CARPOOL: Secure And Reliable Proof of Location

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

Dave Singelée
DistriNet (Group T), KU Leuven
Leuven, Belgium
dave.singelee@kuleuven.be

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

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

Farhad Aghili
Sirris
Leuven, Belgium
farhad.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

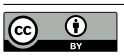
Abstract

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.

ACM Reference Format:
Sayon Duttagupta, Dave Singelée, Xavier Carpent, Takahito Yoshizawa, Farhad Aghili, Aysajan Abidin, and Bart Preneel. 2026. CARPOOL: Secure And Reliable Proof of Location. In *Proceedings of the 31st ACM Symposium on Access Control Models and Technologies (SACMAT '26), July 08–10, 2026, Waterloo, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3750555.3811885



This work is licensed under a Creative Commons Attribution 4.0 International License.
SACMAT '26, Waterloo, ON, Canada
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2107-6/2026/07
https://doi.org/10.1145/3750555.3811885

1 Introduction

User authentication remains a fundamental component of modern information and communications technology systems. Despite extensive evidence that passwords are insecure and cumbersome, the username and password paradigm continues to dominate end-user authentication [8]. Common authentication mechanisms rely on one or more of the following factors: proof of possession, proof of knowledge, or proof of characteristics. However, these mechanisms do not address an orthogonal but increasingly important question: *where* is the user located at the time of access?

Modern identity and access management (IAM) systems increasingly operate in centralised, cloud-based infrastructures, where users authenticate remotely to access sensitive resources. While multi-factor authentication strengthens identity verification, it does not ensure that the user is physically present in a designated secure environment. In many practical settings, access to sensitive data must be conditioned not only on identity but also on verified physical presence within a controlled indoor space. Examples include secure laboratories, regulated production floors, healthcare facilities, and export-controlled zones. In such environments, permitting remote access, even by a correctly authenticated user, may violate compliance policies or organisational security requirements.

A seemingly simple solution would be to confine sensitive resources to a dedicated computer physically located inside the room. However, this approach is often impractical in modern infrastructures. Enterprises rely on centralised cloud storage, unified logging, version control, and federated IAM systems. Resources cannot simply be duplicated or isolated on standalone machines without sacrificing auditability, scalability, and operational consistency. Moreover, users increasingly rely on managed personal devices rather than fixed terminals. These realities motivate the need for secure and deployable mechanisms that condition access on verified physical presence while preserving integration with existing back-end infrastructures.

1.1 Motivating Location-*Conditioned* Access

We consider access control scenarios in which users must be physically present in a designated room to access resources managed

by an existing back-end system. For instance, during a confidential board meeting, participants may access documents hosted on a cloud platform only while inside the meeting room. In healthcare, practitioners may access sensitive patient data only when physically present in a specific hospital ward. In regulated industrial environments, access to production data may be restricted to personnel inside certified facilities. In governmental or defence contexts, certain resources may only be accessed within defined secure perimeters.

In all these cases, what is required is a mechanism that verifies the user's physical presence within a trusted indoor environment and binds this property to the authorisation decision.

1.2 Problem Statement

Realising secure and deployable location verification indoors is challenging. The fundamental question is: *how can a system ensure that a user is physically present in a specific room at the time of authentication?* Existing academic approaches to location-based authentication rely on assumptions or mechanisms that limit practicality, such as specialised hardware (e.g., distance-bounding radios or UWB anchors) or probabilistic context extraction with non-negligible false acceptance and rejection rates [19, 26, 37]. As discussed in Sect. 8, these limitations hinder deployability and usability in real-world indoor settings. As a result, no prior solution simultaneously achieves strong security guarantees, low operational overhead, compatibility with commodity devices, and seamless integration with existing IAM infrastructures.

Our goal is therefore to design a location-conditioned access control mechanism that (i) provides meaningful resistance against remote adversaries, (ii) operates using commercial off-the-shelf devices already present in typical indoor environments, (iii) requires no specialised ranging hardware, and (iv) integrates cleanly with existing authentication and access control systems.

1.3 Overview of CARPOOL

In this paper, we propose CARPOOL: seCure And Reliable PrOof Of Location without any additional hardware devices. CARPOOL augments conventional authentication by requiring a user to interact with multiple devices physically installed in a room. These devices, such as teleconferencing systems, screens or projectors, smart lighting, and environmental sensors, already expose communication interfaces (e.g., visual, audio, light, or short-range RF). CARPOOL leverages these interfaces as location-bounded channels to distribute secret shares that can only be collected by a user physically present in the room.

Instead of introducing new ranging hardware or specialised infrastructure, CARPOOL orchestrates existing devices through lightweight firmware extensions. The protocol combines secret sharing with policy-driven threshold enforcement, ensuring that successful access requires interaction with multiple independent room-bound devices during a single authentication session. This design increases the cost of remote attacks while maintaining compatibility with commodity smartphones and existing back-end services.

Location-based authentication versus location-conditioned access. Unlike schemes that use location as the sole authentication factor, CARPOOL does not replace conventional authentication. Rather, it conditions access on both identity verification and verified physical presence. The result is a second-factor mechanism that strengthens access control decisions in environments where physical presence is security-relevant.

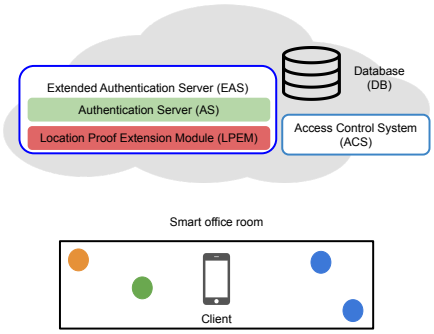


Figure 1: High-level architecture of the CARPOOL system. The dots represent various different types of smart devices.

1.4 Contributions

This paper makes the following contributions:

- We design CARPOOL, a threshold-based location-conditioned access control protocol that composes multiple independent location-bounded communication channels using additive secret sharing.
- We introduce a policy-driven architecture that enables flexible trade-offs between security assurance and usability while remaining compatible with standard IAM and access control systems.
- We demonstrate a deployable design that leverages pre-existing commercial off-the-shelf (COTS) devices and standard smartphone interfaces, avoiding specialised ranging hardware or additional infrastructure installation.
- We provide an end-to-end proof-of-concept implementation integrating Android smartphones, commodity embedded devices, and a cloud-based back-end, and evaluate the security, usability, and performance trade-offs of different policy configurations.

Availability Statement. The entire 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 their 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 equipped with devices suitable for CARPOOL, provided they are (i) fixed within the room, making unnoticed relocation difficult, and (ii) expose at least one *location-bounded* output interface for interaction with users or mobile clients. Such interfaces require close proximity, with screens, loudspeakers, and short-range RF channels being typical examples (see Sect. 5). We distinguish between *online devices*, which are network-connected (e.g., via Wi-Fi or cable) and can communicate with the LPEM, and *offline devices*, which lack network connectivity and direct communication with the LPEM. Our objective is to leverage pre-existing devices in office environments. Once made CARPOOL-compliant through manufacturing integration or firmware updates, these devices can participate in the protocol by enabling 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 thei 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 EAS, providing authentication details (username and password or another authentication method supported by the AS) along with its location to the 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 is 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 ACS.
- **Resource Access:** Once authorised, the mobile client can access the desired resource in the database.

3 Threat model and security assumptions

CARPOOL is a second-factor mechanism that binds access to a claimed location. The mobile client must first authenticate to the EAS and obtain a valid token issued by the AS. CARPOOL aims to prevent a *remote* adversary from completing the location proof without being physically present in the designated room. We do not claim to prevent an adversary who is *already physically present* in the room at authentication time from completing the proof, nor do we address post-authentication exfiltration of resources by authorised users.

Scope and environmental assumptions. CARPOOL is designed for controlled indoor environments (e.g., offices, meeting rooms, healthcare facilities) in which a fixed set of infrastructure devices is present and managed. We assume a baseline level of physical security for the premises and device infrastructure. The scheme is not intended for fully open or adversarial public environments without administrative control over deployed devices.

Security assumptions. The LPEM maintains an up-to-date view of CARPOOL-enabled rooms, including the devices present and their supported location-bounded interfaces. Security policies define valid device interaction combinations for location verification, are readable by all entities, and can only be modified by authorised personnel. A loose time synchronisation between the LPEM and offline devices is assumed, with minor deviations tolerated. Online communications between the back-end system, mobile clients, and online devices are secured via TLS. Device placement within a room does not affect protocol correctness and devices may be repositioned within the room; adversarial relocation is treated explicitly as an attack scenario below.

Attacker classes and scope. We distinguish three attacker classes. *External remote attackers* attempt to authenticate without being physically present in the designated room at the time of authentication. *Insider attackers* are physically present in the room and are either authorised users or adversaries with physical access who can directly observe or interact with devices. *Colluding attackers* combine an external attacker with an insider accomplice. CARPOOL is designed to raise the cost for external remote attackers. Insider misuse and resource leakage by authorised users are out of scope and require orthogonal controls such as Digital Rights Management (DRM) or Data Loss Prevention (DLP).

A notable out-of-scope collusion setting is the terrorist fraud attack, where an accomplice inside the room forwards shares from smart devices to a remote adversary. This scenario has been studied extensively [6, 15, 27, 28], but practical countermeasures typically require additional hardware support or distance-bounding techniques. Since such defences introduce substantial complexity and deployment overhead, CARPOOL does not attempt to address it.

Attack scenarios. An adversary who completes the primary authentication step and receives an encrypted token at time T may still attempt to defeat the location proof. We consider these scenarios:
(1) Remote attacker at authentication time (T). The adversary is outside the room at time T but attempts to obtain the required shares remotely, for example using long-range cameras for visual channels, directional microphones for audio, or high-gain antennas for BLE advertisements. CARPOOL mitigates this by enforcing multi-device interaction thresholds, requiring simultaneous compromise of multiple independent location-bounded channels within the same protocol run.
(2) Adversary with prior physical access ($P < T$). The adversary previously accessed the room and leaves behind an eavesdropping device to capture shares at time T. Some channels (e.g., BLE or audio) are more susceptible to such placement than others (e.g., visual), which typically require line of sight and careful positioning. CARPOOL mitigates this by requiring combinations of channels, increasing operational complexity and detection probability. Alternatively, the adversary may tamper with devices to extract cryptographic material. We assume deployments apply physical and platform protections (e.g., secure key storage and tamper resistance) and that compromise of room devices is detectable over time, enabling administrators to update policies and revoke affected devices.
(3) Adversary relocating devices ($P < T$). An adversary with prior access may relocate a device to enable remote authentication at time T. This is mitigated through physical security controls and policy updates upon detection. Since CARPOOL relies on multiple devices, relocating a single device is insufficient to satisfy the location proof under multi-device policies.
(4) Physically present attacker at authentication time (T). If the adversary is physically present in the room at time T, CARPOOL will not prevent completion of the location proof, as the mechanism is designed to attest presence. This corresponds to a failure of physical access control or credential security and must be addressed through complementary controls (e.g., strong primary authentication, physical access restrictions, monitoring).

Threshold assumption. CARPOOL increases attack cost by enforcing a minimum number of required device interactions defined by a threshold l. We assume an external adversary can practically compromise at most y devices or channels during a single protocol run, where $y < l$. This captures the intended regime in which large-scale, simultaneous compromise is infeasible.

4 CARPOOL scheme

4.1 Protocol flow

The CARPOOL protocol, illustrated in Figure 2, operates in six phases: (1) authentication, (2) authorisation, (3) encryption and generation of secret shares, (4) collection of online shares, (5) reconstruction of the key and decryption, and (6) verification and access to the resource. The circled numbers in the text correspond directly to the steps in Figure 2. Before discussing these phases in detail, we first describe the one-time setup process required for the local devices.

Setup phase. Before deploying CARPOOL, each local device must be initialised. Vendors install firmware to make devices CARPOOL-compliant, either during production or via firmware update. For newly produced devices, an initialisation key is embedded at manufacture and provided with the device (e.g., in the manual or as a QR code). The system administrator uses this key to configure the device and provision keying material, such as a unique secret offline key shared with the LPEM (for offline share generation) or a TLS key pair with certificate. For existing devices, secure configuration depends on the available interfaces.

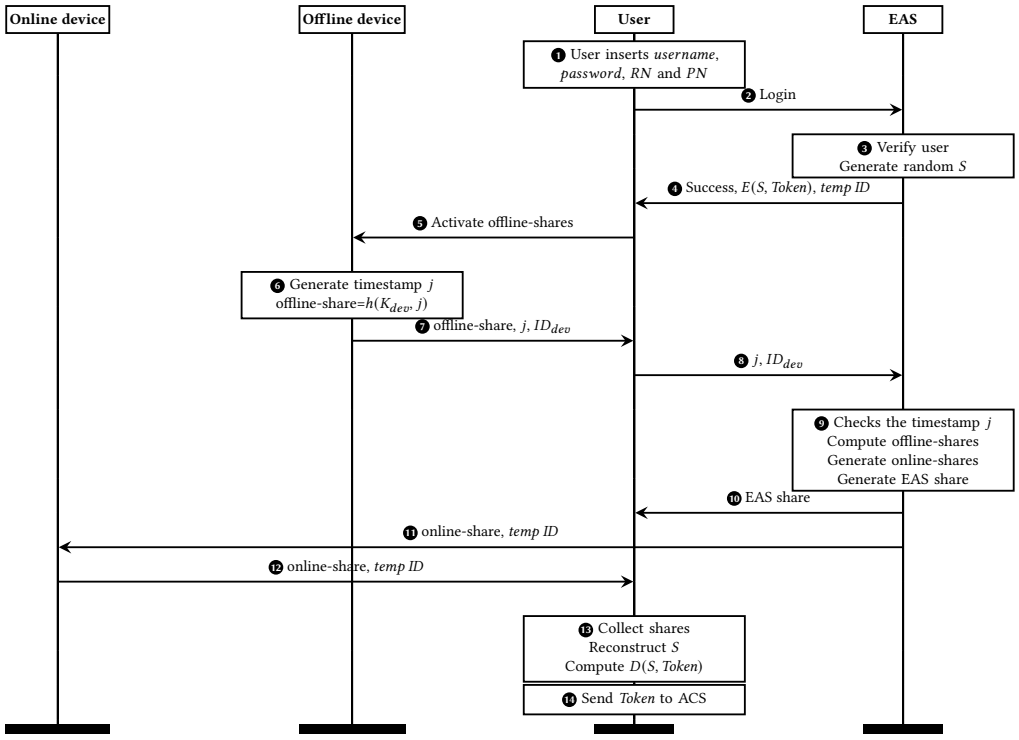


Figure 2: Message diagram of CARPOOL

When a new device is added, the LPEM is updated with its parameters, including IP and MAC address (for online devices) or offline key (for offline devices), physical location, and supported location-bounded interface.

Authentication phase. The protocol begins when the user inserts *username*, *password*, room number (RN), and policy number (PN) ❶ and sends a login request to the EAS ❷.

The AS verifies the user credentials ❸, while RN and PN are forwarded to the LPEM. RN denotes the claimed room. Each room is associated with one or more security policies defining valid device interaction combinations. PN specifies the selected interaction combination.

Authorisation phase. Upon successful verification, the AS generates a freshly signed JSON access token for the authenticated user and a random session key S is generated ❸. The EAS returns the encrypted token $E(S, Token)$ together with a temporary identity *temp ID* ❹. The *temp ID* distinguishes concurrent sessions while concealing the username from passive observers.

Encryption and generation of secret shares. After receiving the encrypted token, the user activates the offline device(s) via the location-bounded interface ❺. Each offline device generates a timestamp j and computes its offline share as $h(K_{dev}, j)$ ❻. The offline share, together with j and ID_{dev}, is transmitted to the user ❼. The user forwards j and ID_{dev} to the EAS ❽.

The EAS checks the freshness of j and computes the corresponding offline shares ❾. It then generates the remaining additive shares (online shares and its own share) according to the selected policy. The EAS sends its own share to the user ❿ and distributes the online shares to the respective online devices together with the *temp ID* ⓫. The online devices return the online shares to the user via their location-bounded interfaces ⓬.

At this stage, the user holds the encrypted token, offline shares, the EAS share, and the online shares required by the selected policy.

Reconstructing the key and decryption. The user collects the required shares and reconstructs the session key S ⓭. Using S, the user decrypts $E(S, Token)$ and obtains the plaintext token.

Verification and access to the resource. Finally, the decrypted token is sent to the ACS ⓮. The ACS validates the token and, if authentic and unexpired, grants access to the protected resource; otherwise, access is denied.

4.2 Additive secret sharing

4.2.1 Background. Our protocol employs an additive secret-sharing based threshold scheme formulated using the disjunctive normal form (DNF). DNF is a logical structure composed of a disjunction of conjunctions, that is, an OR of ANDs. Let $\mathcal{P}$ denote the set of n players, $\mathcal{P} := \{P_1, P_2, \cdots, P_n\}$, representing both online and offline devices in our setting. The goal is to distribute a secret key S among these players such that only specific authorised combinations can reconstruct S. The access structure, denoted Γ, defines these authorised subsets. The DNF expression specifies the exact

conditions under which the secret key S can be reconstructed by the participating 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 $s_{2,j}$. 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 relies on local interaction and communication between the mobile client and nearby devices within the smart office room. Each device operates through a specific location-bounded communication channel. For the solution to remain practical and deployable, these channels must be easy to use, widely supported by mobile clients, and offer a reasonable level of security (that is, an adversary should find it non-trivial to exploit the channel from outside the room). Furthermore, the chosen channel should support secure key transport, such as the exchange of secret shares.

5.1 Design requirements

Before selecting suitable communication channels, it is necessary to define the design requirements for location-bounded communication channels based on CARPOOL's needs. Drawing inspiration from Bonneau et al.'s authentication framework [8], these requirements are grouped into five categories: Security (S), Usability (U), Reliability (R), Communication (C), and Deployability (D). Further details are provided in Appendix A.

5.2 Candidates for location-bounded communication channels

The attributes outlined earlier also apply to location-bounded communication channels, each characterised by distinct SURCD (Security, Usability, Reliability, Communication, Deployability) properties. This paper considers four primary types suitable for deployment in office environments: (i) visual, (ii) acoustic, (iii) light intensity, and (iv) short-range RF, such as Bluetooth Low Energy (BLE).

Device types and user interactions. In CARPOOL, each channel corresponds to a concrete class of commodity room devices and a clearly defined client-side interaction. Table 1 summarises representative device types and the exact user action required to obtain a secret share. Any COTS device that can (i) securely store a device secret or receive an online share, and (ii) output data over the corresponding interface, can instantiate the channel.

No single channel satisfies all five macro-design requirements simultaneously. Some offer stronger confinement but reduced reliability, whereas others are easier to deploy but provide weaker physical containment. The following paragraphs analyse these four channel classes in terms of their practical and security properties.

Visual. This channel embeds shares within QR codes or barcodes displayed on screens. The user scans the code using the smartphone camera. Visual channels are deterministic and therefore highly reliable, with a low rejection rate. They are easily deployable due to the widespread availability of displays in meeting rooms. A limitation is the possibility of long-range capture using high-resolution optical equipment.

Acoustic. In this channel, shares are modulated onto frequencies receivable by smartphones, including near-ultrasonic bands (≥18 kHz). Transmission requires only speakers, and collection requires activating the smartphone microphone. Acoustic channels provide high bandwidth and minimal manual interaction. However, they are probabilistic and require careful encoding to balance reliability and bit rate. Their penetrative nature increases interception risk, and vibrations may be recoverable using advanced remote sensing techniques [11].

Light. Light intensity modulation allows secret shares to be transmitted through controllable light sources. The user captures the modulated signal using a smartphone camera. This channel offers high bandwidth and minimal additional hardware requirements, but shares acoustic limitations: signals may propagate beyond the intended space, ambient lighting may interfere, and rapid intensity modulation may be unsuitable for sensitive users.

Bluetooth. Short-range RF communication via BLE supports either passive beacon reception or paired encrypted communication.

Table 1: Mapping from channel type to representative commodity devices and the corresponding user interaction.

Channel	Example room devices that can provide a share	User interaction to collect the share
Visual	Projector or meeting room display, TV screen, digital signage, conference room monitor, tablet used as a screen	Scan a QR code or barcode shown on the screen using the smartphone camera
Acoustic	Conference speaker system, smart speaker, laptop speakers, TV speakers, meeting room soundbar	Press a record button in the app and capture the (near-)ultrasonic signal with the smartphone microphone
Light intensity	Smart light bulb or LED panel, controllable desk lamp, display backlight, phone camera flash controlled by a room device	Point the smartphone camera at the light source to capture intensity modulation, or place the phone within line of sight of the emitter
Short-range RF (BLE)	BLE beacon, smart lock, smart plug, IoT hub, conference room dongle, badge reader with BLE	Either passively receive a beacon advertisement in the app, or perform an initial pairing step and then exchange encrypted BLE messages

Beacon mode requires no explicit pairing, whereas paired mode involves an initial setup step but provides encrypted transport. BLE is widely supported and easy to deploy, yet unpaired advertisements remain susceptible to long-range interception using specialised equipment.

Although none of these channels alone fulfils all SURCD properties, CARPOOL allows users or administrators to combine channels such that the resulting policy collectively satisfies the required security and reliability constraints.

6 Proof-of-concept implementation

We developed a full, end-to-end, working prototype of CARPOOL to demonstrate its feasibility and ease of integration into existing security infrastructures, including commercial Identity and Access Management (IAM) systems.

6.1 Our demonstration setup

Our demonstration setup, illustrated in Figure 3, validates CARPOOL's concept and practical applicability for location-based authentication. As described in Sect. 2.1, the system comprises a cloud-hosted back-end, a user's mobile client, and local devices, including both online and offline types with distinct communication interfaces. Online devices connect to the back-end via the meeting room's Wi-Fi, as summarised in Table 2. We now describe each entity in our demonstration setup in detail:

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 [39].

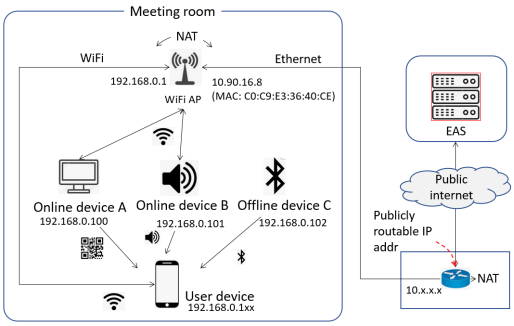


Figure 3: Network topology of demonstration setup

- LPEM: We utilised an AWS EC2 instance with Ubuntu 18.04 for flexibility and scalability, leveraging Amazon EC2 for simplified setup and capacity management. 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.
- 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.
- DB: For ease of implementation, we used an off-the-shelf AWS RDS MySQL database instance, an isolated database environment running in the cloud.

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 now describe the CARPOOL protocol's implementation, beginning with the back-end server and mobile client, followed by the online and offline devices and their role in location-bounded communication.

Table 2: Entities in our proof-of-concept

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

During protocol execution, users are guided through login and subsequent device interactions via specific location-bounded interfaces. These steps aim to (i) improve usability by clarifying required actions and (ii) reduce user errors during authentication.

6.2.1 Authentication phase. As detailed in Sect. 4, the client begins by logging into the EAS and specifying the room and policy numbers. After entering the room number, the mobile client retrieves that room's policies from an Amazon Simple Storage Service (Amazon S3) bucket,* enabling straightforward access to security policies for any office room. Only administrators are authorised to modify these policies. For ease of selection, devices in each 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 defining their access level, linked to the ID token from the AS, both sharing the same validity period. AWS Cognito was selected for the AS and ACS due to its seamless integration and support via the `boto3` Python library. The LPEM intercepts all communications between the mobile client and the EAS, verifies room and policy numbers, and rejects invalid requests. After successful authentication, AWS Cognito issues the ID and access tokens.

6.2.3 Encryption and generation of secret shares. After issuing the user's tokens, the LPEM intercepts them, generates a 128-bit key S, and encrypts the tokens using AES-CBC with a fixed salt via the Fernet library. It sends $E(S, Token)$ and a temporary identity ($temp\ ID$) to the mobile client. The LPEM then awaits the user's offline-share data, identified by ID_{dev} and timestamps, verifies these timestamps, computes the corresponding offline shares, and determines active online devices based on the selected security policy. For each device, it generates a random 128-bit share along with its own. In our setup, with one offline and two online devices, 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 executed solely by the mobile device. The Android client interacts with the location-bounded interfaces of online and offline devices using native audio (`MediaRecorder` and `AudioRecord`) and Bluetooth libraries, along with the Budiyev-QR scanner for data capture. The smartphone's camera, microphone, and Bluetooth components are therefore essential to our prototype. After collecting all shares relevant to the chosen policy and verifying them via the temporary identity, the client aggregates them to reconstruct S. Using S and a hard-coded salt, the token is decrypted and transmitted to the EAS to access database resources.

6.2.5 Verification and access to the resource. In this phase, the Extended Authentication Server (EAS) receives the ID and access tokens from the mobile client. AWS Cognito performs token verification and validation. If successful, database access is granted. This process employs the `warrant` and `pycognito` Python libraries, which required no modification for our implementation.

*https://docs.aws.amazon.com/s3/index.html#amazon-s3

6.3 Implementation details: location-bounded communication channels

6.3.1 Online device with visual communication channel. In our demonstrator, the visual communication device operates online, receiving a 128-bit share from the LPEM. The setup defines the visual device as a UDP server and the LPEM as a UDP client, enabling the LPEM to trigger share computation and transfer. Consequently, the online device can receive shares whenever required. Upon receipt, it converts the share into a QR code displayed on its screen, which the user scans with a smartphone to obtain the online share.

6.3.2 Online device with audio communication channel. The second online device employs an audio channel on a Raspberry Pi 3 Model A+. A 128-bit share is transmitted to the mobile client using near-ultrasonic frequencies ($f_c \geq 18$ kHz), which are inaudible to most users, reproducible by speakers, and detectable by smartphones. The share is modulated via on–off keying (OOK), using the carrier wave's amplitude to encode the secret share, as shown in Fig. 4. Rather than developing a complete signal-processing pipeline, we relied on Android's native audio libraries. With a carrier frequency of 18 kHz, a sampling rate f_s of 44.1 kHz, and a 10-second window, we modulate the digital data (share s) into an analogue wave ($s \sin(2\pi f_c t)$, where $t = 1/f_s$). Each bit of s is repeated 3330 times, forming a $(1, 3330)$ repetition code, to maximise reliability, eliminating errors and false positives. The smartphone's microphone then captures and demodulates the signal to recover s.

6.3.3 Offline Bluetooth beacon device. The third device, an offline Bluetooth interface on the same Raspberry Pi model as the online devices, does not connect to the LPEM for share reception. It generates an HMAC-SHA-256 of the timestamp as its offline share to prevent replay, requiring only loose synchronisation with the LPEM's clocks. The share comprises the timestamp and its HMAC, using a key pre-shared with the LPEM. Unique device IDs in the QR code differentiate shares from multiple offline devices.

We use the *Eddystone URL* message [46] to broadcast offline shares, embedding the lowest 10 octets of the HMAC output owing to payload limits. On receipt, the mobile client extracts the share and forwards it to the LPEM.

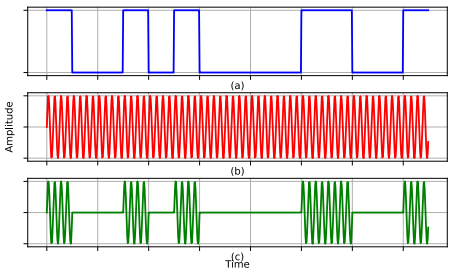


Figure 4: The audio encoding process – (a) the binary data (digital wave) to be encoded, (b) the carrier (analogue) wave, and (c) the modulated OOK wave (analogue signal) which is being played by the speakers.

Unlike online devices triggered by the LPEM, the offline device is manually activated via a Raspberry Pi touchscreen to emit a new BLE beacon. While secure, truncation slightly weakens protection compared with online devices. A standard Bluetooth connection could improve security but would increase user interaction.

7 Evaluation and Discussion

Our CARPOOL solution satisfies the design requirements outlined in Sect. 5.1. It fulfils the **deployability** criterion by leveraging personal devices (e.g., smartphones) and commodity smart devices that are already present in modern office environments, without requiring any dedicated hardware tokens, anchors, or specialised ranging equipment. Importantly, CARPOOL does not impose additional per-device hardware costs: the required communication interfaces (camera, speaker, microphone, display, Bluetooth) are already integrated into existing devices and can be utilised through software configuration or firmware updates. Consequently, deployment primarily involves provisioning cryptographic material and enabling the relevant interfaces, rather than purchasing new physical infrastructure. Compared with approaches that rely on dedicated hardware (e.g., specialised distance-bounding radios or UWB anchors), CARPOOL avoids incremental capital expenditure per room and scales with existing device ecosystems. Additionally, CARPOOL imposes no constraints on device placement for protocol correctness: devices may be moved or interchanged within the room, and the protocol continues to function accurately in the presence of dynamic objects or people.

Efficiency and usability trade-offs. CARPOOL is policy driven, and therefore its end-to-end completion time depends on the number and type of required device interactions. We evaluate two contrasting configurations that make this trade-off explicit. **Scenario A** prioritises usability by selecting a single automatic BLE beacon interaction. Over 10 trials, the median end-to-end authentication time was **1.3 seconds**. This configuration is intentionally lower assurance because it relies on a single location-bounded channel, but it is almost automatic and requires no explicit user action beyond initiating the session. **Scenario B** prioritises security by selecting the strongest policy in our prototype, combining three independent channels (visual QR, ultrasonic audio, and BLE beacon). Over the same 10 trials, the median end-to-end authentication time was **14.8 seconds**, dominated by the fixed 10 second audio capture window. Together, these scenarios explain the observed range of **2 to 15 seconds** and with 100% success rate (no false positives).

Usability with multiple device interactions. Beyond timing, CARPOOL's usability stems from the fact that additional security is realised by composing familiar, low-effort actions rather than introducing new credentials or specialised hardware. In Scenario A, the user experience is near-transparent because the BLE beacon can be received passively. In Scenario B, the user performs a small sequence of simple actions already common in mobile applications: scanning a QR code and pressing a single button to record an audio signal, while BLE reception remains automatic. Crucially, the policy knob changes the *number* of such micro-interactions, but not their complexity, and the smartphone application guides the user step-by-step to reduce confusion and interaction errors. In this sense, CARPOOL increases assurance primarily through composition and orchestration, rather than through cognitively demanding tasks.

Across both scenarios, the computational overhead is minimal: online devices transmit or display only a 128-bit share, offline devices perform a single hash operation, and each device requires fewer than 100 lines of code. The mobile client reconstructs the key by addition and performs one decryption. Such operations are trivial even for battery-powered microcontrollers, though our model primarily targets commercially available, AC-powered devices. Most computational overhead resides on the server, which in our case runs on Amazon AWS.

Reliability. Although the audio channel is probabilistic, the $(1, 3330)$ repetition code yielded **no errors** in our trials, even in noisy environments. Across the 10 trials for each scenario, we observed a **100% acceptance rate** with **no false positives or false negatives**. **Communication and reliability** are further discussed in Sect. 5 and Appendix A. Regarding **security**, CARPOOL integrates several protective mechanisms, elaborated 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, and revoke malicious hardwares as well from the policy tree.
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 does not 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 brute-forcing

one ciphertext requires 2^{128} trials. Reconstructing the session secret demands that an adversary compromise *all l* independent, location-bounded channels *within the same protocol run*, capturing the on-screen QR code, relaying BLE advertisements, and eavesdropping ultrasonic audio, before the per-run timestamp window (Sect. 3) expires. This simultaneity requirement expands the effective attack space to $2^{128\times l}$, making relay or guessing attacks economically infeasible even for well-resourced adversaries. Dormant (offline) devices might face invasive tampering, such as clock manipulation, but CARPOOL can simply exclude them: the policy engine may rely solely on *online* devices whose freshness is verified at run time, or use several channels from the *same* device to enhance robustness[†]. If a device is later compromised, the administrator revokes it immediately; the additive threshold still holds for other trusted combinations, ensuring uninterrupted authentication and forward secrecy for past keys.

8 Related work

Location-aware authentication solutions rely on contextual information to infer or prove a user's physical presence. The concept was pioneered by Denning and MacDoran [12], who proposed location as an authentication factor. Since then, numerous approaches have been explored. Appendix A defines the design requirements considered in this work, and Table 3 (Appendix) compares representative systems. As shown there, prior work typically optimises one or two dimensions (security, usability, or deployability) but does not satisfy all simultaneously. CARPOOL is the only approach that simultaneously achieves S1–S8, U1–U6, R1–R2, D1–D3 under realistic indoor deployment assumptions.

GNSS. A straightforward approach is to rely on GNSS technologies[‡] such as GPS or Galileo using smartphone or laptop receivers [5, 9, 13, 16, 36]. However, GNSS-based authentication is vulnerable to signal spoofing [40], undermining strong cryptographic protection (S1), and performs poorly indoors, limiting practical deployability (D1–D3). Marforio et al. [23] proposed signing GPS coordinates using a TPM to prevent compromised devices from forging locations. Nevertheless, this does not prevent spoofing at the signal level, fails indoors, and increases execution overhead, affecting usability (U4).

Hardware-based security keys. Installing dedicated USB or NFC security tokens per room provides a simple second factor. However, such devices have been shown to suffer from security weaknesses [31, 32, 44] and reliability issues [10, 17]. More importantly, they require dedicated hardware per room, increasing deployment cost and reducing scalability (D2, D3). They also introduce a single point of failure: if the token is stolen or malfunctions, authentication halts. In contrast, CARPOOL allows policy reconfiguration using existing devices.

Context-based pairing. Numerous pairing schemes derive keys from contextual similarity, using acoustic [37], visual [25, 34, 43], gyroscopic [21, 24], touch [47], or vibrational [2, 35] channels. SoundProof [20] leverages ambient audio and claims minimal interaction, yet Shrestha et al. [41] demonstrated security flaws, and Basin et al. [7] highlighted usability weaknesses due to human-factor errors. Visual-channel techniques are susceptible to long-range camera capture [33], weakening physical observation resistance (S5). Miettinen et al. [26] combine ambient sound and light, but require substantial error correction due to probabilistic channels, increasing latency and reducing usability (U4). Vibrational channels remain vulnerable to eavesdropping attacks [3, 18]. Gesture- or touch-based schemes [21, 22, 24, 47] improve user interaction but often require additional or proprietary hardware, limiting deployability (D1–D3). Agadakos et al. [1] combine user interaction and IoT sensors for faster authentication, yet physical locations remain spoofable and error rates remain non-negligible. Perceptio [19] relies on temporal correlation across devices, but requires repeated protocol executions over hours or days and fixed device positioning, significantly impairing usability (U4) and practical deployability.

Distance bounding. Distance-bounding protocols combine cryptography with time-of-flight measurements to establish upper distance bounds. ZeroTouch [4] performs geofencing using RSS without additional hardware, but exhibits false acceptance and rejection rates that affect reliability (R1) and usability. Move2-Auth [48] correlates RSS traces during gestures, reducing usability when repetitions are required. Other approaches rely on RF or Wi-Fi signals [30, 45]. Rasmussen *et al.* [29] proposed ultrasound-based distance bounding, later shown vulnerable to wormhole relays [38]. While UWB radios [30] and Wi-Fi CSI rigs [45] can mitigate relay attacks, they require specialised hardware and professional installation, increasing deployment cost and limiting scalability (D2, D3).

Key take-away. *Existing solutions either require dedicated hardware, incur significant error rates, depend on environmental assumptions, or degrade usability through repeated interactions. CARPOOL differs by leveraging devices already present in the environment, requiring only software provisioning, and enabling configurable trade-offs between security and usability without additional capital expenditure.*

9 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.

[†]Selecting multiple channels per device is optional and adds only one extra share transmission.

[‡]https://www.gps.gov/systems/gnss/

Acknowledgement

This work was supported by the Flemish Government through the Cybersecurity Research Program, grant number VOEWICS02.

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*.

[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)*.

[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 ACM on Asia Conference on Computer and Communications Security (ASIA CCS)*.

[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*.

[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).

[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, 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)*.

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

[16] 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).

[17] 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)*.

[18] 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)*.

[19] 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)*.

[20] 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)*.

[21] 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)*.

[22] 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)*.

[23] 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*.

[24] Rene Mayrhofer and Hans Gellersen. 2009. Shake Well Before Use: Intuitive and Secure Pairing of Mobile Devices. *IEEE Transactions on Mobile Computing* (2009).

[25] 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)*.

[26] 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)*.

[27] 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)*.

[28] 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*.

[29] 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)*.

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

[31] Thomas Roche. 2024. EUCLEAK. Cryptology ePrint Archive, Paper 2024/1380.

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

[33] 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)*.

[34] 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)*.

[35] N Saxena, Md. Borhan Uddin, J Voris, and N. Asokan. 2011. Vibrate-to-unlock: Mobile phone assisted user authentication to multiple personal RFID tags. In *IEEE International Conference on Pervasive Computing and Communications*.

[36] 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)*.

[37] Dominik Schürmann and Stephan Sigg. 2013. Secure Communication Based on Ambient Audio. *IEEE Transactions on Mobile Computing* 12, 2 (2013), 358–370.

[38] Sahar Sedighpour, Sridjan Capkun, Saurabh Ganeriwal, and Mani Srivastava. 2005. Implementation of Attacks on Ultrasonic Ranging Systems. In *SenSys*.

[39] Amazon Web Services. 2021. AWS. https://docs.aws.amazon.com/.

[40] 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*.

[41] 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)*.

[42] 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)*.

[43] 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)*.

[44] 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)*.

[45] 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)*.

[46] Wikipedia.org. 2021. Eddystone (Google). https://en.wikipedia.org/wiki/Eddystone_(Google).

[47] 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)*.

[48] Jiansong Zhang, Zeyu Wang, Zhice Yang, and Qian Zhang. 2017. Proximity based IoT device authentication. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*.

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

Design		GNSS / GPS					token-	Context-based pairing														Distance bounding/RSSI					CARPOOL
Requirements		[5]	[9]	[13]	[16]	[36]	ens	[1]	[2]	[19]	[20]	[21]	[22]	[24]	[25]	[26]	[34]	[35]	[37]	[43]	[47]	[4]	[29]	[30]	[45]	[48]	[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	●	●	●	●	●	●	●	●	❍	●	●	●	●	●	●	●	●	●	●	●	●	❍	❍	❍	❍	●

A Design requirements

CARPOOL, designed to authenticate user location, adopts Bonneau et al.'s framework [8], a standard for evaluating authentication protocols [14, 17, 20, 42]. Here's a summary of its design requirements and CARPOOL's compliance.

A.1 Security properties

Strong cryptographic protection (S1): CARPOOL ensures robustness against cryptographic attacks, minimising false acceptance and achieving 128-bit security. Eavesdropping in location-bounded channels is costly.
Revocation (S2): Access revocation for clients should be possible.
Isolation (S3): An authenticated device remains unaware of keys and shares between the server and other devices, past and future.
Availability (S4): Availability should be constant, enabling it to manage device failures or malfunctions within the room.
Resilient-to-Physical-Observation (S5): Observing interactions must not enable adversary impersonation or authentication.
Resilient-to-Internal-Observation (S6): Intercepting another client's input or eavesdropping on room device communications must not enable impersonation or authentication.
Resilient-to-Guessing (S7): Adversaries should not bypass security by guessing low-entropy secret variables.
Requiring-Explicit-Consent (S8): Starting the authentication process should require the client's explicit consent, ensuring security and privacy.

A.2 Usability properties

Limited user participation (U1): Minimise client-device interactions, prioritising automation (e.g., BLE beacons) and limiting manual tasks (e.g., camera scanning) to necessity.
Easy-to-Learn (U2): Users unfamiliar with the scheme should quickly grasp, learn, and recall its use without significant difficulty.
Easy-to-use (U3): Actions should be straightforward for users without special training, like QR code scanning versus mimicking gyroscopic movements.
Efficient-to-Use (U4): Authentication time should be short, with acquiring device shares via location-bounded channels also quick.
Memory-wise effortless (U5): Clients shouldn't need to recall, passwords, keys, or shares beyond initial login details for the Authentication Server.
Nothing to carry (U6): Clients should authenticate using just their personal devices, e.g., smartphones, with no extra hardware. Office devices must function similarly.

A.3 Reliability properties

Low Rejection (R1): The CARPOOL scheme should have a low false rejection rate to limit repeated authentication attempts.
Infrequent-Errors (R2): CARPOOL should reduce errors and accurately manage them. Channels vary in reliability; noise and error correction can affect the speed of 128-bit share transmission.

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): Supports BYOD, enabling standard smartphones or tablets without additional hardware.
Scalability (D2): Should enable share transport with a few devices and easily expand to include more for location proof.
Lightweight (D3): Should work with all common office devices, from smart-projectors to BLE beacons, ensuring CARPOOL is lightweight.