Final Project Report

Security Design for a Smartphone Car Access App

May 8th, 2025

Group 9

Besar Jukaj, Francisco Alves, Stefanos Panagoulias,
Leonardo Ratti, Furkan Poyraz, Maurizio Perriello

Contents

1. Assumptions

The architecture of the app has been designed taking into account as if it were correlated to a single car manufacturer. The app will be able to interact only with the cars of that manufacturer brand that are in line with the standards required by it.

2. Introduction

2.1 Contextual Background

Over the past two decades, car access technology has evolved from traditional mechanical keys to electronic smart cards, offering increased convenience and enhanced security. However, the rise of smartphones as central devices in personal digital ecosystems has sparked a new shift in automotive access — moving toward smartphone-based applications as digital car keys. These apps allow users not only to unlock and start their vehicles but also to manage access rights and monitor car usage, all from a single device.

2.2 Industry Trends

Automotive industry leaders such as BMW, Tesla, and Hyundai have introduced digital key systems compatible with smartphones. According to a 2024 Statista report, the smartphone penetration rate in Europe is projected to reach approximately 82. The Car Connectivity Consortium (CCC) has also published standards (e.g., CCC Digital Key) to support interoperable and secure smartphone-based car access. Notably, the CCC surpassed 300 member companies in 2024, reflecting significant industry commitment to standardizing digital key technology.

2.3 Transition Justification

This transition is driven by multiple factors:

- **Ubiquity of Smartphones**: Users already carry smartphones daily, reducing the need for additional devices.
- **Security Capabilities**: Smartphones support advanced security features such as biometrics and secure enclaves.
- **Update Flexibility**: Unlike smart cards, mobile apps can be patched and updated remotely.
- **Remote Management**: Digital car keys can be shared, revoked, or restricted from anywhere with an internet connection.

Feature	**Smart Cards** vs. **Smartphone Apps**
Convenience	Extra device needed vs. Built-in on phone
Authentication	Single factor vs. Multi-factor (biometrics, PIN)
Access Sharing	Physical handover vs. Digital delegation
Security Updates	Static vs. Remote updates
Battery Dependency	Passive, no battery vs. Phone battery
Backup Options	Minimal vs. Cloud-based recovery
Attack Surface	Low vs. Broad (OS, apps, wireless)
Cost	Extra hardware vs. Uses existing phone
Replacement	Needs physical swap vs. Reinstallation

Table 1: Smart Cards vs. Smartphone Apps

3. Security Requirements

Key security requirements Authentication, Confidentiality, Data Integrity, Non-Repudiation and Availability.

- Availability is very important one for something like accessing your car. It would be a pity if you need to walk to your destination because the servers are down.
- Authentication is needed such that only authorized users can access the car.
- Confidentiality is important such that the communication is protected against eavesdropping.
- Non-repudiation is important to consider to allow the identification of the driver when a fine for some traffic violation must be paid or when an accident occurs.
- Data integrity ensures that any changes to messages during communication will be noticed.

4. Authentication Mechanisms and Access Control

4.1 Multi-Factor and Risk-Based Authentication

To ensure that only authorized users can access the car, the system employs a robust multi-factor authentication (MFA) approach. This includes three distinct factors, and an additional risk-based layer:

Factor	**Description**	**Example**	**Optional?**
Something you have	Device-bound identity verified at registration	Unique smartphone ID	No
Something you are	Non-transferable biometric traits	Face ID / Fingerprint	No (if supposrted)
Something you know	Backup authentication method	6-digit fallback PIN	No (backup only)
Risk-based layer	Adjusts auth requirements based on context	Location, behavior analysis	Yes (adaptive)

Table 2: Multi-Factor and Risk-Based Authentication Layers

The app implements several safeguards, like PIN recovery, lost device handling, to make fallback, in cases of verification malfunctions, secure and user-friendly:

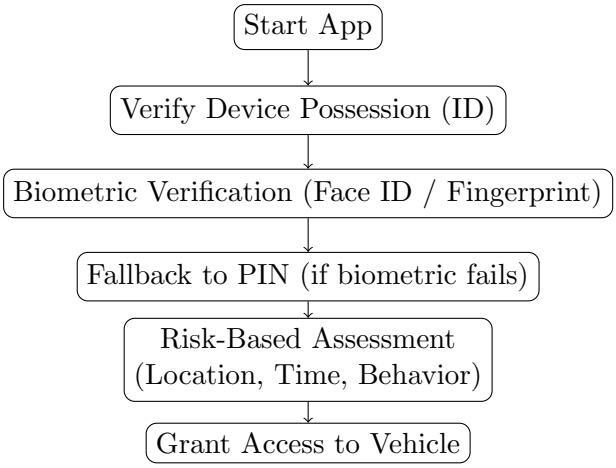

Figure 1: Simplified Authentication Flow for Vehicle Access

In case the smartphone does not support biometric authentication, the app will be access directly without the need of a password.

4.2 Application-Level Security Protections

Beyond secure communication protocols like TLS, the app implements additional protections to maintain security and user trust. It manages sessions carefully to require re-authentication after inactivity, prevents replay attacks using temporary codes, and securely encrypts sensitive data stored on the smartphone.

4.3 Access Revocation Mechanisms

We present two revocation systems:

- **License-based**: Periodic checks; gradual restrictions and full revocation if the license is suspended. Every time the user access the car, the server checks for possible license restrictions.
- **Emergency revocation**: Remote kill switch through web portal; automatic triggers upon suspicious behavior.

4.4 Secure Access Sharing

Car owners can share access through digitally signed delegation tokens containing recipient ID, validity period, and usage scope. Tokens are revocable at any time by the owner. The *Access Sharing Workflow* proceeds as follows:

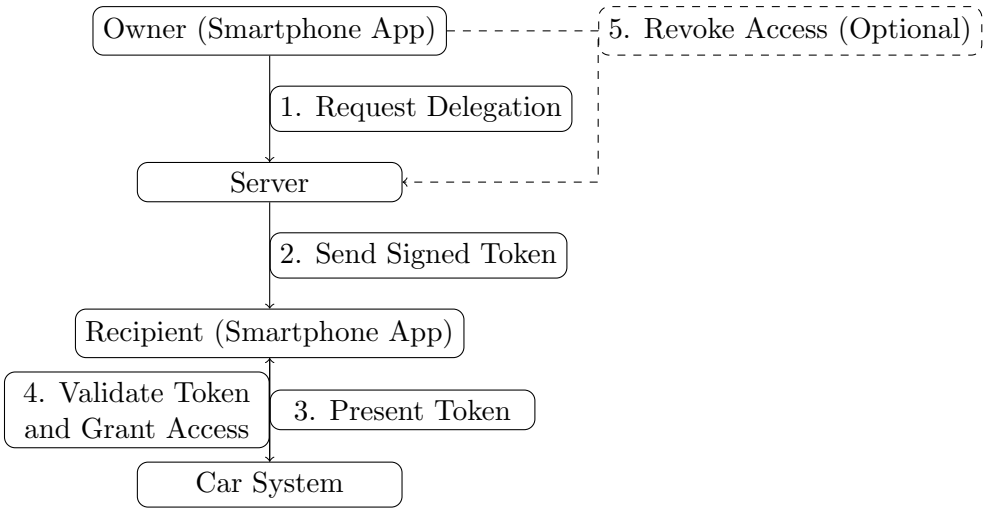

Figure 2: Access Sharing Flow: From Owner to Recipient

Considerations:

- The delegated user will be able to access the functionalities only with a stable internet connection.
- The owner can share one key at the time.

4.5 Driver Identification

Whenever the car is turned on while a shared token is active, it will log in the server the identity of the driver. The identity is retrieved using BLE or UWB. The phone closer to the driver seat will be used as reference.

In order for the authorities to retrieve the driver's identification, a formal request shall be sent to the app management.

4.6 Multi-Device Management

The system supports users accessing their account and managing vehicles from multiple personal devices, ensuring convenience without compromising security.

Core Principles

- **Individual Device Identity & Authorization:** Each device is uniquely identified and must be explicitly authorized by the user before use.
- **Independent Cryptographic Keys:** Each authorized device generates and stores its own cryptographic keys within its secure hardware (TEE/Secure Enclave).
- **Centralized Management & Revocation:** Users can view and revoke access for any authorized device via the app or a secure web portal.

Adding a New Device

1. **Account Login:** User installs the app on the new device and logs in with existing account credentials.
2. **Multi-Factor Authentication (MFA):** Server initiates an MFA challenge, preferably via a push notification to an already trusted device, or alternatively via OTP to a verified contact method.

3. **Device Authorization & Key Generation:** Upon successful MFA, the new device generates its unique keys (private key in TEE). The public key certificate is sent to the server and associated with the user's account.

Device Revocation

Users can manage their list of authorized devices through the app (from any trusted device) or a secure web portal.

- **Standard Revocation:** From the device list, the user selects a device and revokes its access. The server invalidates the device's keys and sessions.
- **Revocation of a Sole Stolen Device:** If the only authorized device is lost/stolen, the user must:
  i. Access a secure web portal using their primary account credentials and a non-device-dependent MFA method (e.g., recovery codes, OTP to verified email/SMS, personal questions).
  ii. Navigate to the device management section and revoke the stolen device.
  iii. The server immediately invalidates the device's credentials.
  iv. The user can then authorize a new device following the "Adding a New Device" procedure.

Security Considerations

- Robust MFA for account login and new device authorization is critical.
- Secure storage and handling of recovery codes (if used) are essential for sole device loss scenarios.
- The web portal must be protected against phishing and require strong authentication.

5. Secure Communication

A smartphone application replacing a smart card for car access must implement robust security mechanisms to ensure safe and reliable communication between the smartphone and the car. This includes protection against data interception, replay attacks, and unauthorized access.

5.1 Secure Communication Between Smartphone and Car

The app and the car must communicate over a secure channel to prevent eavesdropping, man-in-the-middle attacks, and data tampering. The following security measures should be implemented through the TLS (Transport Layer Security) protocol and beyond:

- **TLS 1.3:** Ensures end-to-end encryption, integrity, and authentication of messages. TLS 1.3 is recommended because it eliminates weaker encryption algorithms and reduces the risk of attacks.
- **Mutual Authentication:** Both the car and the smartphone must authenticate each other using digital certificates (e.g., X.509 certificates).
- **Elliptic Curve Cryptography (ECC):** Given the constraints on power and computation in embedded systems, ECC provides strong encryption with lower computational overhead compared to RSA.
- **Session Key Establishment:** Secure key exchange using Diffie-Hellman (ECDH) ensures that session keys are not transmitted over the air.

- **Perfect Forward Secrecy (PFS):** Ensures that even if the session key is compromised, previous communications remain secure.
- **Communication with Server** Both the car and user app have to communicate securely with the server. In this case, U and C will authenticate the server through certificate authentication using a CA. The rest of the communication between these entities will be carried out just like the communication car-user described in the following points.

5.2 Protection Against Attacks

To ensure that attackers cannot intercept or manipulate communication between the smartphone and the car, multiple security layers must be implemented:

- **Protection Against Interception (Eavesdropping and MITM)**
  - **Encrypted Communication:** All communication must be encrypted using strong cipher suites (e.g., AES-GCM with 256-bit keys).
  - **Certificate Pinning:** The app should verify that it is communicating with the genuine car system by pinning the car's digital certificate.
  - **Secure Hardware (Trusted Execution Environment - TEE):** The smartphone's Secure Enclave or TEE should be used to store cryptographic keys securely.
  - **Use of Physical Layer Security:** UWB ensures precise distance measurements to prevent relay attacks. For those smartphones that do not support it, Bluetooth Low Energy (BLE) is going to be used instead.
- **Protection Against Replay Attacks**
  - **Signed Timestamps and Sequence Numbers:**
    i. For every critical command (e.g., unlock, start), the app client includes a current Unix timestamp and a strictly increasing sequence number (unique per user session with the car) in the command payload.
    ii. This entire payload, including the timestamp, sequence number, command type, user ID, and target car ID, is then digitally signed by the app client using the user's private key.
    iii. The car server, upon receiving the command, first verifies the signature using the app's public key (obtained from the client certificate during mTLS handshake and validated with the backend).
    iv. If the signature is valid, the car server then checks if the (now authenticated) timestamp is within an acceptable time window.
    v. Finally, it verifies if the (now authenticated) sequence number is greater than the last seen sequence number for that specific user.

5.3 Secure Communication Flow Diagram

Secure communication flow:

- **App Initiates Connection:**
  - The smartphone sends a connection request to the car over a secure protocol using UWB or BLE depending on the model.
  - The car responds with its digital certificate.
- **Mutual Authentication:**

  - The smartphone verifies the car's certificate with the server which works as a CA. In case of absent internet connection, it uses certificate pinning instead.
  - The car requests authentication from the smartphone.
  - The car verifies the smartphone in the same way described above.
- **Challenge-Response for Command Authentication:**
  - The car sends a random nonce to the smartphone.
  - The smartphone signs the nonce and returns the signed message.
  - The car verifies the signature before proceeding with the confidentiality protocol.
- **Key Exchange and Secure Session Establishment:**
  - Both devices perform an ECDH key exchange to derive a shared session key.
  - TLS 1.3 handshake occurs to establish an encrypted channel.
- **Command Execution with Security Measures:**
  - Commands such as unlocking the door or starting the car are only executed if authentication is successful.
  - The car verifies the signature again before executing any command (unlock, start, etc.). This time using the Diffie-Hellman temporary key pairs.
  - The car ensures the request is recent (prevents replay attacks).
  - Rolling codes are used for additional security.

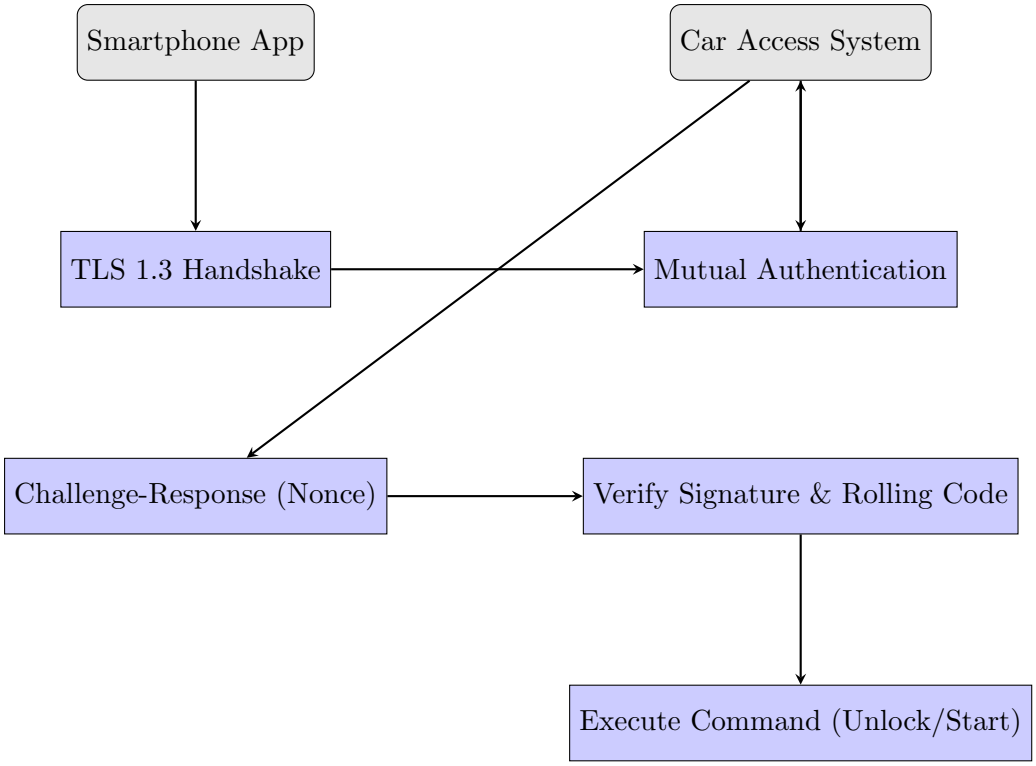

Figure 3: Secure communication flow between smartphone and car

5.4 Secure Access Sharing

The communication between a user authorized by the car owner and the car is very similar to the one reported above. It variates only in the way the car authenticates the new user:

- **Access Token Presentation and Verification:**
  Once the recipient's app authenticated the car through its digital certificate:
  - The app presents the signed delegation token ($\text{Token}_{\text{Owner}\to R}$) which includes:
    - The recipient's public key (PK_R).
    - Validity constraints (time, scope, location).
    - Binding to PK_R.
    - Owner's digital signature.
  - The car verifies the token signature with the server asking it if the owner with has granted access to other users with that public key.
  - The car challenges the recipient and verifies their possession of the private key corresponding to the token.
  - The car checks the validity of constraints (expiry, feature restrictions, geofencing, etc.).
- **Push Notifications:**
  The server will send to the user app notifications every time the allowed user will initiate a new action.

6. Key Management, Storage, and Privacy

These aspects ensure the security and privacy of sensitive data related to car access and driver identification, even if the smartphone is compromised. We will consider the presence of three main actors: the Server (S), the User (U) and the car(C).

6.1 Key Management

- **Certificate Generation and Distribution**
  The users' certificates are generated and distributed by the app server. The cars' certificate are preinstalled by the car manufacturer from factory. In this scenario, S plays the role of a CA. Its job is to monitor certificates and provide authentication when requested. In case of 'License Removal', S won't guarantee authentication upon request.

  Note: the user's sign up will take place if and only if the car is already unlock with the engine running.

  Certificate Generation Trigger Flow:

  1. The user installs the app and registers an account (if not already done) uploading her/his picture of a valid ID document associated with her/his own selfie and the car ownership document. The registrations will also require other common information as email, phone number, backup email and personal questions (e.g. elementary school teacher name).
  2. Once validated by the backend, the user can access to all the app functionalities.
  3. The user initiates a "Pair New Car" operation from the app interface.

4. The app contacts the server (S) to request a new pairing with a specific car identified by a QR code showed on the car's dashboard.
5. The server validates:
   - The user's authorization to pair with the requested car based on ownership information.
   - The eligibility of the car for pairing.
6. If the checks pass, the server proceeds with certificate issuance. It generates key pairs and X.509 certificates for the app.
7. The server issues the corresponding certificates, acting as the Certificate Authority (CA). Then certificates will be exchanged between the car and the app in order to allow for certificate pinning to happen in the future.
8. The app and the car securely store their respective private keys and certificates:
   - The app uses the device's secure enclave or keystore.
   - The car uses an embedded secure element (e.g., TPM or HSM).

S will also store it's own certificate. This certificate is generated before deployment, during the server provisioning and application development phase. C and U will use it to authenticate the communication with S.

- **Secure Access Sharing:**
  Once the server receives the sharing access request by one of the owners, it creates the Temporary Access Token (OAuth-style). The server will then send the token to the recipient user through the established secure communication protocol.
- **Key Generation and Distribution:**
  Cryptographic key pairs are going to be ECC-based. They are generated on the smartphone using a hardware-backed secure random generator within the Secure Enclave or Trusted Execution Environment (TEE). Public keys are sent to the Server and the Car, while private keys remain on the device.

  The same storing process applies to the car. The vehicle has to be equipped with a secure hardware component, such as a Hardware Security Module (HSM), a Trusted Platform Module (TPM), or a TEE-like environment integrated into its onboard system. This secure module:
  - Stores the private key securely without exposing it.
  - Makes the public key certificate available to be shared with the smartphone and the backend server.
- **Public Key Infrastructure:**
  After the user's sign up procedure, the car and the app will share their respective certificates with one another. Both C and A will use the pre-saved public key to authenticate each other in case of absent internet connection through a PKI without certificates. This authentication will allow the user to just open the car. For all the other functionalities the car needs to have an available internet connection.

  On the other hand, a delegated user will not be able to access any of the functionalities without a stable internet connection. The recipient public key will never be stored by the car.
- **Secure Storage of Keys:**

- *The app:*
  Private keys are stored in the smartphone's TEE or Secure Enclave to protect against extraction, even in the event of device compromise or jailbreaking. Biometric authentication (e.g., fingerprint or face recognition) gates access to key usage. Public keys and certificates will also be stored along side the private key in the secure module.
- *The car* will store both the private along side with the public key in its secure module.
- *The Server* Stores all information in a secure database. The identity information of users are going to be hashed and salted for further security. Moreover, instead of storing only the public key of the users, the server is going to store the fingerprint related to the user entire certificate. The chosen hashing algorithm is SHA-256.

6.2 Data Storage

- **On the Smartphone:**
  - Access credentials (in secure hardware-protected storage) Biometric credentials (via OS APIs, e.g. Android Keystore / Apple Secure Enclave).
  - The User's private keys (of the certificate and of the ECDH) within the Secure Enclave or Trusted Execution Environment (TEE).
- **On the Server:**
  - Pairs of X.509 certificated related to the car and its owner (S works as a CRL system).
  - Driving license status synchronized with competent authorities.
  - Access revocation lists and delegation logs.
  - Delegation tokens.
  - Full activity history of delegated users.
- **General Storage Policies:**
  - Sensitive data is encrypted both at rest and in transit.
  - Only the minimum required information is stored (data minimization).
  - All information regarding the user usage of the car or her/his delegations, is going to be canceled after three months. This time limit allows for driver identification in case of accidents or tickets.
- **Drawbacks:**
  The down side of having many associated users to the app is that the load of memory will be large. The app infrastructure will need a proper database system.

6.3 Privacy Protection and GDPR Compliance

- **Data Minimization:** The app only collects and stores data essential for its operation.
- **User Rights Management:** Users can access, export, or delete their personal data via in-app settings to comply with GDPR rights (e.g., right to be forgotten, data portability).
- **Pseudonymization and Consent:** Identifiers used in backend logs and analytics are pseudonymized. All optional data collection (e.g., for diagnostics or analytics) requires user confirmation first.

- **Avoidance of Tracking:** The app does not track user behavior or location unless explicitly enabled. Session identifiers are used instead of persistent ones to maintain privacy. Even when tuned on, the server won't store the past locations of the user.

6.4 Threat Model and Attack Mitigation

- **Phone Theft:** Protection using biometric gating, hardware-based secure key storage, and remote revocation options.
- **Man-in-the-Middle (MITM):** Secure mutual authentication, certificate pinning, and encrypted communication protect against interception.
- **Replay Attacks and Delegation Abuse:** Use of challenge-response mechanisms and signed delegation tokens with time and permission constraints.

6.5 Ease of Use Considerations

- Fast biometric access to keys and vehicle commands.
- In order for the app to reduce validation times, AI-based OCR and facial recognition will be implemented. This will allow to automate ID document verification, matching selfie with ID photo and reading car ownership info (VIN, license plate, etc.).

7. Attacks, Mitigations and Limitations

Attack Vectors and Threat Analysis

This section examines potential attack vectors against the smartphone car access application, analyzes their impact, and outlines corresponding mitigation strategies. Our threat model considers attacks from various entry points, focusing on both technical and human factors.

7.1 Network-Based Attacks

Man-in-the-Middle (MITM) Attacks **Threat:** An attacker positions themselves between the smartphone and the car to intercept and potentially modify communications.

Potential Impact:

- Interception of authentication credentials
- Unauthorized access to the vehicle
- Command injection (e.g., sending malicious unlock commands)

Mitigations:

- Implementation of mutual TLS 1.3 authentication between all components
- Application-Level Out-of-Band Car Certificate Validation: the app client verifies the fingerprint of the car's presented TLS certificate against a trusted record maintained by the backend server. This thwarts attackers presenting otherwise validly signed (but incorrect for the specific car) certificates.
- Use of public key infrastructure (PKI) for managing and verifying X.509 certificates.
- Hardware-backed key storage for certificate private keys on both smartphone and car.

Replay Attacks **Threat:** An attacker captures valid communications and replays them later to gain unauthorized access.

Potential Impact:

- Unauthorized vehicle access using captured legitimate commands
- Bypassing authentication mechanisms
- Exploitation of delegation tokens

Mitigations:

- Challenge-response mechanisms with nonces for each connection
- Timestamps with short validity periods (1-4 seconds)
- Rolling code implementations
- Session-specific cryptographic keys

Signal Jamming and DoS Attacks **Threat:** Disruption of communication channels through signal interference or server overload.

Potential Impact:

- Prevention of legitimate access to the vehicle
- Exploitation during failed-open recovery mechanisms

Mitigations:

- Fallback authentication mechanisms
- Local caching of credentials for offline authentication
- Rate limiting and request throttling
- Distributed server architecture to mitigate DDoS attacks

7.2 Device-Level Attacks

Stolen/Lost Smartphone **Threat:** Physical theft or loss of a device with the car access application installed.

Potential Impact:

- Complete unauthorized access to the vehicle
- Potential extraction of cryptographic material
- Identity theft for delegation misuse

Mitigations:

- Biometric authentication requirements (as detailed in the Authentication Mechanisms section)
- Remote revocation capabilities
- Secure hardware storage of cryptographic keys (TEE/Secure Enclave)
- Optional: geofencing to detect unusual locations

Malware and Application Compromise **Threat:** Installation of malicious software on the user's device that targets the car access application.

Potential Impact:

- Extraction of credentials
- Keystroke logging of PINs
- Command hijacking

Mitigations:

- Runtime application self-protection (RASP)
- Jailbreak/root detection
- Code obfuscation and anti-tampering measures
- Secure inter-process communication
- Integrity checks on application binaries

Side-Channel Attacks **Threat:** Exploitation of implementation vulnerabilities through timing, power analysis, or electromagnetic emissions.

Potential Impact:

- Extraction of cryptographic keys
- Bypass of security mechanisms

Mitigations:

- Constant-time cryptographic operations
- Hardware-backed security modules
- Regular security audits and penetration testing
- Side-channel resistant implementations of cryptographic algorithms

7.3 Social Engineering Attacks

Phishing and Spoofing **Threat:** Creation of fake applications or interfaces to trick users into revealing credentials.

Potential Impact:

- Collection of authentication credentials
- Installation of trojanized applications

Mitigations:

- User education and awareness
- Clear branding and verification indicators
- Limited credential input scenarios
- App store verification

Unauthorized Delegation Abuse **Threat:** Convincing owners to share access inappropriately or coercing them to delegate access.

Potential Impact:

- Unauthorized access through legitimate channels
- Difficult-to-detect misuse

Mitigations:

- Clear delegation expiration periods
- Granular permission controls
- Activity logging and notifications
- Periodic review prompts for active delegations

7.4 Backend System Attacks

Server Compromise **Threat:** Unauthorized access to backend servers managing user accounts and authentication.

Potential Impact:

- Mass compromise of user accounts
- Certificate authority (CA) compromise
- Modification of license verification status

Mitigations:

- Secure server hardening practices
- Regular security updates and patch management
- Intrusion detection systems
- Anomaly detection for unusual certificate operations
- Hardware security modules (HSMs) for CA operations

Database Breaches **Threat:** Unauthorized access to stored user data, vehicle information, and access logs.

Potential Impact:

- Identity theft
- Privacy violations
- Vehicle targeting for theft

Mitigations:

- Encryption of all sensitive data at rest
- Data minimization principles
- Secure database access controls
- Anonymization and pseudonymization where possible

- Regular security audits

7.5 Comprehensive Mitigation Strategy

Our multilayered defense approach combines preventative, detective, and reactive measures:

Preventative Measures

- Strong encryption for all data in transit and at rest
- Secure coding practices and regular code reviews
- Comprehensive input validation
- Principle of least privilege for app permissions
- Hardware-backed security features
- Regular vulnerability assessments

Detective Measures

- Behavioral analytics to identify unusual patterns
- Intrusion detection systems
- Continuous monitoring of authentication attempts
- Anomaly detection for unusual access patterns
- Log correlation and analysis

Reactive Measures

- Incident response procedures
- Remote kill switch capabilities
- Rapid update distribution channels
- User notification systems
- Forensic logging

7.6 Limitations & Vulnerabilities

- **Offline Access and Sync Issues:**
  If either the app or the car is offline, synchronization of permissions or revocation updates may fail. The owner will not be able of tuning on the car in case of missing internet connection.
- **Usability vs. Security:**
  Users might prioritize convenience and disable security features such as PINs or biometric authentication.
- **Social Engineering / Phishing:**
  Attackers may trick users into sharing credentials or unintentionally granting access to unauthorized individuals. Human error is difficult to eliminate entirely, and well-crafted phishing attacks may still succeed.

- **BLE:**
  For those smartphones that do not support UWB, BLE is used. In this case, relay attacks might become more probable. However, the cryptographic communication mechanism should be able to avoid them.
- **Driver Identification:**
  For simplicity the app assumes that the owner can delegate just one access at the time. Furthermore, the process of identity check might be long and take more time if requested by the authorities.
- **Adaptability:**
  The app is developed for a specific car manufacturer. Despite being a general limitation, the ideal app should be able to communicate and interact with all systems. However, standardization in in the automotive world is a relevant problem and not only in this field.

8. PoC Differences

- **License Initialization:**
  All license are being initialized as valid and a with fixed expiration date of one year.
- **License Removal:**
  In the real app a connection with the authorities API will be set to monitor the status of a license. In the PoC the license revocation of a user can happen just by hard coding it.
- **Single Car:**
  The PoC simulates the app behavior taking in consideration just one car. Multiple car can still be registered, but all interactions between users take in consideration just one car.
- **No Offline Mechanism:**
  The current PoC does not take into consideration the possibility of offline communications. All communications are carried out by TLS considering available internet connection. The car will need internet connection to even unlock. In the real implementation, offline methods would be implemented using techniques that cannot be developed on a single PC.
- **Sign-in mechanism:**
  The user signs in with just a unique username and a 4-digit pin. The documentation uploading for identity and license check is not considered.
- **Login mechanism:**
  The user logs in with a 4-digit pin instead of using biometric authentication.
- **Key pair generation:**
  The key pair generation for both the car and the server has been achieved by using RSA-2048 instead of ECC. The users key pairs are instead generated using ECC.
- **Validation mechanism:**
  The certificates validation of the app and the car through the server happens after the TLS connection establishment. The mechanism is inverted with respect to what is specified in the document. Still the authentication result is the same. The reason behind this behavior is that the implementation uses predefined libraries to build the TLS connection. Therefore, the communication set-up cannot be interrupted to retrieve the certificates and validate them meanwhile.

- **No delegation tokens:**
  Delegation tokens have not been implemented for the PoC demonstration. Instead, all delegations checks are carried out directly by querying the server for authorization.
- **Attack Scenario Simplifications:**
  - *Replay Attack Message Capture:* For the Replay Attack PoC, the "capture" of a legitimate message was simulated internally by the app client storing its last sent car command. In a real-world scenario, an attacker would need to employ network sniffing or other interception techniques.
  - *MitM Proxy Sophistication:* The 'mitmproxy.py' used in the PoC was a basic TCP proxy attempting TLS termination and re-establishment. It did not employ advanced certificate spoofing techniques beyond presenting a different, albeit validly signed (by the same CA), certificate. The primary aim was to test the backend validation logic for the car's certificate, not to bypass fundamental TLS CA trust.

9. Conclusion

In conclusion, the smartphone-based car access system presents a highly secure and adaptable approach to managing vehicle access, leveraging cutting-edge technologies to address modern security challenges. The system integrates advanced cryptographic techniques, such as mutual TLS 1.3 authentication, public key infrastructure (PKI), and certificate pinning, to ensure that both the smartphone and the vehicle authenticate each other securely. This robust cryptographic framework helps prevent unauthorized access, protecting sensitive vehicle data and user credentials from malicious actors. By utilizing biometric authentication (such as face ID and fingerprint scanning) alongside multi-factor authentication (MFA), the system adds an additional layer of protection, ensuring that only authorized users can gain access to the vehicle. Furthermore, real-time monitoring and risk-based authentication mechanisms adjust access controls dynamically based on factors like location, time, and user behavior, further enhancing security.

In addition to its technical robustness, the system is designed to handle a broad range of potential attack vectors, including man-in-the-middle (MITM) attacks, replay attacks, and device-level compromises. The integration of secure hardware elements (like Trusted Execution Environments (TEEs) and secure enclaves) ensures that cryptographic keys and sensitive information are securely stored and cannot be easily extracted, even if the smartphone is compromised. With remote revocation capabilities, users can instantly revoke access in case of a lost or stolen device, adding an essential safeguard in real-world scenarios. These features collectively ensure that the system provides strong protection against both technical and social engineering threats, ensuring vehicle access is highly secure, regardless of the context.

However, while the security features of the system are highly advanced, usability concerns and the challenges of offline access remain important considerations. In scenarios where the smartphone or the vehicle is not connected to the internet, synchronization of permissions, revocations, or updates could fail, potentially hindering user access to the vehicle. Striking a balance between robust security measures and the user experience will be crucial for widespread adoption. Users may inadvertently disable security features like biometrics or PIN authentication for the sake of convenience, which could expose the system to potential risks. Therefore, continuous efforts to enhance user education and streamline security settings without sacrificing usability will be essential to the system's long-term success.

Another significant challenge is the adaptability of the system to a wide range of car manufacturers. As the system is primarily designed with a specific manufacturer in mind, it may require further development to accommodate various vehicle models, their embedded hardware,

and software systems. Standardization and flexibility in adapting to different manufacturers will be key for the system to reach broader markets. Moreover, the integration with a variety of vehicle models, each with different security requirements and configurations, will need to be addressed for seamless interoperability.

Despite these challenges, this multi-layered approach to car access represents a significant leap forward in the integration of smartphones and vehicles. It offers an innovative, secure, flexible, and efficient solution to the problem of vehicle access, blending cutting-edge cryptography, authentication, and real-time monitoring to meet modern security demands. As automotive and smartphone technology continues to evolve, the system is well-positioned to evolve alongside these advancements, providing both users and manufacturers with a powerful tool for managing car access in an increasingly interconnected world. For the system to achieve long-term success and widespread adoption, continued attention to user experience, adaptability, and seamless integration with diverse vehicle ecosystems will be essential.

10. AI Contribution

General LLMs have been used to help brainstorm possible solutions relative to problems encountered when designing the app architecture. Retrieving ideas and up-to-date real applications allowed for a broader vision on some matters. The solutions offered were always double checked with available online and lecture material.