CYBERNETICA

Public

PEDB Use Cases For Identity Providers

Document ID: D-16-380

Version 1.0
14.02.2024

Table of Contents

1. About This Document

This document presents use cases for our new Partially Encrypted Database (PEDB) architecture pattern[1] in the context of identity providers. In general, a service provider which uses the PEDB architecture to protect sensitive data of their users is better protected against negative outcomes of data breaches. The sensitive data is encrypted and the encryption keys are protected by a trusted execution environment based on Intel® technology.

A reader of this document gets a better understanding of the capabilities of the PEDB architecture.

For a high level description of the PEDB Architecture, please refer to the *Sharemind HI for Web Services - White Paper*.

2. Use Case 1: Protecting Authentication Logs

This is a simplified description of confidentiality-aware business process for logging the history of user activity.

The identity provider provides an authentication service to end users. End users log in to various web sites and use the identity provider for authentication. Occasionally, end users want to see a list of past authentication attempts.

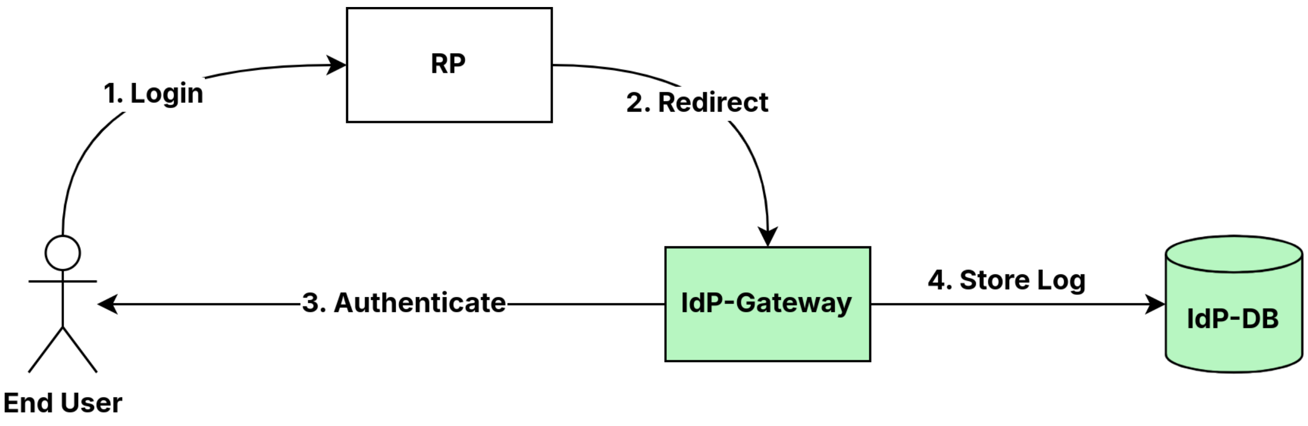

Figure 1. A high level scheme of an authentication flow, where the end user uses an identity provider (IdP) to log in to a web site (RP). The identity provider stores information about the event in their database.

Table 1. If no protection is employed, the sensitive data is stored in plain text and leaks in a data breach.

User Identifier	Service	Date
505151ab-a2eb-4...	www.example.com	2023-08-12

505151ab-a2eb-4...	www.example2.com	2023-08-12
49c49d5c-b9fe-4...	www.example3.com	2023-08-13
1d72a51e-330e-4...	www.example4.com	2023-08-13

This list contains sensitive user information of what web sites an end user used, and when. Hence a leak of this list could result in severe consequences for the service provider and for the end users.

One solution is to encrypt the database fields. Without further sophistications, this does solve the immediate threat of information leakage when intruders break only into the database, but will not protect against a malicious intruder with infrastructure-level administrator privileges. Even if the encryption key is not directly accessible, then the gateway or the application hosting the business logic must still have a mechanism to decrypt and process the data, and the same mechanism can be used by intruders to recover the unencrypted sensitive data. With the PEDB architecture, the encryption key management, cryptographic operations and data processing are moved into a *trusted execution environment* and are thus protected from intruders.

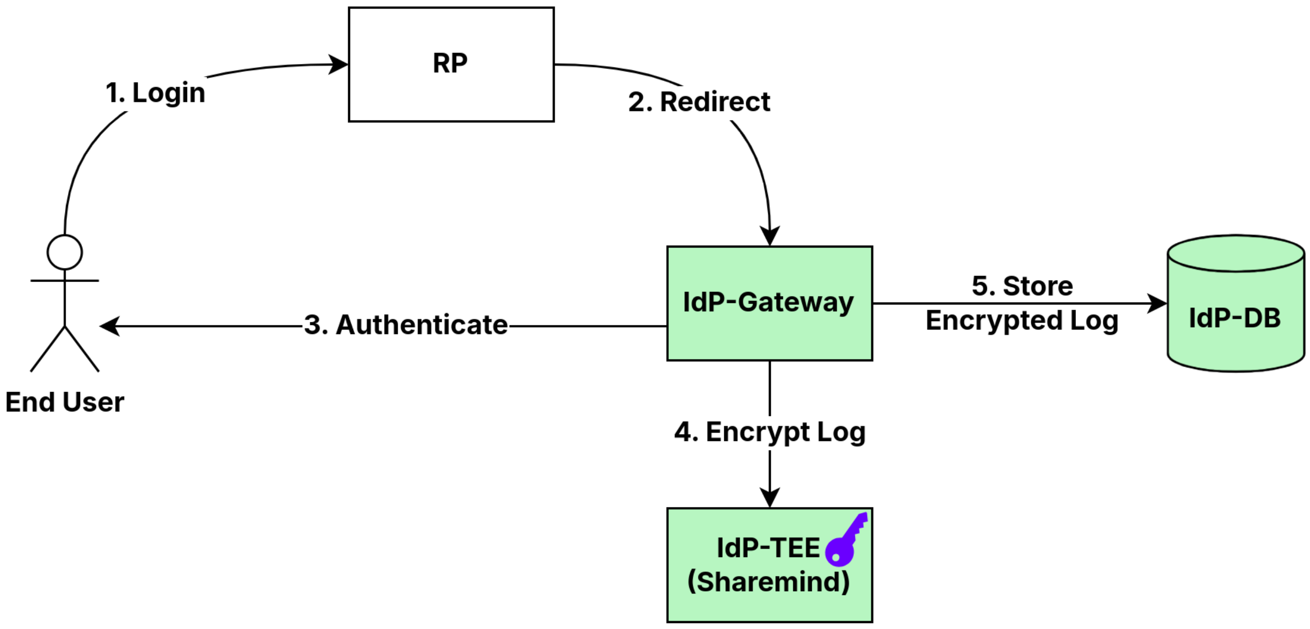

Figure 2. The high level scheme of an authentication flow when using the PEDB architecture. The identity provider uses the TEE to encrypt the sensitive data, whereas the encryption key is not accessible outside of the TEE. The encrypted data itself can be stored in any database, which means database administration tasks can be performed as usual.

Table 2. By using the PEDB architecture, the sensitive data is stored in encrypted form and does not leak in a data breach.

User Identifier	Service	Date
505151ab-a2eb-4...	IMuCOFJeV4...	MshFGWhtvc...
505151ab-a2eb-4...	zevAKndl1m...	JZZLXvSXRE...
49c49d5c-b9fe-4...	kK1D4O5D5b...	XO7rZzJCkW...

1d72a51e-330e-4...	Ru8PjmEQzn...	kI4CpArpEw...

The main strength of the PEDB architecture is in governing the subsequent read procedure, as TEE technology is flexible enough to enforce arbitrary access policies. This could mean that end users can only see the authentication logs associated with their user identifier. Law enforcement can have additional permissions to query authentication logs from specific users, adhering to further solution specific restrictions.

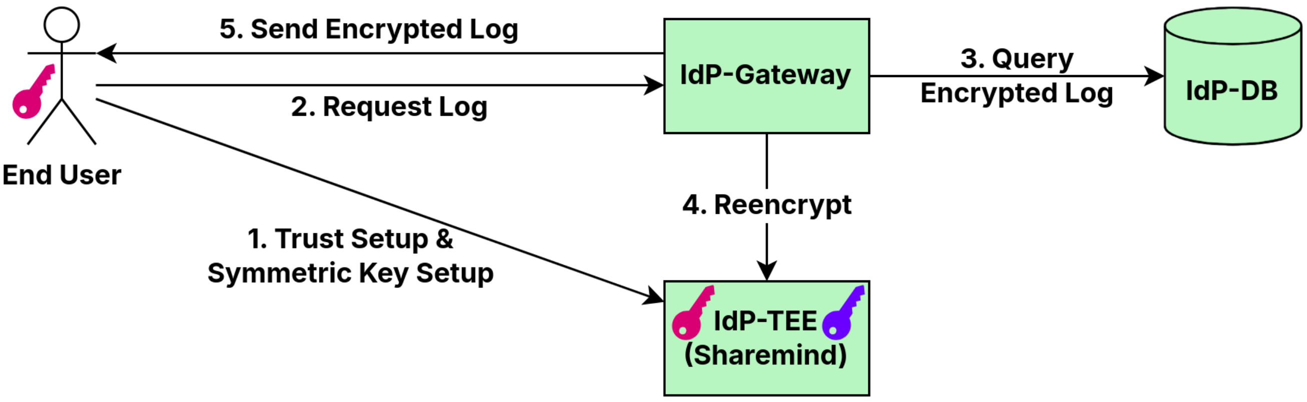

Figure 3. Access control for the sensitive data is enforced by the TEE. The end user and the TEE mutually authenticate each other and create a secure communication channel. Then the end user can request data: The TEE decrypts the queried data from the database and sends them via the secure communication channel to the end user. Access control measures can be implemented at all steps to satisfy complex requirements.

The sensitive data and encryption keys are protected throughout the whole lifecycle of the solution. Every change to the environment, like granting permissions to end users or updating software, needs to be approved by more than one person. This multi-party authorization principle ensures that a single compromised account is incapable of extracting raw sensitive data.

3. Use Case 2: Protecting End User Attributes

This chapter describes a simplified business use case where a service provider requests the attributes of an end user from an identity provider, for example to implement age confirmation or whether the end user has an acquisition permit for weapons. The PEDB architecture can be employed to protect the attributes at the identity provider from data breaches. Some attributes might originate from external registries and have expiration times depending on their type (birth date vs acquisition permissions for weapons).

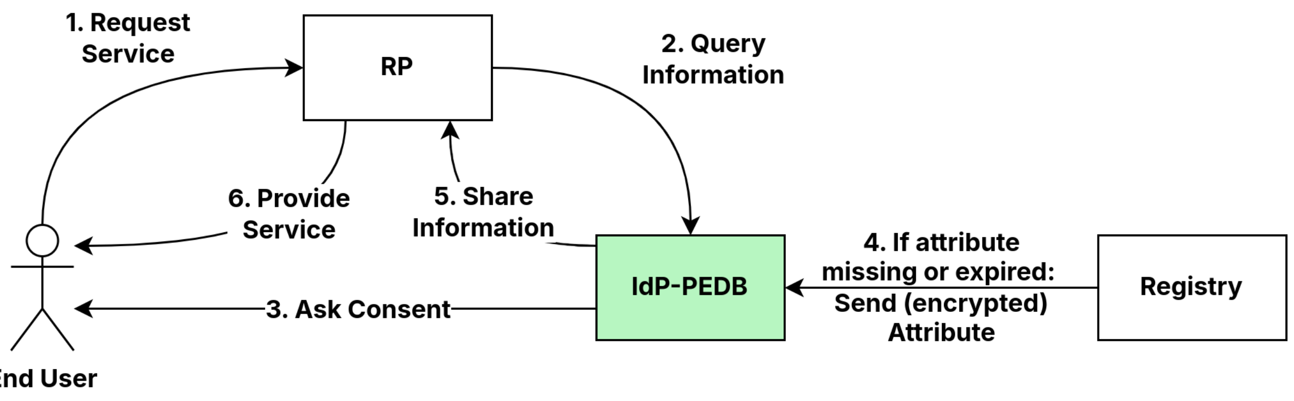

Figure 4. The identity provider can use the PEDB architecture to protect end user attributes from data breaches. In this image the gateway, TEE and DB components of the identity provider are merged into a single PEDB component.

The response from the identity provider is not restricted to the value of attributes, but can also provide answers to specific questions in order to protect sensitive data even further. In order to implement an age restriction, the RP does not need to know the birth date of the end user, they just need to know whether the end user is older than a given threshold, e.g. older than 18.

Step 3, "Ask Consent", is necessary to prevent the RP from simply downloading all attributes of all users from the identity provider. However, this redirect or similar mechanism also means extra friction which could hurt the user experience, especially since the end user already likely had to do this during the login phase. Therefore, the correct balance between user experience and security needs to be determined for each specific solution. For example, in Estonia the Smart-ID and Mobile-ID authentication mechanisms are widely used for authentication and authorization. The login procedure typically uses the four number *PIN1* for authentication, and giving a consent is done with the five number *PIN2*. However, if the explicit consent step should be omitted for the sake of increased user experience, then another technical measure needs to be put in place to prevent RP from querying arbitrary attributes, like the following: (1) Using the login action as a short-lived grant to request attributes for the end user, although this requires a custom authentication mechanism for the login procedure in the Estonian context, (2) Using Step 3 to give access for a longer period of time for a restricted set of attributes, which does not fully eliminate step 3 but still reduces the friction if attributes are queried frequently.

[1] Patent pending