A "privacy by design" eID scheme supporting Attribute-based Access Control (ABAC)

Denis Pinkas. President of DP Security Consulting SASU. France.

Abstract. This eID scheme built along "privacy by design" principles covers a full range of identification using a single mechanism starting from the use of pseudonyms, followed by a gradual disclosure of some attributes with the consent of the end-user, up to the disclosure of a sufficient number attributes that allows a full identification of an end-user under a given context, again with the consent of the end-user.

This eID scheme is resistant to the Alice and Bob Collusion attack (ABC attack) and for that reason mandates the use of secure elements that must have specific functional and security properties. Such secure elements may be provided by any manufacturer, initially without any personal data in it, and may be distributed or sold by anybody, e.g. in supermarkets, in coffee shops or by Internet web sites.

Keywords: secure element, eID mechanism, Attribute-based Access Control, ABAC, Alice and Bob Collusion attack, ABC attack, privacy by design, privacy by default.

Introduction

It is generally considered that there are two kinds of people: honest people who use a system that are located inside of the system and attackers located outside of this system. While this vision is helpful and necessary, it is insufficient. In the OAuth Threat Model Document (RFC 6819) collusions between users located inside of a system are not mentioned but nevertheless need to be considered.

In a discussion that took place on the OAuth mailing list, the following statements were made:

> "Sharing bearer tokens is a well known attack surface and **there's really no way to stop that**. Even PoP-style tokens can be shared since nothing stops Bob and Alice from sharing their secrets with each other".

The same mail also stated:

> "There's literally **nothing in the world that can prevent that level of collusion** -- PoP, token binding, DRM... nothing".

As this paper will explain, "*something*" can be done to prevent that collusion.

The Alice and Bob Collusion attack (ABC attack)

In order to illustrate what the issue is, let us use an example.

An uncle (Bob) living in New York is willing to collaborate during a short period of time with his young niece (Alice) who is less than 18 and living in Moscow. The niece is opening her own session. So the uncle does not hand over his own session to her niece at any point of time.

Let us assume that a crypto expert has written two specific pieces of software. One has been installed on the laptop of the uncle and another one on the laptop of the niece. The two laptops are able to communicate using a network (e.g. a WAN).

The niece connects to a resource server which asks her (or him ?) to demonstrate that she (or his ?) is over 18. She forwards the information received from the resource server to her uncle using the network. The uncle receives that information and connects to an Authorization Server. The uncle asks for an access token containing information demonstrating that he is over 18 and passed it back to his niece. The niece then presents it to the resource server.

Since the niece has been able to demonstrate once that she is over 18, the resource server can remember this attribute and in the future she will not need to demonstrate it again. She may keep the advantages related to this attribute until she will really be over 18.

Bob is not in any way allowing Alice to impersonate him. Bob is only allowing Alice to get advantage of one of his attributes, where such an attribute does not allow to identify him. Since Bob will not be identified, he will probably take advantage of the paradigm " *not seen, not taken* " and will thus accept to help Alice.

Access token binding protection methods developed either by the Token Binding WG [Token Binding WG] or by the OAuth WG [OAuth WG] do not allow to counter the ABC attack. Either the legitimate user can provide his key to another user, or if it can't (because it is protected by a secure element) he sends requests to his secure element to perform the cryptographic computations that the other user needs. The service provider will be unable to know which piece of software or hardware has performed the cryptographic computations.

The ABC4Trust project provides an interface able to support two solutions: "U Prove" from Microsoft and "Identity Mixer" (IdeMix) from IBM Zurich. 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.

Even when "U Prove" or "IdeMix" are used in combination with secure elements, these two solutions are unable to defeat the ABC attack. The IRMA Project which is a combination of the use of a smart card and of Idemix is also unable to counter the ABC attack.

The first description of this attack, named the "Alice and Bob Collusion" attack is available at:
https://www.ietf.org/mail-archive/web/oauth/current/msg16767.html

As the above link states:

> Whatever kind of cryptographic is being used, when two users collaborate, a **software-only solution** will be unable to prevent the transfer of an attribute of a user that possess it to another user that does not possess it .
>
> The use of a secure element simply protecting the confidentiality and the integrity of some secret key or private key will be ineffective to counter the Alice and Bob collusion attack. Additional properties will be required for the secure element.

The important point to bear in mind is that it is indeed possible to counter the ABC attack, but "one way or another" this mandates the use of a secure element with specific functional and security properties.

This does not mean that there is a single solution to defeat the ABC attack, but that any solution will mandate the use of secure elements with specific functional and security properties.

This paper describes one eID scheme able to defeat the ABC attack. That method is first explained. Afterwards a comparison is done with another method promoted by the German government. The common points and the main differences are highlighted.

This eID scheme has been constructed along "privacy by design" principles. This means that privacy principles have been applied from the very beginning for the construction of the scheme.

The basic eID model

It is necessary to start with a model. The basic model is simply a Client-Server model which has two components:

- a 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.

Taking into account the privacy requirements

The ISO Privacy framework (ISO 29100) [ISO 29100] provides a high-level framework for the protection of personal data within information and communication technology (ICT) systems. It helps organizations to ensure privacy and to achieve compliance with various applicable laws.

It focuses on organisations that processes personal data. ISO 29100 provides references to known privacy principles for information technology. Hence, most of these privacy principles apply to organizations.

However, it is important to identify privacy principles that are considered to be important for the user. From a user perspective, only one of the privacy principles that have been identified in ISO 29100 is applicable to the basic eID model. It is called "Data minimization".

Taking into consideration the first privacy requirement

The minimum data a user will present to a service provider will be a pseudonym. In order to prevent the linkability of transactions of a user among different servers, a different pseudonym will be used for every server. This privacy property is called **unlinkability**.

The mechanism described in this paper is based on a subset of the FIDO architecture (First Identity On Line) [FIDO] which allows to support pseudonyms and to perform users authentication using public key cryptography with the U2F (Universal Second Factor) protocol [U2F].

The User Agent first opens an account on the Service Provider under a given pseudonym. The human user is authenticated using a private either by signing a challenge sent by the server or by signing a unique number that the User Agent has itself generated. The server associates with the user account a pseudonym and a public key. The server does not know who the user is, but every time the users logs on to his account, the server will be confident that it is the same user, ... as long as the user does not share his private key with someone else and as long as the user does not perform any computation using his private for the benefit of another user.

As a result, a user can open a user account on a server using a pseudonym. This account may be permanent or temporary. In both cases, it is thus possible prevent the linkability of transactions of a user among distinct servers or even for the same server.

FIDO can be implemented using either software-only or hardware. Since we already know that a secure element must be used, the hardware variant has been chosen. More precisely, pseudonyms and key pairs will always be generated by the secure elements and stored within the secure elements. Pseudonyms will be large pseudo random numbers. Users will have no possibility to choose them (nor to export or import a private key). These are the fundamental characteristics that will be used later on.

FIDO allows to support "attestation certificates" allowing Service Providers to know that a secure element is being used. Since a different certificate for each secure element would allow Service Providers to trace the users, FIDO mandates a batch of secure elements to share the same attestation certificate. Since this approach has more drawbacks than advantages, the eID scheme does not support "attestation certificates". As a consequence, Service Providers do not know whether a secure element is being used or not when an account is created under a given pseudonym on a server and when the account is being used.

As it will be explained later, the secure element will need to support another kind of certificate called a "conformance certificate". Such "conformance certificate" will never be presented to a Service Provider.

The eID model enhanced with Attribute-based Access Control

On top of the FIDO architecture [FIDO], Attribute Based Access Control (ABAC) has been added. In order to be able to present attributes, an on-line Attribute Issuer is added to the basic model.

An Attribute Issuer delivers in an access token users' attributes of any kind. In the literature, an Attribute Issuer is sometimes referred as an Attribute Provider. Attributes are obtained by a User Agent from an Attribute Issuer.

The eID mechanism is based on a 'push model', where the user's attributes are pushed by the User Agent towards a Service Provider using an access token. i.e. a string of bytes cryptographically signed by an Attribute Issuer, which has been obtained from an on-line Attribute Issuer.

The enhanced model which has now three components needs to take into consideration other "Privacy by Design" (PbD) principles and "privacy by default".

Additional privacy requirements

From a user perspective, one *additional* privacy principle that is identified in ISO 29100 is important for the basic eID model: "Consent and choice". Thus, from a user perspective, in addition to the minimization of the amount of attributes being disclosed by a user to the minimum necessary to achieve a stated purpose (Data minimization), it is necessary:

- to enable users to agree to consent for the communication of their attributes through an affirmative process (User Consent).

It is also necessary to take into consideration a privacy principle that is not identified in ISO 29100 (because the ISO only considers a model with two components: a user and a data controller) :

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

These three privacy requirements, i.e. Data minimization, User Consent and Untraceability, will be used to finish the construction of the eID mechanism.

Taking into consideration the three additional privacy requirements

Data minimization: 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.

User Consent : 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 into an access token.

The user should be able to verify which attributes have effectively been placed in an access token by the Attribute Issuer before forwarding it to a Service Provider.

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

1 there shall be no direct interaction between an Attribute Issuer and a Service Provider, and

2 the access token shall not contain any data that would allow the Attribute Issuer to identify the Service Provider.

This last condition is rather important otherwise Attribute issuers would have the possibility to act as "Big Brother". This allows to support the following property: "**Big Brother impossible by design**".

Security requirement

There are cases where it is desirable to present a subset of the user's attributes, e.g. demonstrate to a server that a person is over 18 years, without the need to reveal his birth date or any other attribute. The eID scheme must be able to defeat the Alice and Bob Collusion attack (ABC attack).

Taking into consideration the security requirement

Every access token shall include the pseudonym and the public key used by the legitimate user on the targeted Service Provider. When the access token will be presented to the target Service Provider, the user will need to demonstrate (or to have demonstrated) that he has the knowledge of the corresponding private key. Access tokens will only be granted by an Attribute Issuer if the Attribute Issuer has been able to verify that the access token request originated from a secure element holding a conformance certificate (and the corresponding private key).

This means that conformance certificates will only be presented to Attribute Issuers (but never to Service Providers). One might argue that conformance certificates will allow Attribute Servers to establish links between their users. This is true. However, it should be observed that Attribute Issuers need anyway to know their users and hence, they already know the family name, the first name, and more. This means that they are already in a position to establish such links using other data, if they wish to do so.

The main benefit is that Attribute Servers will know that a secure element is indeed being used. Since Attribute Issuers will never accept to issue an access token if such a verification has not taken place, when a Service Provider receives an access token, it will also know it, but only indirectly. This is a transitive trust relationship.

To close the loop, an element of the access token request shall contain a specific field that designates the owner of the to-be-issued access token. The access token request shall be generated by the secure element and be signed using the private key which was initially used to create the account on the Attribute Server. That specific field may only contain a public key and a pseudonym that both already exist within the secure element. These two data will be blindly copied into the access token by the Attribute Issuer in order to designate the legitimate owner of the access token.

In this way, a secure element will be unable to generate an access token request for a pseudonym and a public key that exist externally to the secure element, either contained within another secure element or not (software implementation of FIDO).

The Attribute Issuer needs to know that the access token request has been generated by a secure element. This can be done either at the first logon to the Attribute Issuer by countersigning the authentication request using the private key corresponding to the public placed in the conformance certificate, or at the first access token request to the Attribute Issuer by countersigning the access token request in the same way. Bob will have no possibility to designate a pseudonym and a public key that belongs to Alice. If Alice uses a software-only solution and creates an account using a pseudonym and a public key that belongs to Bob, she will not know the corresponding private key and hence she will never be able to use Bob's account.

This means that an access token targeted to a public key and a pseudonym cannot successfully be used by another user, even if the legitimate holder of the access token (e.g. Bob) agrees to forward this access token to some other user (e.g. Alice).

To summarize, this eID scheme is using a combination of several mechanisms :

1) pseudonyms are pseudo random numbers sufficiently large that must be generated by the secure elements. Since neither users, nor Services Providers will be able to choose their values, their values will be unique.

2) key pairs associated with pseudonyms are randomly generated by the secure elements. Private keys cannot be exported, nor imported. Their values will also be unique.

3) access token requests will be granted by an Attribute Issuer only if they have been generated by a secure element. An element of the access token request shall contain a field that designates the owner of the to-be-issued access token. That field may only contain a public key and a pseudonym that already exist within the secure element. That field will be blindly copied into the access token by the Attribute Issuer in order to designate the legitimate owner of the access token.

4) access tokens will be presented to Service Provider either after an authentication exchange using the private key associated with the public key and the pseudonym contained in the access token or as part of a data origin authentication message using the private key associated with the public key and the pseudonym contained in the access token.

Conformance certificates

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

Another advantage of such an approach is to allow the use of individual "conformance certificates" that allow to revoke a given secure element when necessary. This is not the case when using "attestation certificates".

The secure elements

This eID scheme mandates the use of secure elements in order to generate and store pseudonyms and associated key pairs.

Every secure element shall be manufactured in order to support a specific set of APDUs (Application Protocol Data Units) along with specific functional and security properties and once verified by an independent body, the manufacturer 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 manufacturer and distributed by anyone, e.g. sold in a supermarket.

Hence, the secure elements do not originally contain any personal data.

Obviously, these secure elements need to be protected by a mechanism like a PIN or/and some biometric recognition.

In the past a smart card with contacts was the favored implementation of a secure element. Many 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).

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 is using a User Agent, a piece of software present in his computer environment.

The user registers to the Service Provider through his User Agent. The User Agent issues a SERVICE_ENROLL command (1) to the Secure Element and. 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 under a pseudonym associated with a public key is created by the Service Provider. The Secure Element contains a pseudonym and an associated key pair (i.e. both the private key and the public key).

Subsequently, the User Agent connects to the Service Provider and indicates the kind of action the user wishes to perform. For this specific action, 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 considering the action he is willing to perform. If he accepts, he indicates to the User Agent 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 has been checked in the presence of the customer in line with regulatory "know-your-customer" requirements.

The exchanges are illustrated in Figure 2.

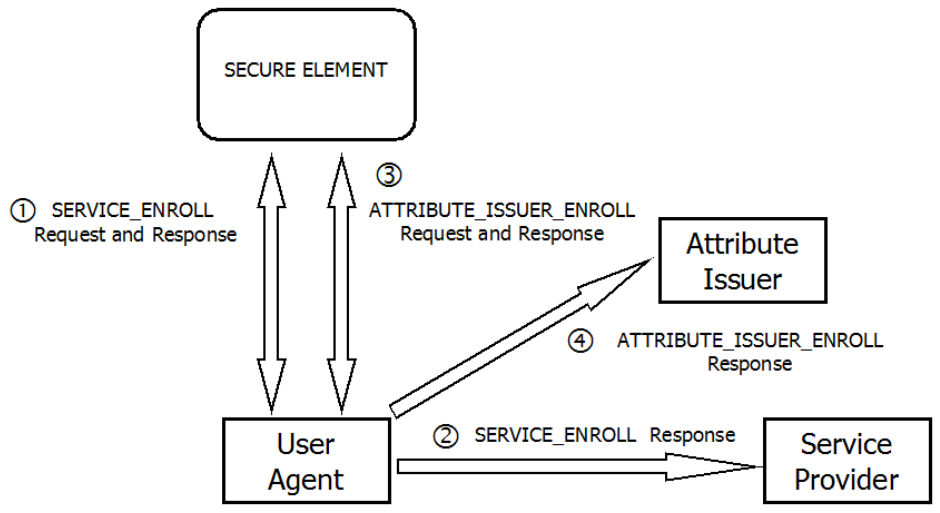
SECURE ELEMENT
① SERVICE_ENROLL Request and Response
③ ATTRIBUTE_ISSUER_ENROLL Request and Response
Attribute Issuer
④ ATTRIBUTE_ISSUER_ENROLL Response
User Agent
② SERVICE_ENROLL Response
Service Provider

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

The user asks to 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 user able to use the private key corresponding to the public key that has been used to create an account on the Service Provider. This process will be done using several steps.

The User Agent issues a GET_ACCESS_TOKEN command (5) to the Secure Element. It indicates to the Secure Element the name of a Service Provider for which the user has already created an account as well as the name of an Attribute Issuer for which the user has already created an account. 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 valid digital signature using the public key associated with the account of the user as well as a valid counter digital signature using the private key associated with the conformance certificate of the secure element. If both validation are successful, it generates the requested access token (7).

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

The exchanges are illustrated in Figure 3.

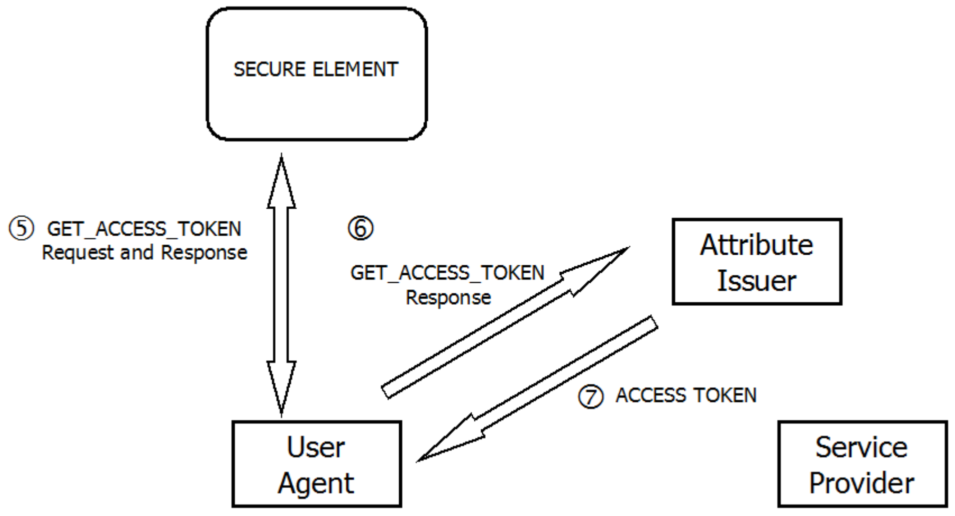

Figure 3 — The exchanges to get an access token

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.

The exchanges are illustrated in Figure 4.

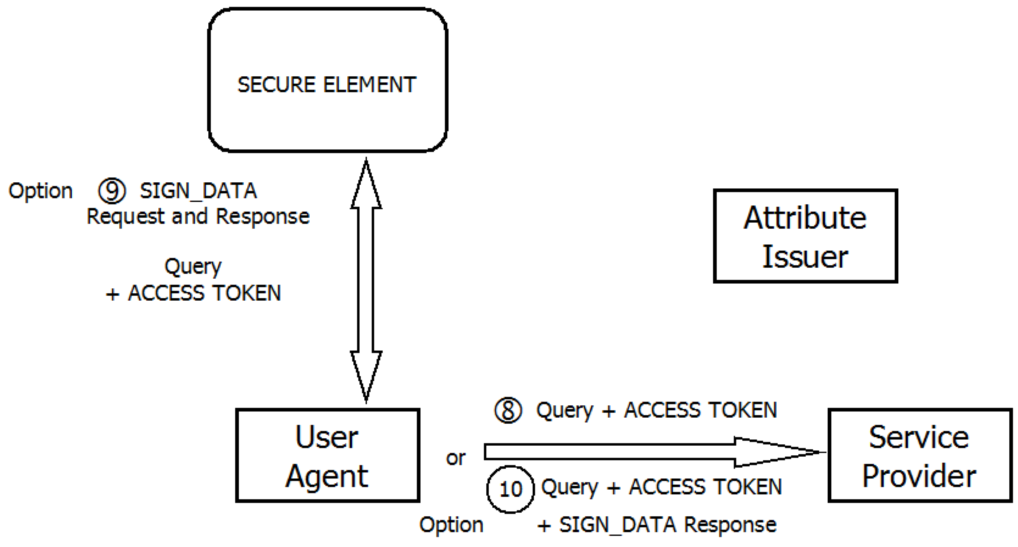

Figure 4 — The exchanges to push an access token

Alternatively, the User Agent may be wishing to provide an end-to-end security in case some modifications would be performed on the Service Provider side after a proxy. He can thus generate a SIGN_DATA command (9) 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 (10) 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.

A continuous scale from anonymity to full identification

A user can use a full grade 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 over 18 or/and that he is living in a given state or/and a given town. This is achieved using "computed attributes".

What would need to be done to integrate this eID scheme within OAut 2.0 ?

This eID scheme makes use of access tokens. Obviously, the JWT format is the most appropriate.

This eID scheme mandates the use of secure elements. Secure elements generally support APDUs (Application Protocol Data Units) which are defined in ISO 7816.

This eID scheme uses the following main exchanges:

- one exchange for registration under a pseudonym with a Service Provider or an Attribute Issuer,
- one exchange for authentication under a pseudonym with a Service Provider or an Attribute Issuer,
- one exchange for an access token request to an Attribute Issuer, and
- one exchange for the presentation of an access token to Service Provider.

Since these exchanges are encoded using APDUs, this means that OAuth 2.0 messages should be able to carry APDUs. They would thus need to carry binary blobs which would need to be forwarded to a part of the receiving entity able to decode and process these APDUs. Up to now, the IETF has not defined APDUs and such definition is thus unlikely to happen within the IETF, but should be done elsewhere.

The German eID scheme

On February 20, 2017, Germany initiated the notification procedure of the German eID contained on the German ID card and the electronic residence permit according to Article 9 of eIDAS Regulation (EU). The current German ID card is an ID-1 (credit card size) plastic card with an embedded 13.56 MHz RFID chip that uses the ISO 7816 protocols.

The main idea behind this eID scheme are as follows. The attributes of the users are only issued by the German government and are already stored in the card. They may only be revealed to Service Providers once they have obtained a CV certificate (Card Verifiable certificate) from a German agency. All the exchanges with the card are performed using "secure messaging" (all the messages are encrypted and integrity protected). The core idea is that the card is a "slave" of the Service Providers.

The protocols are defined in an European Norm EN 419 212 [EN 419 212] which has the generic title "Application Interface for Secure Elements for Electronic Identification, Authentication and Trusted Services".

EN 419 212 is composed of five parts :

- EN 419212-1 Part 1: Introduction and common definitions.
- EN 419212-2 Part 2: Signature and Seal Services.
- EN 419212-3 Part 3: Device authentication protocols.
- EN 419212-4 Part 4: Privacy specific Protocols.
- EN 419212-5 Part 5: Trusted eService.

Another document currently in ISO at the DIS stage within ISO/IEC/JTC1/SC17 is DIS 19286 "Identification cards— Integrated circuit cards— Privacy-enhancing protocols and services" which has the advantage of providing an overview of the exchanges rather than an individual description of every single APDU.

This document describes how to support pseudonyms by using 27 exchanges or identity attributes by using 45 exchanges. This is far from simple and debugging at the level of the user workstation is not that easy since most of the exchanges are using 'secure messaging'.

Pseudonyms are supported: a secret key stored in the card is used to derive a unique pseudonym for every Service Provider (but only if it got a CV certificate). In order to use this German eID card, a Service Provider, whether it is located in Germany *or elsewhere in the world* must obtain a CV certificate from a German agency.

For every registered Service Provider, the German agency will issue a certificate revocation list that will be specific to that Service Provider. This is necessary to prevent Service Providers to establish links between the users using unique card identifiers that would be contained in a single revocation lists for revoking the CV certificates.

The main advantage, but not the least, is that the German scheme is able to counter the ABC attack.

Since the card is contact less (RFID chip), a protocol (called PACE) is needed to avoid skimming attacks and hence a password is first necessary before being able to enter the eID PIN. This places some burden on the user.

If the eID card is lost (or stolen), it is likely that the legitimate holder of the card will have to wait a few days (or weeks ?) before being able to get an access to the web sites that he visited.

Since, the user will use his German ID card quite often, the card will leave his pocket quite often and the likelihood of losing it will increase.

Comparison between this eID scheme and the German eID scheme

Common points

Obviously, both schemes are using secure elements. In both cases ISO 7816 protocols, are being used. This means that in the German case, Service Providers, eID-servers and Attribute Servers need to directly "understand" this protocol, while for this eID scheme, Attribute Issuers only need to decode access token requests and Service Providers only need to decode access token presentations.

Differences

Pseudonyms are also supported by the German eID scheme. However, they can only be used if the Service Provider got a CV certificate from a German agency. As far as privacy is concerned, whereas the basic FIDO mechanism which is used with this eID scheme does not allow to know anything more than the pseudonym, the German eID scheme is leaking additional information: the fact that the user is a German citizen or is a Germany resident (since he got a German eID card from the German government).

With this eID scheme, the secure element does not need to be provided by a governmental agency. There is no need for any country to issue an national eID card. Service Providers do not have to register to any governmental agency. Pseudonyms can be used worldwide (there is no need for Service Providers to get a CV certificate from the German agency). Thus this eID scheme would be easily worldwide scalable.

The key point, with this eID scheme, is that the user has the full control of his secure element, not the Service Provider, nor any governmental agency.

This eID scheme does not use secure messaging (and hence debugging is rather easy). Nevertheless, this eID scheme, in addition to TLS/SSL, is able to support an end-to-end security between the secure element and the Service Provider by using the private key used for the authentication of the user to integrity protect all the messages (and the access tokens) that are sent by the user.

This approach highlights another difference in terms of privacy: User Consent (i.e. to enable users to agree to consent for the communication of their attributes through an affirmative process). This is supported by the German eID scheme with a coarse granularity level: either global approval or global denial.

The attributes that can be retrieved out of the card are defined through the content of a CV certificate (Card Verifiable certificate) delivered to the Service Provider by a German government agency. These attributes can be retrieved *whatever kind of action* will be performed by the user. On the contrary, with this eID scheme, the user can individually choose the attributes that he believes are proportionate *to perform a given action*.

With this eID scheme, the secure elements may be distributed or sold by anybody, e.g. in supermarkets, in coffee shops or by Internet web sites (like a SIM card). Secure elements are initially provided empty of any personal data. The user may choose its own PIN (hence there is no PIN distribution problem to be solved).

The last advantage, but not the least, is that this eID scheme allows to directly support attributes that are not necessarily issued by a national authority. Attributes Issuers may be : Banks, Universities, Golf-Club, etc... Since banks "need to know" their customers they would be well placed to issue access tokens that contain attributes like the family name, the first name, the gender, the date of birth or even the current home address (with a better accuracy than a national authority).

One might argue that an Attribute Issuer will have the knowledge of some pseudonyms and public keys that are being used on Service Providers and of which attributes are going to be used. This is true. However, it should be observed that Attribute Issuers will not be able to know to which Service Provider a given pseudonym/ public key relates.

With this eID scheme, at any time, a user will be able, to backup his personal data from one secure element into one or more secure elements but will only be allowed to have, at any instant of time, one and only one secure element placed into an operational state. Hence, when a secure element is lost or stolen, its legitimate holder will be able to use another secure element immediately after the lost or stolen secure element has been revoked. Users will thus not have to wait a few days (or weeks ?) before being able to get an access to the web sites that he visited, as it would be the case for a national eID card. Ease of use is important.

The format of the secure element is very flexible and does not need to be a RFID card as in the German case.

Conclusion

This eID scheme has been built along Privacy by Design principles and supports "privacy by default". It is based upon a model with four components: User Agents, Service Providers, Attribute Issuers and secure elements and takes into consideration both user's privacy requirements and security requirements.

As with FIDO [FIDO], this eID mechanism avoids the need for a user to remember multiple identifiers and the associated authentication data (e.g. passwords) and hence it takes into consideration the user's needs (i.e. to get rid of more than 100 passwords). However, it extends the FIDO model in order to support user's attributes.

This eID scheme mandates the use of secure elements and is based on the use of access tokens. It does not mandate the deployment of national eID smart cards.

This eID scheme has been built to defeat the Alice and Bob Collusion attack (ABC attack).

This eID scheme would be worldwide scalable for both the private and the public sector.

This eID scheme uses conventional asymmetric cryptography and hence there are no performance issues. It does not use either sophisticated / complex techniques like 'secure messaging'.

This eID scheme exchanges messages that are "in the clear" (but are obviously protected using TLS).

Intellectual Property Rights (IPR) have been requested for this eID scheme to the WIPO (World Intellectual Property Organization).

The OAuth 2.0 framework could be extended to support this eID scheme by allowing OAuth 2.0 messages to contain stream of bytes dedicated to the transmission of some APDUs.

References

- [EN 419 212] EN 419 212. Application Interface for Secure Elements for Electronic Identification, Authentication and Trusted Services. Parts 1 to 5.
- [FIDO] Fast IDentification On-line. https://fidoalliance.org/specifications/download/
- [OAuth WG] OAuth WG: https://tools.ietf.org/wg/oauth/
- [ISO 29100] ISO 29100 : Information technology — Security techniques — Privacy framework.
- [Token Binding WG] Token Binding WG: https://tools.ietf.org/wg/tokbind/
- [U2F] Universal 2nd Factor (U2F) Overview. https://fidoalliance.org/specifications/overview/