A-SIT

Secure Information Technology Center – Austria

Identity Recovery

Identity Recovery

Author:
Stefan More
Mail: smore@tugraz.at
Date: December 2023

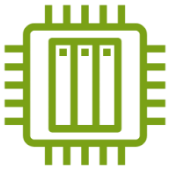
Abstract:

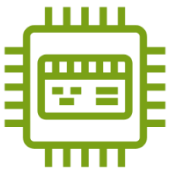
The increasing prevalence of digital identities and cryptographic key materials on various devices poses a high security risk. If a device is lost or stolen, unauthorized individuals may gain access to the stored identities and key materials. To minimize this risk, hardware-supported techniques are employed to protect the key material. These techniques, for example, prevent malware from accessing sensitive material. A problem arises when these devices are lost or damaged/destroyed, causing a user to lose access to their digital identity. The goal of this project is to further explore systems and mechanisms for the backup and recovery of cryptographic key material, as well as additional methods for identity recovery. The aim of this research project is to find a way to securely back-up and restore identities and keys stored in a Trusted Execution Environment (TEE), ensuring that identities are not lost when the device is lost.

Table of Contents

1. Introduction

Digital identity wallets are an emerging technology to enable users to store and use their identity data in a sovereign way. Common examples are use cases like electronic identities (eID), academic diplomas, public transport tickets, and other forms of certification and access credentials. By using the metaphor of a physical "wallet" that contains ID cards and credentials of various forms, identity wallets thrive to become a central place in everyone's (digital) life. In contrast to physical wallets, "digital identity wallets" can utilize digital features, such as *remote* credential showing, and modern cryptographic concepts, such as privacy-enhancing technologies. In the same way, *digital* wallets need to account for the peculiarity of digital systems. Most importantly, since a digital credential is just data, it can be copied (deliberately or malignantly) unboundedly.





This violates the assumption of their physical counterparts, where copying a credential (or forging it) is either infeasible or at least hard. Since many use cases rely on the fact that a credential can only exist a single time, and some use cases also assume that it is not easy to transfer a credential to a different person, this must be considered when designing a digital identity wallet-based system. To fulfill this requirement, technologies exist that prevent the export of cryptographic data from a device, while still allowing users to use it for digital exchanges. Examples are Trusted Execution Environments (TEE), Hardware Security Modules (HSM) and Secure Elements (SE). Some operating systems also provide systems to manage and use cryptographic keys using such hardware technologies, for example Android's Keystore system.[1]

In the same way, Internet of Things (IOT) devices use their device identities to authenticate at update servers, in their management domain, or with other devices. To do so, they use cryptographic keys and credential, which are often managed with the secure hardware technologies mentioned above. This results in an identity that is bound to the specific device and cannot be transferred.

However, devices get replaced and phones break, potentially resulting in a loss of access to the identity data stored on the device. Further, phones can be stolen or hacked, and devices can be lost, which represents another threat to the availability and confidentiality of identity data.

2. Types of Identities

In the context of identity data stored on mobile and IOT devices, we differentiate between the following types of identity data:

2.1. Physical Identities and Logical Identities

A physical identity is an identity that is bound to a specific physical device. Since identities are usually represented by cryptographic keys and credentials, this means that the (private) key is used to proof the possession/ownership of the identity. To bind the private key to the device, the key is stored on the device

[1] https://developer.android.com/privacy-and-security/keystore

in a way that it cannot leave that device.[2] This is commonly done using secure hardware features, e.g., a TEE.

In contrast, a logical identity is not bound to a specific physical device. Instead, it represents a higher abstraction of the device's identity in some use cases. For example, the device's role in a use case or the identity data of a third device that is managed by that device (but has no means to store the identity data itself). In addition, in the context of digital identities stored on mobile devices, logical identities are used to represent the identity of a (natural or legal) person. In the latter case, the identity data does not represent the physical device, but can nevertheless still be device-bound if regulations require it. In the context of this report, we differentiate between device-bound (physical) and transferable (logical) identities.

In the context of backup and recovery, the difference between the two types of identities is that the physical identity of a device can never be transferred to a second device, even if that second device is a full replacement of the functionality and role of the first device. If this is required by a use case, then logical identities must be used. However, physical identities can still serve a role in those use cases, as they can be used to facilitate the secure recovery of logical identities.

2.2. Credentials and Keys

Credentials are data structures that contain a set of attributes about a data subject (e.g., a person or device). They are used to encode identity data and stored on a mobile phone or IOT device (the holder). To provide authenticity and integrity, credentials are cryptographically signed by some issuer.

Cryptographic keys are used to prove the possession of a credential. This is commonly done by embedding the public key into the credential. In the process of presenting a credential, the credential's holder then uses their private key to sign some challenge presented by the service that requested the credential.

2.3. Other Data

Other types of identities and identity data that are subject to the backup and recovery mechanisms are:

- Decentralized identifiers (DIDs) are identifiers commonly used in the (self-)sovereign identity model. They are represented by an identifier that can be resolved into the corresponding key material by different means. They can be used to encode either logical or physical identities. In contrast to credential-based identities, the DIDs don't contain any information (i.e., attributes) about the subject apart from cryptographic keys and other technical information needed to authenticate the identities' subject. However, in the context of backup and recovery, the (private) keys belonging to the DID are subject to the same recovery requirements as the key material of credentials.
- Other data needed to operate the device or wallet, such as enrollment information of configuration data. Depending on the use case, this data is device specific (so no backup is required) or more generic and bound to the logical identity (hence backup is important).

[2] https://datatracker.ietf.org/doc/html/rfc7800

3. Identity Recovery

In this report we use the following definitions for the involved components. These components and their relation are also visualized in Figure 1 below.

- Device: An IOT device or smartphone used to store, manage, and present digital identities. In the context of this report, the sensitive information (i.e., cryptographic material) is stored on the device in a protected way (e.g., using a TEE). The device can also fulfill other functions for which it requires identity information.
    - Source Device: A device that should be replaced by another device (i.e., the corresponding target device) in case it becomes unavailable (e.g., broken, malfunctions, lost, stolen, …).
    - Target Device: The device that replaces the source device. In general, the target device has the same hardware and software capabilities as the source device it replaces. However, small variations are possible, as long as the core functionality required by the recovered system and the recovery system are present.
- Identity Backup and Recovery System (B&R): All systems needed to achieve an identity recovery process. This comprises software components on both the source device and the target device (B&R client). Furthermore, there are additional components in the overall system's architecture, e.g., a file server needed to store the backups, a device (and software) used to manage the recovery process, and the network connecting all components.
    - Management Component: The component used to manage the backup and recovery process. This component is used by the system's administrator to enroll new devices, setup up cryptographic keys, configure the backup location (i.e., Storage Component), and initiate a recovery process. Depending on the concrete use case, the management component can be, for example, a server running a web interface, or an app on the administrator's phone. This component manages the cryptographic keys used for the backup and recovery system but does not necessarily store the actual data. Hence, it is a lightweight component that does not need to be online all the time but has higher trust and security requirements since it manages access to sensitive data.
    - Storage Component: The component used to store the backed-up identity information, receive it from the source device, and send it to the target device. Additionally, it performs access control operations needed to make the (protected) backed up data available for the target device. Depending on the overall architecture, the management component and the storage component can be merged into a single component. In contrast to the management component, the storage component needs to be available and online all the time so that devices can submit new backup snapshots. Since it never receives or processes any identity data in plain text, it has lower trust and security requirements than the management device.

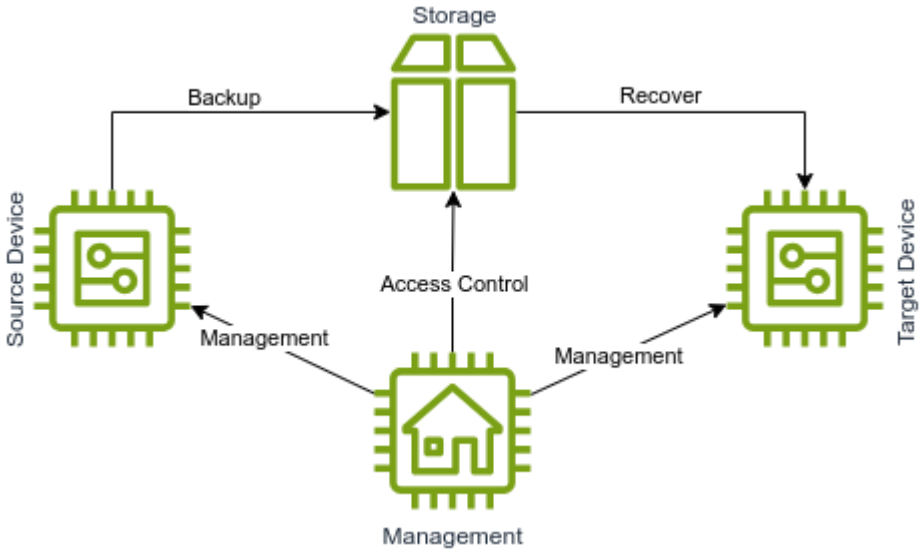


Figure 1: Overview of the B&R components and their relation

We differentiate two cases in which identity recovery is needed:

- Case 1 enables the reuse of the source device's identity, while the device is replaced in a regular way, e.g., during a schedule maintenance. In that case both the source and the target device exist at the same time.

- Case 2 enables the recovery of the protected identity data, even if the source device became unavailable. In that case the target device exists only after the old device becomes unavailable and hence no direct transfer process is possible. Since this is the more general case, solutions to case 2 also enable the resolution of case 1 scenarios.

3.1. Types of Identity Recovery

In the following we describe different types of identity recovery. All types of identity recovery discussed here cover both of the recovery cases discussed above.

3.1.1. Identity Backup and Restore

In this type, the identity data is continuously exported by the source device to the B&R storage. In case of device failure, the system's administrator uses the management component to select a suitable backup snapshot and import it to the target device. The B&D client on the target device then takes care of importing the identity data into the secure storage of the device. By doing so, the target device re-uses the cryptographic material and credentials of the source device and thereby takes over the logical identity of the source device.

To enable a secure backup and recovery flow, it is required that no unauthorized party has access to the backed-up data. This is especially crucial since the data exported from the devices originates from the device's secure storage component, and an unprotected export would break this data protection layer. To fulfill this requirement, the data is never exported from the secure storage in plain-text, and only transmitted and stored in encrypted form. Furthermore, great care is taken to ensure that the keys used

to encrypt the data are protected as well and managed in a secure way to ensure that only legitimate entities (i.e., the authorized target device) can decrypt the data. The cryptographic mechanisms utilized to facilitate this secure backup and restore are discussed in Section 4.3 below.

Full identity recovery using identity backup and restore is only possible for logical identities, as physical identities are device bound, and hence their corresponding private) key cannot be backed up and exported from the device.

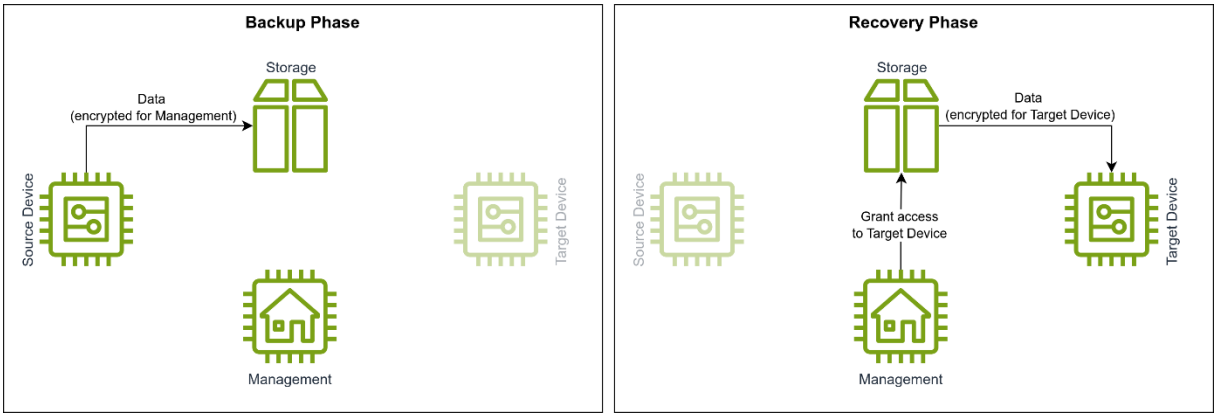


Figure 2: Conceptual overview of the B&R phases

3.1.2. Identity Re-Issuing

The alternative to a full backup and restore of identity data is a more logical recovery process. In this type of identity recovery, the cryptographic data corresponding to the identity information never leaves the device. This can be the case if the identity data is built on a form of device-bound identity, or if the device is not able to export its identity data. In that type, the identity of the target device is freshly instantiated, e.g., by a system administrator.

This is done by first enrolling the target device in the system. Next, the management device configures and generates new cryptographic material on the target device. After authenticating the device or its holder, this cryptographic material is then used to re-issue new identity information, i.e., credentials. Since the management component can re-use/issue identity information from the source device, this effectively recovers the old identity on the target device. By doing so, the logical identity is recovered, but the device's physical identity is a new one.

3.1.3. Hybrid Recovery

A third type of identity is a combination of both types discussed above.

One form of doing so is to transfer only parts of the identity information to the target device, and then use that information to re-issue all other (device-bound) identity information and bind it to the new device.

Another form of hybrid recovery is to never bind identity information to a physical device, but instead introduce *device binding credentials* that do so. In that scenario, identity data can simply be transferred to the new device (e.g., using the mechanism discussed in Section 3.1.1 above). To establish the link between the device and the identity, and thereby (re-)enable proof-of-ownership, the management component manually enrolls the device, and issues a device binding credential to it binding the device's key with the credentials.

4. Technical Aspects

In this section, we discuss the technical aspects of secure identity recovery. In specific, we focus on the storage of identity information on the device, on the binding of an identity to a specific device, as well as cryptographic mechanisms that secure the identity information in the backup and recovery process.

4.1. Secure Storage of Identity Information

Identity information often contains personal data about individuals, and as such is sensitive information that needs to be protected to comply with data protection regulations and ensure the privacy of data subjects. Similarly, identity information of devices can be considered sensitive, e.g., if it can be used to gain access to personal information, or access and control other critical resources. Thus, it is important that identity information stored on devices is protected during operation. The concrete threat model depends on the device's use case and environment.

For example, smartphones can get lost or stolen, and an attacker/thief should be prevented from accessing the user's identity wallet. This can be challenging given the malicious entity has physical access to the device and can read its storage and sometimes even its memory. To prevent this type of unauthorized access, as a minimum-security measure, wallet systems should utilize the operating system's file and disk encryption capabilities to protect sensitive data at rest and remove it from the device's memory if the device is locked. Further, users of these apps should be encouraged to use the device's security measures, e.g., configuring a device lock and always installing the latest updates.

Other common threats to identity data security are malicious apps or malware, or apps with software vulnerabilities that process sensitive identity data. An obvious protection against this threat by system and app developers is to follow security and software development best practices. Additionally, the attack surface can be minimized, e.g., by minimizing the software components that process sensitive data. To prevent other components from accessing the sensitive data, mobile platform isolation features and sandboxes should be utilized. To protect the data even further, hardware-backed protection features should be used. This is especially the case for cryptographic material such as the private key material of identities. To do so, it is advised that the "Key Management System" (KMS) software components processing this cryptographic data (e.g., using it to encrypt/decrypt or sign other data) are moved to isolated execution environments, such as a Trusted Execution Environment (TEE), Secure Element, or Hardware Security Module (HSM).

4.2. Mechanisms for Device-bound Identities

While credentials can usually be exported from the source device and imported/recovered into the target device, the corresponding cryptographic keys are sometimes device-bound. The hardware-based isolation mechanisms discussed in Section 4.1 above form the basis for the identity storage, and at the same time can be utilized to realize the device binding. To support different use cases, there are different ways to realize this:

- Allow that all data gets exported: All data can be exported from the device's secure storage and backed-up, and later imported/recovered on the target device. The data should still be protected (i.e., encrypted) before the export (see Section 4.3 below).

- Prevent that any data gets exported: All data is bound to device cannot be exported. This typically references private/sensitive data, as other data (such as public keys) need to be exported for the device and business logic to function. This protection can be realized by limiting the secure storage's interface to cryptographic operations and import functions. By doing so, other software on the device (and even other software components of the same app/tool) cannot access the sensitive data and hence can neither export it nor back it up. This is a means to realize pure physical identities.
- Allow that some data gets exported: A combination of both cases is to restrict the export of some identity data, while allowing if for other data. A way to realize this is to define export/backup interfaces for the secure storage component that apply access control mechanisms and restrict the data or types of data that can be exported. By appending access-control permission metadata to all data stored in the secure storage, fine-grained and flexible access/export control is possible. This enables various recovery types, as discussed in Section 3.1 above.
- Ensure that recovery can only be initiated for specific (class of) devices: Another option is to allow the export of data from the device's secure storage but restrict the device which can access the exported data. This can be achieved by protecting the data before exporting it, i.e., encrypting it to the specific target device's cryptographic (public) key. By combining the encryption process with key attestation features (e.g., [1, 2]), the source device can verify that the applied public key belongs to the legitimate target device, and hence ensure that only that device can decrypt the data. If the specific target device is not (yet) known, more advanced protection mechanisms are required (see Section 4.3 below).

4.3. Cryptographic Mechanisms for Secure Backup and Restore (B&R)

While hardware-backed secure storage ensures that sensitive data is protected while stored and processed on the device, cryptographic mechanisms can be used to protect the data while it is processed by the backup and restore system, i.e., while it is exported from the source device, stored somewhere, and later loaded and imported by the target device. To realize this, the data itself is encrypted using standard symmetric encryption (e.g., AES[3]). This ensures that the data is protected and can only be decrypted by parties which are in possession of this symmetric data-encryption key. Since symmetric crypto systems require the same key for encryption and decryption, this symmetric data-encryption key needs to be transferred from the source device to the legitimate target device in a secure way ("key management problem"). As an additional challenge, we assume that source and target device don't exist at the same time.

To solve this challenge, we enable the management component to grant access to new devices during the recovery process, without the need for the source device to know them beforehand. To do so, we encrypt the data-encryption key using an asymmetric key of the management component (key encapsulation mechanism, KEM). Further, we enable the management component to later re-encrypt this encrypted key so that the target device can decrypt it. This process is also visualized in Figure 2. We discuss two mechanisms to achieve this process in the subsections below.

[3] https://csrc.nist.gov/pubs/fips/197/final

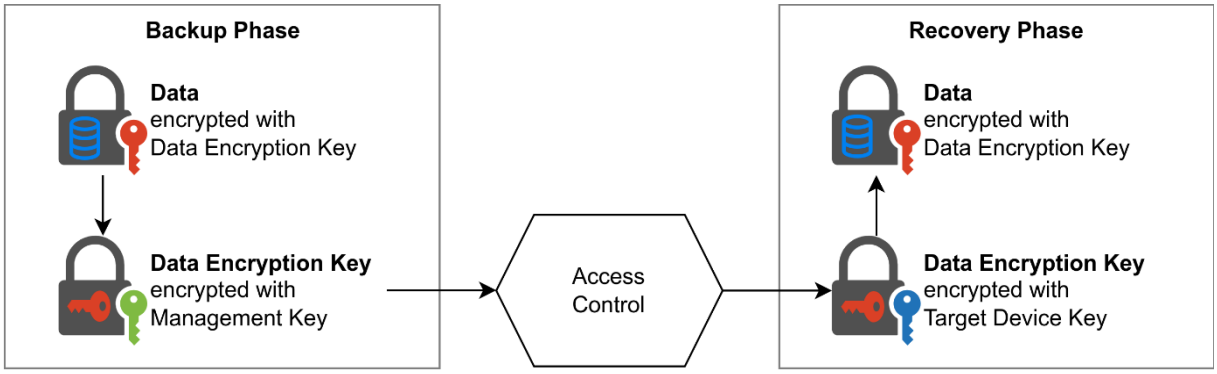


Figure 3: Conceptual visualization of the B&S system's access control mechanism

4.3.1. Using Asymmetric Cryptography

To grant access to the data encryption key of the backed-up information to any target device that does not exist during the back-up phase, the management component. needs to encrypt the data encryption key for this target device during the recovery phase. To do so, the source device uses the management public key (and not the public key of a target device) to encrypt the data encryption key. The management component can then decrypt the key, and encrypt it for the target device. This step can take place at a later stage (even after the source device became unavailable). This process is also visualized in Figure 4. In doing so, the management component can decide which target device can access the data encryption key (and thus decrypt and recover the data). In this step, the management device can also verify a key attestation[4] of the target device's key. While this is not cryptographically enforced, it enables the management component to ensure it deals with the public key of the legitimate target device.

A major drawback of this approach is that the sensitive data encryption key gets decrypted before it gets encrypted again. Thus, the key exists in plain text during the process, either at the storage component or the management component.

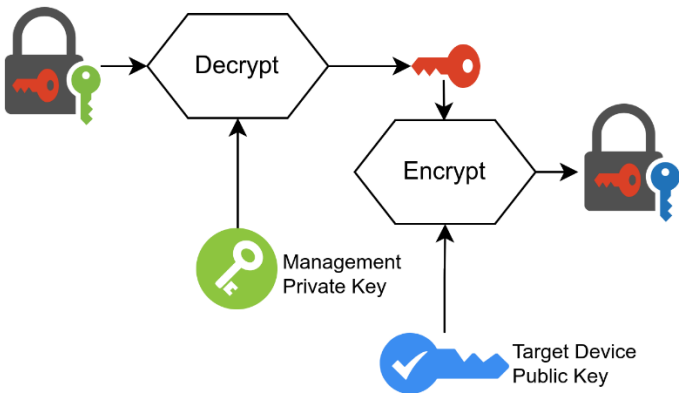


Figure 4: B&R Access Control using Asymmetric Cryptography

[4] https://a-sit-plus.github.io/attestation-service

4.3.2. Using Proxy Re-Encryption

To mitigate the drawbacks of the use of asymmetric cryptography in the context, we propose to utilize *Proxy Re-Encryption* (PRE). Using PRE, the management component combines its (private) key and the target device's (public) key to form a re-encryption key. The storage component, or the target device itself, can then use this re-encryption key to re-encrypt the wrapped/encrypted data encryption key in a way that it can then decrypt it using its own (private) key. This process is visualized in Figure 5. In the process, the storage component has never access to the plain data encryption key and can hence never access/decrypt the data. This method also enables a more asynchronous recovery process, as the management device can generate the re-encryption key (and thereby grant access to the target device) whenever it wants and does not need to communicate with the storage component at all.

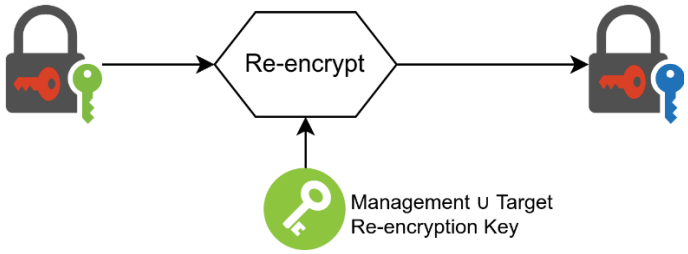


Figure 5: B&R Access Control using Proxy Re-Encryption

5. Prototype Design

To demonstrate the practicability of the backup and recovery system design, we instantiate the architecture from above. To facilitate the access control aspect of the recovery system, we utilize proxy-re-encryption. In doing so, we allow the management component to control access to the data and grants access to new (target) devices, while the storage component has no access to the data. This is an advantage over the use of simple asymmetric cryptography, where the storage component would need to decrypt the data to then encrypt it for the target device. To retain the hardware-backed protection of the data stored on the device, we combine our on-device B&R client with data protection based on a Trusted Execution Environment (TEE) [3, 4]. By doing so, we can ensure that the data is already encrypted for the target environment in the *secure world*, thus protecting it from potential malware present on the device. This prototype design using TEE and two IOT devices is visualized in Figure 6 below.

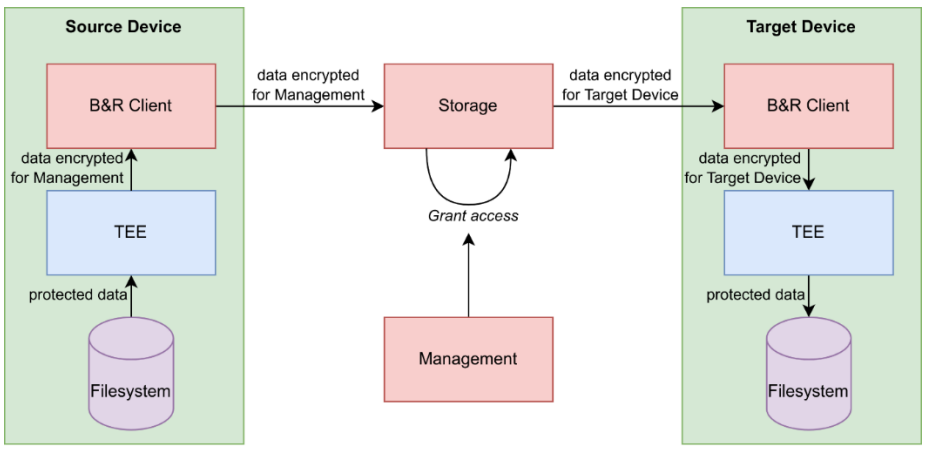


Figure 6: Conceptual overview of the B&R prototype implementation

6. Conclusion

In this report, we discussed the topic of identity recovery in the context of mobile identity wallets and smart/IOT devices. In specific, we focused on applications that use hardware-protected storage of identity data but still want to enable recovery in case of device loss or failure. We highlighted the different types of identity information (physical vs. logical) and recovery processes (backup-and-restore of identity information, re-enrollment). To ensure that the data is protected on the device as well as during the recovery process, we discussed technical aspects of secure storage and cryptographic mechanisms to enable flexible access control during recovery. Here we recommended the use of hardware-backed storage mechanisms and asymmetric cryptography as well as proxy re-encryption. We concluded the report with a technical prototype of the discussed recovery processes.

References

[1] Prünster, B, Palfinger, G & Kollmann, CP. 2019, Fides – Unleashing the Full Potential of Remote Attestation. in Proceedings of the 16th International Joint Conference on e-Business and Telecommunications: SECRYPT, SciTePress. https://doi.org/10.5220/0008121003140321

[2] Czerny, R, Kollmann, CP, Podgorelec, B, Prünster, B & Zefferer, T. 2023, Smoothing the Ride: Providing a Seamless Upgrade Path from Established Cross-Border eID Workflows Towards eID Wallet Systems. in Proceedings of the 20th International Conference on Security and Cryptography, SciTePress. https://doi.org/10.5220/0012091900003555

[3] GlobalPlatform Consortium. 2021, TEE Internal Core API Specification v1.3.1. https://globalplatform.org/specs-library/tee-internal-core-api-specification/

[4] Göttel, C., Felber, P., & Schiavoni, V. 2019, Developing secure services for IoT with OP-TEE: a first look at performance and usability. In Distributed Applications and Interoperable Systems: 19th IFIP WG 6.1 International Conference, DAIS 2019, Held as Part of the 14th International Federated Conference on Distributed Computing Techniques, DisCoTec 2019. Springer International Publishing