yet CARPOOL can simply exclude them: the policy engine may rely solely on *online* units whose freshness is verified at run time, or even require several channels from the *same* device to raise robustness[‡]. If a device is later suspected of compromise, the administrator revokes it immediately, because the additive threshold remains satisfied by any other trusted combination, authentication continues without weakening future sessions, and past keys are not exposed.

[‡]Selecting multiple channels per device is optional but incurs no protocol overhead beyond an extra share transmission.

8 Related work

Location-aware authentication solutions are the ones that rely on parameters or contextual information to gain an idea of the user's location or to prove their physical presence. Location-based authentication was pioneered by the seminal work of Denning and MacDoran [12] who came up with the idea of using location as a tool for authentication. Since then, multiple new solutions have been proposed. In the overview below, we cover different alternative approaches for location-based authentication and highlight their shortcomings. In Appendix A, we list the design requirements we believe such authentication protocols should possess, and we extensively compare and evaluate our work with existing work and we visually compare them in Table. 3 and show that the prior state-of-the-art fails to fulfil all these requirements at once. We claim that none of the existing solutions provide a truly secure, usable and deployable scheme and we are the first to introduce the balance between these three design elements.

GNSS. The first obvious choice would be to locate the user based on GNSS[§] such as GPS and Galileo. Various works are based on relying on the smartphone's or laptop's GPS data [5, 9, 13, 17, 38]. However, this approach faces two major issues. Firstly, GPS-based methods are insecure, primarily due to the ease of GPS signal spoofing [45], making them unsuitable for location-based authentication. Secondly, GPS is ineffective indoors. Marforio et al. [25] suggested using the smartphone's Trusted Platform Module (TPM) to sign GPS coordinates, aiming to prevent compromised devices from sending false locations. Nevertheless, this doesn't stop GPS spoofing, fails to function indoors, and slows down protocol execution.

Hardware-based Security keys. A simpler option is to install a security token, such as USB key or NFC reader in each room for user authentication. However, these devices are both insecure [33, 34, 49] and unreliable [10, 18]. Besides the cost of new hardware for every room (which doesn't scale well), they also present a single point of failure. If the key malfunctions or gets stolen, users can't authenticate until it's replaced. In contrast, CARPOOL allows policy adjustments to utilise other sets of functioning local devices.

Context-based pairing. Many seminal techniques for establishing a cryptographic key or pairing devices use contextual data, such as acoustic [39], visual [27, 36, 48], gyroscopic [23, 26], touch [52], or vibrational [2, 37] channels. However, most of these solutions face either security or deployability challenges. Sound-Proof [22] uses ambient noise for secure device pairing and claims to work both indoors and outdoors with minimal user interaction. Yet, Shrestha et al. [46] identified a security flaw in Sound-Proof, and Basin et al. [7] argue that it lacks usability because it doesn't account for human factors and errors. Visual channel-based techniques have become insecure, as high-resolution cameras can capture images of barcodes or QR codes from a distance, breaching the system's security [35]. Miettinen et al. [28] proposed a pairing solution using ambient sound, light, or both. However, due to the probabilistic nature of these channels, significant error correction is needed, impacting practicality. Vibrational channels are vulnerable to security attacks [3, 19]. Solutions based on simultaneous gestures or touch [23, 24, 26, 52] are user-friendly but often require additional

[§]https://www.gps.gov/systems/gnss/

Table 3: Comparison with other relevant works.
Design requirements are ● satisfied, ◑ partially-satisfied, and ❍ unsatisfied.

Design		GNSS / GPS					tok-	Context-based pairing														Distance bounding/RSSI					CARPOOL
Requirements		[5]	[9]	[13]	[17]	[38]	ens	[1]	[2]	[20]	[22]	[23]	[24]	[26]	[27]	[28]	[36]	[37]	[39]	[48]	[52]	[4]	[31]	[32]	[50]	[53]	[this work]
Security	S1	❍	◑	❍	❍	◑	❍	❍	●	●	❍	●	●	●	❍	◑	❍	●	●	❍	◑	❍	❍	●	●	●	●
	S2	❍	❍	❍	❍	❍	●	●	●	❍	◑	❍	❍	●	◑	◑	●	●	❍	●	❍	●	●	●	❍	❍	●
	S3	❍	❍	❍	❍	❍	●	●	●	❍	●	●	●	●	●	●	●	●	●	●	❍	●	●	●	●	●	●
	S4	❍	❍	❍	❍	❍	◑	●	●	◑	●	●	●	●	●	●	●	●	●	❍	◑	◑	●	●	●	●	●
	S5	◑	◑	◑	◑	◑	●	●	●	●	❍	❍	❍	❍	●	●	●	●	●	●	●	●	◑	◑	◑	◑	●
	S6	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
	S7	❍	❍	❍	❍	❍	●	●	●	●	●	◑	◑	◑	●	●	●	●	●	●	◑	●	●	●	●	●	●
	S8	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	❍	●	●	●	●	●
Usability	U1	●	●	●	●	●	●	❍	❍	◑	◑	❍	❍	❍	●	●	◑	❍	●	●	❍	●	❍	❍	❍	❍	●
	U2	◑	●	❍	●	●	●	❍	❍	◑	◑	❍	❍	❍	●	❍	◑	❍	●	❍	❍	●	❍	❍	❍	❍	●
	U3	◑	●	❍	●	●	◑	❍	❍	◑	◑	❍	❍	❍	●	❍	◑	❍	●	❍	❍	●	❍	❍	❍	❍	●
	U4	◑	◑	◑	◑	◑	●	❍	●	❍	◑	◑	◑	◑	❍	❍	◑	●	❍	◑	❍	❍	❍	◑	●	❍	◑
	U5	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	❍	●	●	●	●	●	●	●	●
	U6	●	●	●	●	●	❍	●	❍	●	●	●	●	●	❍	◑	❍	●	●	❍	●	●	❍	❍	❍	❍	●
Reliability & Communication	R1	●	●	●	●	●	●	◑	❍	◑	◑	❍	◑	❍	●	◑	◑	●	❍	●	❍	❍	◑	◑	◑	●	●
	R2	❍	❍	❍	❍	❍	●	◑	❍	◑	◑	❍	◑	❍	●	◑	◑	●	❍	●	❍	❍	◑	◑	◑	●	●
	C1	●	●	●	●	●	●	●	●	❍	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Deployability	D1	●	●	●	●	●	❍	●	●	●	●	●	◑	❍	●	●	●	●	●	●	❍	●	❍	❍	❍	❍	●
	D2	●	●	●	●	●	❍	●	❍	●	❍	❍	❍	❍	●	◑	●	◑	❍	❍	❍	●	❍	❍	❍	❍	●
	D3	●	●	●	●	●	●	●	●	❍	●	●	●	●	●	●	●	●	●	●	●	●	❍	❍	❍	❍	●

hardware or are limited to proprietary hardware, reducing their functionality. Agadakos et al. [1] proposed a location-based authentication solution using user interactions, IoT sensors, and other factors. Although more efficient and faster than smartphone-only solutions, it has significant error rates and security vulnerabilities, as physical locations can be easily spoofed. Perceptio [20] uses time as the underlying contextual commonality between different IoT devices to perform autonomous pairing and agree on a shared key. However, the overall execution time to complete the protocol is in the order of hours to days, as it requires multiple instances of the same exact pairing to occur to increase the confidence score of the underlying symmetric key. Paired with the inability to move the devices for the entire protocol execution (to increase the confidence score, the protocol needs to run with the exact same devices in the exact same location and order), makes the solution not at all practical and usable.

Distance bounding. Distance-bounding protocols combine cryptography with tight time-of-flight measurements to assert an upper bound on distance. ZeroTouch [4] provides geofencing for devices without using additional hardware and only RSS values, but suffers from false acceptance and rejection rates, which can hamper key generation. Move2-Auth [53] relies on the correlation between two concurrent RSS traces collected while the user performs specific gestures, which hurts usability whenever the gestures have to be repeated to correct errors. Other studies have explored RF and Wi-Fi signals for distance bounding [32, 50]. Rasmussen *et al.* [31] proposed an ultrasound-based distance-bounding system for implantable medical devices, but Sedighpour *et al.* later showed it to be vulnerable to worm-hole relays [40]. The main limitation shared by these protocols is their need for special-purpose radios, anchors or sensors, which clashes with our goal of relying exclusively on commercial off-the-shelf equipment.

Key take-away. *While purpose-built UWB radios ([32]) or Wi-Fi CSI rigs ([50]) can indeed solve the relay problem, they still require new hardware and professional installation. The novelty of CARPOOL is that it "**CARPOOLises**" devices already present; a software update only is enough to obtain secure, relay-resistant location proofs without extra capital expenditure.*

9 Availability

The entire code of our proof of concept implementation is available in https://github.com/KULeuven-COSIC/CARPOOL.

10 Conclusion

Traditional authentication mechanisms aim to verify the identity of the user, based on something the user knows, has or is. However, these authentication factors might not be sufficient for security-sensitive access control applications. We envision a setting where also the location of the user needs to be verified, along with conventional authentication means. In this paper, we proposed CARPOOL, a novel location-based access control scheme where the proof of the user's location is used as an additional factor, and applied this concept to the use case of a smart office room. To realise the notion of proof of location, we leverage the communication and interaction with multiple commercial off-the-shelf devices located in the office room. Our solution does not require any new hardware to be installed and only needs the devices in the smart office room to support our location-based access control protocol. The security of our scheme relies on additive secret sharing, where successful reconstruction of the secret is only possible when the user is physically present at the specific location. We have implemented our work to demonstrate the feasibility and the low effort to integrate with existing access control systems and to validate the efficiency of CARPOOL. Although we applied our solution in a smart office scenario, it can also be used in other settings where location-based access control is required.

Acknowledgement

This work was supported by CyberSecurity Research Flanders with reference number VR20192203.

References

[1] Ioannis Agadakos, Per Hallgren, Dimitrios Damopoulos, Andrei Sabelfeld, and Georgios Portokalidis. 2016. Location-Enhanced Authentication Using the IoT: Because You Cannot Be in Two Places at Once. In *Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16)*.

[2] S. Abhishek Anand and Nitesh Saxena. 2016. Vibreaker: Securing Vibrational Pairing with Deliberate Acoustic Noise. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '16)*.

[3] S Abhishek Anand and Nitesh Saxena. 2017. Coresident Evil: Noisy Vibrational Pairing in the Face of Co-Located Acoustic Eavesdropping. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '17)*.

[4] Nikola Antonijević, Sayon Duttagupta, Dave Singelée, Enrique Argones Rúa, and Bart Preneel. 2025. ZeroTouch: Reinforcing RSS for Secure Geofencing. In *Proceedings of the 30th ACM Symposium on Access Control Models and Technologies (SACMAT '25)*. doi:10.1145/3734436.3734448

[5] Daniel Ashbrook and Thad Starner. 2003. Using GPS to Learn Significant Locations and Predict Movement across Multiple Users. *Personal and Ubiquitous Computing* 7, 5 (2003).

[6] Gildas Avoine, Xavier Bultel, Sébastien Gambs, David Gérault, Pascal Lafourcade, Cristina Onete, and Jean-Marc Robert. 2017. A Terrorist-fraud Resistant and Extractor-free Anonymous Distance-bounding Protocol. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '17)*. 15 pages.

[7] David Basin, Saa Radomirovic, and Lara Schmid. 2016. Modeling Human Errors in Security Protocols. In *IEEE 29th Computer Security Foundations Symposium*.

[8] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. 2012. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *2012 IEEE Symposium on Security and Privacy (S&P'12)*.

[9] Yohan Chon, Hyojeong Shin, Elmurod Talipov, and Hojung Cha. 2012. Evaluating mobility models for temporal prediction with high-granularity mobility data. In *2012 IEEE International Conference on Pervasive Computing and Communications*.

[10] Sanchari Das, Andrew Dingman, and L. Jean Camp. 2018. Why Johnny Doesn't Use Two Factor A Two-Phase Usability Study of the FIDO U2F Security Key. In *Financial Cryptography and Data Security*.

[11] Abe Davis, Michael Rubinstein, Neal Wadhwa, Gautham J. Mysore, Frédo Durand, and William T. Freeman. 2014. The Visual Microphone: Passive Recovery of Sound from Video. *ACM Trans. Graph.* (2014).

[12] Dorothy E. Denning and Peter F. MacDoran. 1996. Location-based authentication: Grounding cyberspace for better security. *Computer Fraud & Security* 1996, 2 (1996), 12–16.

[13] Trinh Minh Tri Do and Daniel Gatica-Perez. 2014. Where and what: Using smartphones to predict next locations and applications in daily life. *Pervasive and Mobile Computing* 12 (2014), 79–91.

[14] Sayon Duttagupta, Arman Kolozyan, Georgio Nicolas, Bart Preneel, and Dave Singelee. 2025. What's the Matter? An In-Depth Security Analysis of the Matter Protocol. Cryptology ePrint Archive, Paper 2025/1268. https://eprint.iacr.org/2025/1268

[15] Sayon Duttagupta, Eduard Marin, Dave Singelée, and Bart Preneel. 2023. HAT: Secure and Practical Key Establishment for Implantable Medical Devices. In *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy (CODASPY '23)*.

[16] Marc Fischlin and Cristina Onete. 2013. Terrorism in Distance Bounding: Modeling Terrorist-Fraud Resistance. In *Applied Cryptography and Network Security*.

[17] Lex Fridman, Steven Weber, Rachel Greenstadt, and Moshe Kam. 2017. Active Authentication on Mobile Devices via Stylometry, Application Usage, Web Browsing, and GPS Location. *IEEE Systems Journal* (2017).

[18] Sanam Ghorbani Lyastani, Michael Schilling, Michaela Neumayr, Michael Backes, and Sven Bugiel. 2020. Is FIDO2 the Kingslayer of User Authentication? A Comparative Usability Study of FIDO2 Passwordless Authentication. In *2020 IEEE Symposium on Security and Privacy (S&P)*.

[19] Tzipora Halevi and Nitesh Saxena. 2010. On Pairing Constrained Wireless Devices Based on Secrecy of Auxiliary Channels: The Case of Acoustic Eavesdropping. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*.

[20] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. 2018. Do You Feel What I Hear? Enabling Autonomous IoT Device Pairing Using Different Sensor Types. In *2018 IEEE Symposium on Security and Privacy (S&P'18)*.

[21] Apple Inc. 2021. Apple Find My Network. https://www.apple.com/newsroom/2021/04/apples-find-my-network-now-offers-new-third-party-finding-experiences/.

[22] Nikolaos Karapanos, Claudio Marforio, Claudio Soriente, and Srdjan Capkun. 2015. Sound-Proof: Usable Two-Factor Authentication Based on Ambient Sound. In *24th USENIX Security Symposium (USENIX Security 15)*.

[23] Xiaopeng Li, Fengyao Yan, Fei Zuo, Qiang Zeng, and Lannan Luo. 2019. Touch Well Before Use: Intuitive and Secure Authentication for IoT Devices. In *The 25th Annual International Conference on Mobile Computing and Networking* (Los Cabos, Mexico) *(MobiCom '19)*.

[24] Xiaopeng Li, Qiang Zeng, Lannan Luo, and Tongbo Luo. 2020. T2Pair: Secure and Usable Pairing for Heterogeneous IoT Devices. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20)*.

[25] Claudio Marforio, Nikolaos Karapanos, Claudio Soriente, Kari Kostiainen, and Srdjan Capkun. 2014. Smartphones as Practical and Secure Location Verification Tokens for Payments. In *Proceedings of the Network and Distributed System Security Symposium Symposium, NDSS '14*.

[26] Rene Mayrhofer and Hans Gellersen. 2009. Shake Well Before Use: Intuitive and Secure Pairing of Mobile Devices. *IEEE Transactions on Mobile Computing* 8, 6 (2009), 792–806.

[27] Jonathan M. McCune, Adrian Perrig, and Michael K. Reiter. 2005. Seeing-is-believing: using camera phones for human-verifiable authentication. In *2005 IEEE Symposium on Security and Privacy (S&P'05)*.

[28] Markus Miettinen, N. Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and Majid Sobhani. 2014. Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS '14)*.

[29] Bart Preneel and Dave Singelée. 2005. Location Verification using Secure Distance Bounding Protocols. In *2005 International Workshop on Wireless and Sensor Networks Security (WSNS 2005)*.

[30] Aanjhan Ranganathan, Nils Ole Tippenhauer, Boris Škorić, Dave Singelée, and Srdjan Čapkun. [n. d.]. Design and Implementation of a Terrorist Fraud Resilient Distance Bounding System. In *Computer Security – ESORICS 2012*.

[31] Kasper Bonne Rasmussen, Claude Castelluccia, Thomas S. Heydt-Benjamin, and Srdjan Capkun. [n. d.]. Proximity-Based Access Control for Implantable Medical Devices. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)*.

[32] Kasper Bonne Rasmussen and Srdjan Čapkun. 2010. Realization of RF Distance Bounding. In *Proceedings of the 19th USENIX Conference on Security (USENIX)*.

[33] Thomas Roche. 2024. EUCLEAK. Cryptology ePrint Archive, Paper 2024/1380. https://eprint.iacr.org/2024/1380

[34] Thomas Roche, Victor Lomné, Camille Mutschler, and Laurent Imbert. 2021. A Side Journey To Titan. In *30th USENIX Security Symposium (USENIX Security 21)*.

[35] Eyal Ronen and Adi Shamir. 2016. Extended Functionality Attacks on IoT Devices: The Case of Smart Lights. In *2016 IEEE European Symposium on Security and*

Privacy (EuroSP).
[36] Nitesh Saxena, Jan-Erik Ekberg, Kari Kostiainen, and N Asokan. 2006. Secure device pairing based on a visual channel. In *2006 IEEE Symposium on Security and Privacy (S&P'06).*
[37] Nitesh Saxena, Md. Borhan Uddin, Jonathan Voris, and N. Asokan. 2011. Vibrate-to-unlock: Mobile phone assisted user authentication to multiple personal RFID tags. In *2011 IEEE International Conference on Pervasive Computing and Communications (PerCom).*
[38] Salvatore Scellato, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Andrew T. Campbell. 2011. NextPlace: A Spatio-Temporal Prediction Framework for Pervasive Systems. In *Proceedings of the 9th International Conference on Pervasive Computing (Pervasive'11).*
[39] Dominik Schürmann and Stephan Sigg. 2013. Secure Communication Based on Ambient Audio. *IEEE Transactions on Mobile Computing* 12, 2 (2013), 358–370.
[40] Sahar Sedighpour, Sridjan Capkun, Saurabh Ganeriwal, and Mani Srivastava. 2005. Implementation of Attacks on Ultrasonic Ranging Systems. In *SenSys*.
[41] Amazon Web Services. 2021. AWS Cognito Identity Pool. https://docs.aws.amazon.com/cognito/latest/developerguide/identity-pools.html.
[42] Amazon Web Services. 2021. AWS Cognito User Pool. https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html.
[43] Amazon Web Services. 2021. AWS EC2. https://aws.amazon.com/ec2/.
[44] Amazon Web Services. 2021. AWS RDS. https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.DBInstance.html.
[45] Daniel Shepard, Jahshan Bhatti, and Todd Humphreys. 2012. Evaluation of smart grid and civilian UAV vulnerability to GPS spoofing attacks. In *Proceedings of the 2012 ION GNSS Meeting.*
[46] Babins Shrestha, Maliheh Shirvanian, Prakash Shrestha, and Nitesh Saxena. 2016. The Sounds of the Phones: Dangers of Zero-Effort Second Factor Login Based on Ambient Audio. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16).*
[47] Sophie Stephenson, Bijeeta Pal, Stephen Fan, Earlence Fernandes, Yuhang Zhao, and Rahul Chatterjee. 2022. SoK: Authentication in Augmented and Virtual Reality. In *2022 IEEE Symposium on Security and Privacy (S&P).*
[48] Ahren Studer and Adrian Perrig. 2010. Mobile user location-specific encryption (MULE): using your office as your password. In *Proceedings of the Third ACM Conference on Wireless Network Security (WiSec '10).*
[49] Enis Ulqinaku, Hala Assal, AbdelRahman Abdou, Sonia Chiasson, and Srdjan Capkun. 2021. Is Real-time Phishing Eliminated with FIDO? Social Engineering Downgrade Attacks against FIDO Protocols. In *30th USENIX Security Symposium (USENIX Security 21).*
[50] Deepak Vasisht, Swarun Kumar, and Dina Katabi. 2016. Decimeter-Level Localization with a Single WiFi Access Point. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16).*
[51] Wikipedia.org. 2021. Eddystone (Google). https://en.wikipedia.org/wiki/Eddystone_(Google).
[52] Zhenyu Yan, Qun Song, Rui Tan, Yang Li, and Adams Wai Kin Kong. 2019. Towards Touch-to-Access Device Authentication Using Induced Body Electric Potentials. In *The 25th Annual International Conference on Mobile Computing and Networking* (Los Cabos, Mexico) *(MobiCom '19).*
[53] Jiansong Zhang, Zeyu Wang, Zhice Yang, and Qian Zhang. 2017. Proximity based IoT device authentication. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications.*

A Design requirements

CARPOOL is a special instance of an authentication protocol, as it is used to prove the location of a user. Therefore, we have applied the authentication framework of Bonneau et al. [8], a standard for evaluating authentication protocols [15, 18, 22, 47], to identify the most relevant design requirements. We discussed the **security** properties in Sect. 7, and below we give a detailed overview of all the properties together, and an assessment of the CARPOOL protocol based on these requirements.

A.1 Security properties

Strong cryptographic protection (S1): In CARPOOL, both the key S and each of its shares have a size of 128 bits. In our threat model, we assume that a remote adversary can obtain at most $y < l$ shares, with l being the lower limit on the number of shares required in any of the security policies. Therefore, even in the worst case, the adversary needs to brute-force at least one random 128-bit share. Note that authentication tokens from the AS are typically signed and cannot be spoofed.

Revocation (S2): In CARPOOL, the secret shares are only used once. After every successful authentication, the old shares become invalid. Moreover, the Authentication Server (AS) can always revoke any user, after which that user will no longer receive a valid token from the AS.

Isolation (S3): CARPOOL achieves isolation, since all keys and shares are generated randomly and has no knowledge of the keys established between other actors. This also includes shares from past or future protocol instances.

Availability (S4): In CARPOOL, the LPEM is always connected to the online devices and the mobile client. Moreover, the office room might support multiple combinations of interactions with devices (i.e., security policies). Therefore, if one device no longer works, the user can still choose another security policy that does not require interaction with the broken device. The system administrator can also update the security policies in case a local device is no longer available.

Resilient-to-Physical-Observation (S5): In CARPOOL, this attack is not applicable since (i) physical observation is not possible for a remote adversary, and (ii) shares of multiple protocol instances are independent.

Resilient-to-Internal-Observation (S6): In CARPOOL, random shares are generated for each protocol instance. Therefore, one client cannot use the shares of another client. Moreover, every protocol instance yields a fresh set of shares. Therefore, old or previously recorded shares are invalid. In other words, CARPOOL is resilient to replay attacks.

Resilient-to-Guessing (S7): CARPOOL is resistant against guessing attacks, because an adversary would either have to guess one or more shares it could not eavesdrop, or the key S. Both have a size of at least 128 bits. Moreover, CARPOOL does not inform the adversary if one of the shares is incorrect in case of a protocol failure. Hence, an adversary has to try and break all the shares to successfully break the entire scheme, and will not gain any advantage by breaking just one scheme as it will be indistinguishable from the rest.

Requiring-Explicit-Consent (S8): This property is realised by CARPOOL since the user initiates the authentication process.

A.2 Usability properties

Limited user participation (U1): The number of interactions between the client and the devices should be kept to a minimum. Interactions which are automatic, i.e., without any actions from the user, such as receiving a BLE beacon message, are most usable. Others might require user involvement, such as pointing the client device's camera towards a screen.

Easy-to-Learn (U2): End users who are unaware of the scheme can understand and learn how to use the scheme without any significant difficulty and then easily recall how to use it.

Easy-to-use (U3): Any action that is required by the client, should be easy to perform for an average-skilled user. It should not require any training or technical expertise. For example, a visual communication channel where a QR code needs to be captured, is relatively easier to perform than mimicking gyroscopic movements.

Efficient-to-Use (U4): The time the client must spend for each authentication instance should be short. As a result, the time to obtain a share from a device via a location-bounded communication channel should be short.
Memory-wise effortless (U5): The client should not have to remember and memorize any passwords, keys or shares, besides what was already required by the initial authentication framework of logging in to the Authentication Server using one's own credentials.
Nothing to carry (U6): The client should not be required to carry any other hardware, besides her own personal device (e.g., smartphone), to carry out the authentication protocol. Also, the devices in the smart office room should not require any additional hardware.

A.3 Reliability properties

Low Rejection (R1): The communication channel and the CARPOOL scheme should have a low false rejection rate to avoid the client having to try multiple times to authenticate themselves.
Infrequent-Errors (R2): Ideally, the communication channel and the CARPOOL scheme should be error-prone, and if not, properly error-calculated. A channel can be either deterministic or probabilistic, the former being the most reliable and the latter having a probability of communication errors. The more noise the channel has, the less reliable it gets. The more error correction is needed, the lower its bitrate (i.e., it will take longer to transport a 128-bit share).

A.4 Communication properties

Efficient to transmit (C1): It should be possible to reliably send and transmit the shares in a short span of time.

A.5 Deployability properties

Support for personal devices (D1): The CARPOOL scheme should support the use of personal devices (i.e. *BYOD*) of the user. These should be any commercial, off-the-shelf smartphone or tablet. No additional hardware is required.
Scalability (D2): The security solution should support at least a limited number of devices to communicate with the client and transport the additive shares. It should be easy to extend the scheme and increase the number of devices in the room that can be used to prove the client's location.
Lightweight (D3): It should be feasible to implement the security solution on the entire spectrum of devices typically present in an office room, ranging from smart projectors to low-power BLE beacons. Hence, the CARPOOL protocol should be lightweight enough to support this.