An eID mechanism built along Privacy by Design principles using secure elements, pseudonyms and attributes

Denis Pinkas[1]

Abstract: This eID mechanism has been built taking into consideration Privacy by Design principles. It uses some of the basic principles of the FIDO model (Fast Identification On-line) adding certain constraints and extending the model to push user attributes. It allows a user to open an anonymous account on a server using a random pseudonym and then to push one or more attributes contained in an access token that has been obtained from an Attribute Issuer. In order to prevent the forwarding of an access token between collaborative users, a Secure Element must be used. That Secure Element shall conform to specific requirements, e.g. defined using a Protection Profile. This eID mechanism will be worldwide usable as soon as the providers of such Secure Elements publish information that can verify the genuineness of these secure elements.

Keywords : Privacy by Design, secure element, eID mechanism, authentication, Attribute Issuer, access token.

1 Introduction

Numerous eID mechanisms have been proposed up to now, but none of them has been built taking into consideration at the very beginning the privacy of individuals. In some cases, the motivation only came from the invention of a new cryptographic algorithm, for example, able to hide some of the attributes placed inside an access token that would contain all the attributes of a user.

In many cases, privacy considerations have been added afterwards while in other cases they have been simply ignored, as it is the case for the eIDAS Regulation [Re14].

The goal of this document is first to define a model and to identify requirements in order to define a world-wide scalable interoperability framework able to support the implementation of the principle of privacy by design in a secure manner and then to describe an eID mechanism that complies with these requirements.

This document uses principles established by FIDO [FIDO]. The principles and the concepts of FIDO are assumed to be understood and are not explained in detail.

[1]DP Security Consulting, 13 Rue Du Pave Des Gardes, 92370 Chaville, France, denis.ietf (at) free.fr

2. Architecture

The model has three components:

- The human user that is represented by a *User Agent*, a software and or hardware component running on a local device (e.g. on the user's computer or the user's mobile phone).
- A *Service Provider* that protects the access to a service (or to a resource within a service). In the literature, a Service Provider is sometimes referred as a *Verifier* or as a server.
- An *Attribute Issuer* that delivers in an electronic form the users' attributes of any kind. In the literature, an Attribute Issuer is sometimes referred as an Attribute Provider.

Attributes are obtained by the User Agent from an Attribute Issuer. These attributes are placed inside an access token, i.e. a string of bytes cryptographically signed by an Attribute Issuer. The use of access tokens allows support for direct trust relationships eliminating the need to include nodes in the architecture. This is illustrated on Figure 1.

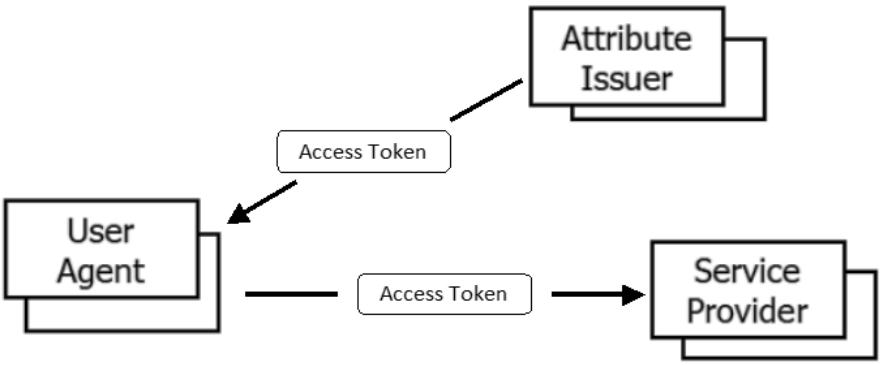

Figure 1 — The basic components of the basic model

The eID scheme is based on a push model, where user's attributes are pushed by the User Agent from an Attribute Issuer towards a Service Provider using an access token.

3. Privacy by Design (PbD) requirements

Most "Privacy by Design" principles are currently defined from the perspective of a single "data controller".

For example, the UK Data protection Act or the EU General Data Protection Regulation Principles relating to personal data processing. The point of view of the user is mainly ignored, as well as unauthorized collaborations between servers from a privacy perspective (or between users from a security perspective).

From a user perspective, it is necessary:

(a) to prevent the linkability of transactions of a user among distinct servers or even within a single server (Unlinkability),

(b) to restrict the amount of attributes being disclosed by a user to the minimum necessary to achieve a stated purpose (Data minimization),

(c) to enable users to give their consent for the communication of their attributes through an affirmative process (User Consent), and

(d) to prevent an Attribute Issuer knowing to which Service Provider an access token will be presented by the User Agent (Untraceability).

These four privacy properties will be used to construct the eID mechanism.

4. Security requirements

It is then necessary to make sure that the eID mechanism will also be secure.

Prevention of token forwarding

Generally, the access of a client to a server is conditioned by the presentation of certain "attributes" of the user.

There are cases where it is desirable to demonstrate to a server that a person is over 18 years, without the need to reveal his birth date nor any other attribute.

It is generally considered that there are two kinds of people: honest people who use a system that are located inside the system and attackers located outside of this system. While this vision is helpful, it is not sufficient. People who use a system that are located "inside the system" may also be considered as attackers in particular when they agree to work collaboratively to cheat a server.

As an example, a person over 18 that can legitimately obtain an access token demonstrating that he is over 18 could attempt to transfer that access token to a person that is under 18 so that second person could take advantage of this property long after the transfer: if someone is over 18 one day, he will necessarily remain over 18 the following days. Thus a server could remember that property and then not ask the person to present the same property again when subsequent accesses are performed by the same person.

The ABC4Trust project provides an interface able to support two solutions: "U Prove" from Microsoft and "Identity Mixer" (IdeMix) from IBM Zurich. As described in the publicly available documentation, despite the sophistication of the cryptographic algorithms that are being used, none of these two solutions is able to prevent the transfer of an attribute of a person that indeed possesses it to a person that does not possess it.

Whatever cryptography is being used, a solution that uses software only will be unable to prevent the transfer of an attribute of a person that possess it to a person that does not possess it. As soon as access tokens are being managed, Secure Elements meeting "some specific requirements" need to be involved in the process.

5. Construction of the eID mechanism

5.1. Taking into consideration PbD requirements

In order to prevent the linkability of transactions of a user among different servers, a different pseudonym will be used for every server. This eID mechanism is based on the concepts developed by FIDO [FIDO]. However one major difference with FIDO should be noted.

The FIDO Universal 2nd Factor (U2F) Overview [Un16] states on line 145:

> "Thus, the Key Handle is simply an identifier of a particular key on the U2F device".

In this eID mechanism, the key handle will be used as a pseudonym. A change in the semantics of the key handle is thus needed:

> The Key Handle shall be a random number sufficiently large (e.g. 16 bytes) generated by the U2F device, that will be used :
>
> - by the U2F device as an identifier of a particular key on the U2F device.
> - by the Service Provider as a pseudonym to identify the user.

As a result, a user can open a user account on a server using only one pseudonym. This account may be permanent or temporary. In both cases, it is thus possible to prevent the linkability of transactions of a user among distinct servers or even for the same server. As in FIDO, Secure Elements are used to store Key Handles (in this case, pseudonyms) and the associated private keys.

In order to minimize the amount of attributes being disclosed by a user to the minimum necessary to achieve a stated purpose, an access token will only contain the set of attributes that is needed. This may be done using conventional cryptography.

In order to enable users to agree to consent for the communication of their attributes through an affirmative process, the user will be able to select the attributes that he would like to be placed in an access token.

He will also be able to verify himself which attributes have effectively been placed in an access token by the Attribute Issuer before forwarding it to a Service Provider.

As a counter example, there exists a mechanism described in EN 419 212 [Ap15] that allows support for pseudonyms after 27 exchanges and identity attributes after 45 exchanges. However, since most of the exchanges are made using secure messaging, a User Agent will be unable to verify the content of the response obtained from the Attribute Issuer and hence the content of the access token.

The debugging of these exchanges is also a concern, since it cannot be done at the level of the User Agent.

In order to prevent an Attribute Issuer from knowing to which Service Provider an access token will be presented by the User Agent:

- there shall be no direct interaction between an Attribute Issuer and a Service Provider, and
- the access token shall not contain any data that would allow the Attribute Issuer to identify the Service Provider.

5.2. Taking into consideration security requirements

5.2.1 Prevention of token forwarding

It is necessary to prevent a legitimate user collaborating with another user to obtain an access token for himself but targeted to a user account (i.e. a pseudonym) belonging to an illegitimate user.

This will be done using the combination of several mechanisms:

1) pseudonyms are pseudo random numbers sufficiently large that can only be generated by secure elements. Since neither users, nor Services Providers will be able to choose their values, this means that their values will be unique.
2) key pairs associated with pseudonyms are randomly generated by secure elements. Private keys cannot be exported, nor imported.
3) access token requests can only be granted if generated by secure elements. An element of the access token request contains a field that designates the owner of the access token and the designated owner can only be a pseudonym that already exists within the secure element. That field will be duplicated into the

access token by the Attribute Issuer to designate the owner of the access token,

4) access tokens shall be presented to a Service Provider either after an authentication exchange using the private key relative to the pseudonym or as part of a data origin authentication message using the private key relative to the pseudonym.

In this way, a Secure Element will be unable to generate an access token request for a pseudonym that exists externally to the Secure Element, either contained within another access token or not (software implementation of FIDO). This means that an access token targeted to one pseudonym cannot be stolen by another user.

5.2.2 Verification that a Secure Element is being used

It is important to notice that only Attribute Issuers need to know that such Secure Elements are being used before accepting to issue an access token.

In other words, Service Providers do not need to check directly that a Secure Element is being used. They will know it indirectly: Attribute Issuers trusted by Service Providers shall not agree to issue access tokens unless they have verified that a Secure Element is being used by a User Agent to request an access token.

When Secure Elements (called "authenticators") are being used, FIDO (Fast Identification On-line) [Fa16] uses "attestation certificates". However, Secure Elements of the same model share the same attestation private key. This is necessary since attestation certificates may be checked by Service Providers. Otherwise a different private key assigned to each Secure Element would permit tracking the users. The drawback is that it is not possible to individually revoke a secure element.

In this eID scheme, a "conformance certificate" shall only be issued by the provider for a Secure Element when it meets specific requirements usually defined using a Protection Profile (PP). This does not also prevent issueing "attestation certificates" in order to remain FIDO compatible when no user attributes are being used, but "attestation certificates" are not used with this mechanism.

Another advantage of such an approach is to allow the use of individual "conformance certificates" that allow, when necessary, to revoke a given secure element.

The verification by an Attribute Issuer of the fact that a Secure Element is being used may be performed at the time of the enrollment of the user towards the Attribute Issuer or when requesting an access token from the Attribute Issuer.

Some messages generated by the Secure Element that will be forwarded to Attribute Issuers shall be signed or counter-signed using the private key initially stored by the device's provider. Obviously, to allow the verification of the digital signature of such messages, the conformance certificate shall be added to the signed or counter-signed

message.

6. The eID mechanism at work

6.1. Basic scenario

All the exchanges between a User Agent and either Attribute Issuers or Service Providers are made either using HTTPS, SSH or a VPN. The User Agent connects to a Service Provider in order to register a user. To do this, it first connects to the Secure Element and issues a SERVICE_ENROLL command (1). It forwards the response obtained from the Secure Element to the Service Provider (2). If the response is accepted by the Service Provider an account is created by the Service Provider. The Service Provider then contains an account composed of that pseudonym and an associated public key while the Secure Element contains a pseudonym and an associated private key.

Subsequently, the User Agent connects to the Service Provider and indicates the kind of service the user wishes to perform.

The Service Provider indicates the attributes types needed for this service. The Service Provider may also provide a list of Attribute Issuers that it trusts, so that the user can make a choice among them.

The user decides whether he wishes or not to disclose these attributes types considering the requested service. He may also select one of the proposed Attributes issuers. If he accepts, he then needs to decide which Attribute Issuer to contact to get them.

If this has not already been done, the User Agent creates an account for the selected Attribute Issuer using an ATTRIBUTE_ISSUER_ENROLL command (3). It forwards the response obtained from the Secure Element to the Attribute Issuer (4).

One way or another the Attribute Issuer already knows one or more attributes of the user.

For example, Mr John Doe has opened a bank account on November 12, 2010 at a branch of the Bank of North Carolina in Raleigh. His identity is checked in the presence of the customer in line with regulatory "know-your-customer" requirements.

The exchanges are illustrated in Figure 2.

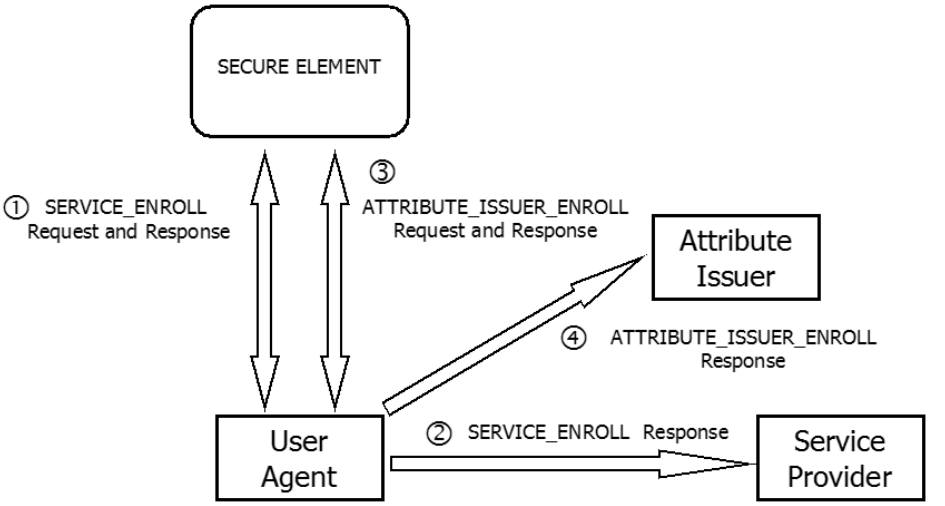

Figure 2 — Enrollment towards a Service Provider and an Attribute Issuer

The user may then wish to ask a server, acting as an Attribute Issuer for the Bank of North Carolina, to generate an access token where the owner of the access token will be the pseudonym that has been used to open an account on the Service Provider. This will be done using several steps:

The User Agent connects to the Secure Element and issues a GET_ACCESS_TOKEN command (5). The response to this command, i.e. an APDU (Application data unit) is forwarded by the User Agent to the Attribute Issuer (6). The Attribute Issuer verifies that this APDU contains a digital signature for the other content of the APDU as well as the conformance certificate of the secure element. It validates the digital signature using a trusted root. If the validation is successful, it generates the requested access token (7).

The FIDO U2F Implementation Considerations [U216] states:

> "Keys generated during a U2F registration must not be used for any purpose other than U2F authentications".

In this eID mechanism, the keys generated during registration may also be used for other purposes. In particular, they are used to provide a data origin authentication service. A change for the use of these keys in the quoted document would be required.

The User Agent receives the access token signed by the Attribute Issuer, it verifies its digital signature and makes sure that the attributes that are contained in it match the attributes that were requested.

The exchanges are illustrated in Figure 3.

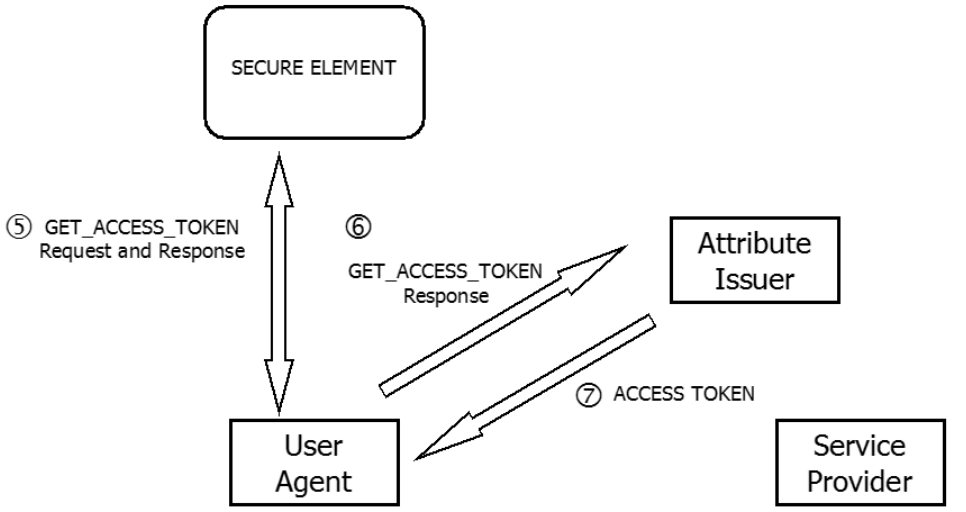

Figure 3 — The exchanges to get an access token

Once authenticated, the User Agent is able to generate a Query associated with the access token that has been received (8). This mandates the use of either HTTPS, SSH or a VPN. Alternatively, the User Agent may be willing to provide an end-to-end security in case some modification could be performed on the Service Provider side after a proxy. He can thus generate a SIGN_DATA command (8) to the Secure Element where the data contains:

(a) the access token that has just been received from the Attribute Issuer,

(b) the pseudonym that has been used to open an account on the Service Provider, and

(c) the query addressed to the Service Provider.

The response to this command, i.e. an APDU (Application Data Unit) is forwarded by the User Agent to the Service Provider (9) together with the Query and the access token. The Service Provider verifies that this APDU contains a digital signature for the Query and the access token using the public key associated with this account.

If the verification is successful and if the attributes contained in the access token matches with the expectations of the Service Provider, the Service Provider responds to the Query.

The exchanges are illustrated in Figure 4.

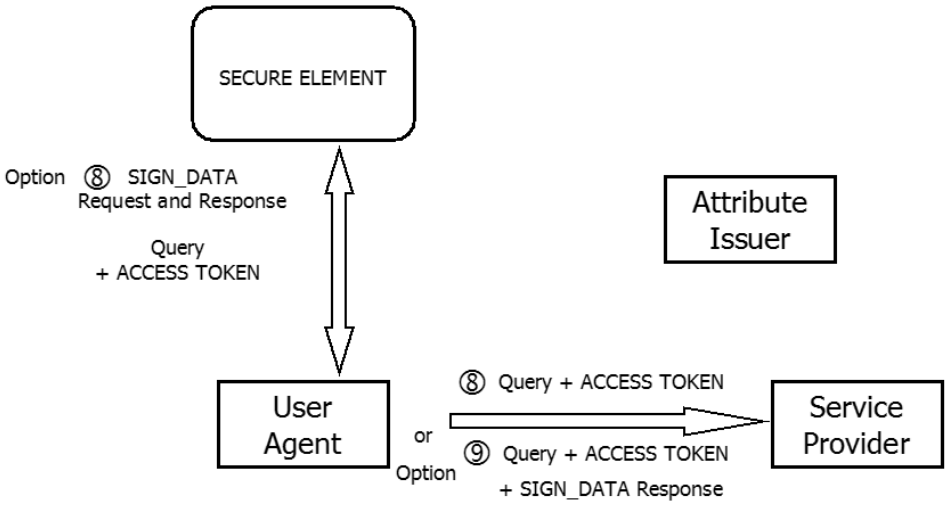

Figure 4 — The exchanges to push an access token

7. A continuous scale from anonymity to full identification

A user is able to use a full range of identification options starting from anonymity using pseudonyms to full identification using a set of attributes that will be sufficient to identify him among a set of users in a given context. He can also choose to only reveal that he is older that 18 or that he is living in a given state or/and a given town. This is achieved using "computed attributes".

8. Accumulation of attributes and Unlinkability

When the user is willing to use an existing account to access a given Service Provider, he can accumulate different attributes that will then be remembered by the Service Provider during some time period. The duration of the attributes may be controlled through information placed in the access token.

When it is desirable to prevent the linkability of transactions of a user even within a given service, once a set of transactions has been completed, the pseudonym that has been used can be deleted from the secure element. In this way, if another set of transactions is performed on the same Service Provider, a new account (and pseudonym) will automatically be created and the Service Provider will have no way of knowing that it comes from the same user.

9. The use of Secure Elements

Originally, a smart card with contacts was the favored implementation method for a secure element. Alternatives exist today, e.g. NFC (Near Field Contact) smart cards, embedded SE or SE Advanced Security SD card (SD card with an embedded SE chip).

Whatever type of Secure Element is chosen, a given user should have the ability to backup the data contained in his Secure Element in case he looses it. However, he shall not be able to use more that one Secure Element that contains that data active at any one time. The back up phase as well as the transfer of activity from one Secure Element to another Secure Element shall be controlled.

10. Interoperability

Interoperability between access tokens shall be facilitated using a limited number of formats. Access tokens may be formatted using SAML [Se16], JSON [Th16] or ASN 1 as in RFC 5755 [An16].

It will also be necessary to standardize APDUs. The SERVICE_ENROLL response shall be processable by Service Providers. The ATTRIBUTE_ISSUER_ENROLL and the GET_ACCESS_TOKEN responses shall be processable by Attributes Issuers. The SIGN_DATA response should be processable by both.

11. Conclusion

This eID scheme has been built from the very beginning along Privacy by Design principles. It is based upon a model that ends up with four components: User Agents, Service Providers, Attribute Issuers and Secure Elements taking into consideration first privacy requirements and then applying security requirements.

As with FIDO, this eID mechanism does not need the user to remember multiple identifiers and the associated authentication data. However, it extends the FIDO model in order to support the user's attributes. Changes in the FIDO specifications will be required.

This eID scheme mandates the use of Secure Elements in order to generate and store pseudonyms and associated private keys. Every Secure Element shall be manufactured in order to support a specific set of APDUs and once verified by an independent body, the provider will insert a private key and a conformance certificate inside each such manufactured secure element.

Secure elements manufactured to support the specific set of APDUs may be provided by any provider and distributed by anyone, e.g. sold in a supermarket. Hence, they do not

originally contain any personal data. Obviously, these Secure Elements should be protected by a mechanism like a PIN or/and some biometric recognition.

This eID scheme does not mandate the deployment of national eID smart cards. It is not restricted to the EU, nor to relationships with government administrations. It is worldwide scalable for both the private and the public sector.

A patent application has been made on this scheme. The next phase will be to demonstrate the mechanism through a PoC (Proof of Concept).

In comparison, the GOV.UK Verify project [Go16] which is limited to accesses to government services mandates the use of a GOV.UK Verify Hub that acts as an orchestration point for authentication requests and responses. It is stateless, i.e. it doesn't store any part of the message exchange any longer than a session. However, how users make sure that messages exchanged with that central UK hub will not be intercepted and forwarded elsewhere?

With a similar approach, the EU eIDAS Regulation [Re14] mandates the use of cross-border nodes, where such nodes could act as Big Brother.

Given the impracticality/impossibility of the eIDAS Regulation to support interoperability for eID where roughly 756 translation protocols (27 x 28) would need to be developed (it has not even been demonstrated that this is feasible) and then maintained (the amount of money for it has not been estimated), this mechanism is a realistic, simple and tangible solution.

References

[Re14] Regulation (EU) N°910/2014 on electronic identification and trust services for electronic transactions in the internal market (eIDAS Regulation).

[Ap15] EN 419 212. Application Interface for Secure Elements used as Qualified electronic Signature (Seal-) Creation Devices.

[Fa16] Fast IDentification On-line. https://fidoalliance.org/specifications/download/

[Th16] The JavaScript Object Notation (JSON) Data Interchange Format. RFC 7159

[An16] An Internet Attribute Certificate Profile for Authorization. RFC 5755

[Se16] Security Assertion Markup Language (SAML) V2.0 http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html

[Go16] GOV.UK Verify https://www.gov.uk/government/publications/introducing-govuk-verify/introducing-govuk-verify

[U216] U2F Implementation Considerations https://fidoalliance.org/specs/u2f-specs-master/fido-u2f-implementation-considerations.html

[Un16] Universal 2nd Factor (U2F) Overview. https://fidoalliance.org/specifications/overview/