SAML Privacy-Enhancing Profile*

Moritz Horsch[1], Max Tuengerthal[2], Tobias Wich[2]

[1] Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt
horsch@cdc.informatik.tu-darmstadt.de
[2] ecsec GmbH, Sudetenstraße 16, 96247 Michelau
{max.tuengerthal,tobias.wich}@ecsec.de

Abstract: We present the SAML Privacy-Enhancing (PE) profile which empowers users to take control of the authentication process and their personal data. Users have the full control of the application flow and get detailed information about the involved participants and the revealed attributes. This enables users to give informed consent for the authentication. The new profile builds on well-established standards and technologies. We use the common SAML Authentication Request and provide the additional information as extensions based on SAML Metadata.

1 Introduction

The Security Assertion Markup Language (SAML) is a widespread framework for exchanging authentication and authorization information between entities. SAML in particular takes place in single sign-on solutions through which users authenticate to a dedicated identity provider (IdP) to get access to multiple service providers (SP). The SAML Web Browser Single Sign-On (Web Browser SSO) profile [HCH+05, Section 4.1] covers the scenario of users requesting services through a browser and getting redirected to an IdP for authentication. To provide means for more sophisticated authentication and transmission protocols the SAML Enhanced Client or Proxy (ECP) profile [HCH+05, Section 4.2] describes an client application which is capable of directly contacting the IdP.

However, there is a gap between the SAML Web Browser SSO and the ECP profile. SAML Web Browser SSO is restricted to the browser's capabilities. Thus, authentication methods more sophisticated than username/password need a browser extension, which is hard to develop and to maintain for the wide variety of web browsers. SAML ECP does not match the general user flow in the Internet, because the client application needs to determine the corresponding IdP which is challenging for new and unknown services. Thus, it lacks of the common use case in which the user uses a web browser and an additional client application for strong and more sophisticated authentication. The technical guideline TR-03124 [BSI14] somewhat covers this use case by piggybacking the SAML Authentication Request on the local HTTP-based client activation mechanism. However, this is restricted

*This work was supported in part by the European Union within the 7FP project FutureID (ICT-318424).

to the infrastructure and interfaces of the German Identity Card. Our SAML profile provides a more universal solution.

Furthermore, SAML Web Browser SSO has serious privacy issues. For instance, the automatic redirection to the IdP lacks user consent and reveals information about users even if they abort the authentication later. SAML also does not provide detailed information for users about the SP and the IdP like terms of usage, which makes it hard for users to see which attributes get revealed and to whom. In comparison to the STORK project, which developed an extension to the SAML Authentication Request including among other things the specification of an assurance level and required attributes [AMHAJ+10], our presented SAML profile provides means for client applications and a more sophisticated informed user consent.

We present the SAML Privacy-Enhancing (PE) profile which is based on the results of the FutureID Reference Architecture as described in D21.04 [BHS+14]. It provides means for using browsers to support the common browsing habits of users and client applications for more sophisticated authentication methods. The profile empowers users to take control over the authentication process and their personal data and provide detailed information of the participants and the authentication to enable an informed user consent. Our presented SAML profile builds on the existing messages flows, protocols, bindings, and data structures of the SAML standard [CKPM05] and in particular the SAML Web Browser SSO and SAML ECP profile [HCH+05], TR-03124-1 [BSI14], and the Holder-of-Key binding [KS10]. This enables an easy integration in existing SAML infrastructures.

The usage of an additional application which performs the authentication causes issues regarding secure bindings. A solution as described in the technical guideline TR-03124 for the German eID card has certain drawbacks and cannot be applied to any other credentials to date. We present a method to use our proposed SAML profile with the secure Holder-of-Key binding [KS10] which also provides privacy preserving properties.

The paper is organized as follows. We give a brief introduction to SAML in Section 2. In Section 3 we present the SAML PE profile and we provide more technical details in Section 4. In Section 5 we describe how a secure channel binding is realized and we conclude the paper in Section 6.

2 SAML

The Security Assertion Markup Language (SAML) [CKPM05] is a framework for exchanging authentication and authorization information between entities. It specifies the syntax and processing of assertions about a user issued by an IdP. SAML specifies multiple protocols and bindings for message transport, which are combined in SAML profiles. Messages and assertions are encoded in XML.

Protocols SAML protocols are used to exchange messages between the participants and are based on the common request-response paradigm. The most common used proto-

col is the Authentication Request Protocol [CKPM05, Section 3.4] which comprises an `AuthnRequest` to request an authentication process and a `Response` representing the authentication result including an assertion.

Bindings Bindings specify how SAML messages are transported between the participants. For instance, the SAML SOAP Binding [CHK+05, Section 3.2] specifies how SAML messages are mapped into SOAP messages. The SAML PAOS Binding describes the *Reverse HTTP Binding for SOAP* in which HTTP requests are used to transmit SOAP responses and HTTP responses to transmit SOAP requests. Furthermore, there exists SAML bindings for Redirect, POST, Artifact, and URI [CHK+05].

Profiles A SAML profile describes the application flow for a scenario and specifies the data structures, protocols, and bindings which are used within the profile. The SAML standard specifies five SAML profiles [HCH+05]. In the following we provide a short description of the two major profiles, the Web Browser Single Sign-on (Web Browser SSO) profile and the Enhanced Client or Proxy (ECP) profile.

The SAML Web Browser SSO profile [HCH+05, Section 4.1] specifies a scenario in which a user agent (UA) requests a service or resource by a SP and gets redirected to an IdP to perform the user authentication. The UA is usually a plain-vanilla web browser and is used by the user to access services provided by the SP. As illustrated in Figure 1, the Web Browser SSO profile comprises a UA (i.e., web browser), a SP, and an IdP. In the first step the UA requests a resource by the SP. The SP responds with a SAML `AuthnRequest` in Step 2 using the POST, Redirect, or Artifact binding [CHK+05]. In Step 3 the UA forwards

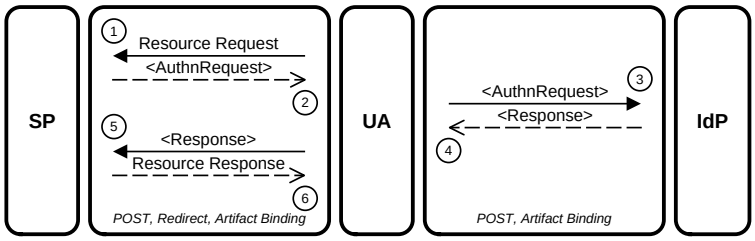

Figure 1: SAML Web Browser SSO profile.

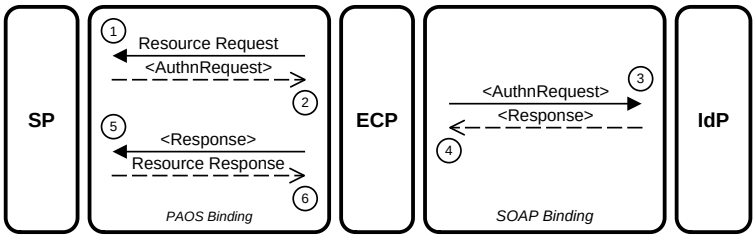

Figure 2: SAML ECP profile.

the AuthnRequest to the IdP and performs the user authentication. The result of the authentication is returned in a SAML Response in Step 4. In Step 5 either the POST or Artifact binding can be used by the UA to transmit the Response to the SP. Finally, in Step 6 the SP transmits the requested resource to the UA and the SAML protocol finishes.

The SAML ECP [HCH+05, Section 4.2] profile is a single sign-on authentication profile and specifies a client application which is capable of directly determine and contact the user's IdP, without getting redirected by the SP. It is particularly useful for client-side and server-side applications with a fixed set of services. The ECP profile focuses on applications with enhanced functionality, for instance, supporting more sophisticated protocols and bindings like SOAP and PAOS. As illustrated in Figure 2, the ECP profile comprises an ECP application, a SP, and an IdP. The application flow starts with a service or resource request to the SP by the ECP using the PAOS binding. The SP responses with an SAML AuthnRequest in Step 2. In Step 3 the ECP determines the IdP and transmits the AuthnRequest using the SOAP binding to the IdP and performs the user authentication. The result of the authentication is returned in the SAML Response in Step 4. The ECP conveys the Response to the SP in Step 5. Finally, in Step 6 the SP transmits the requested resource to the ECP and the SAML protocol finishes.

3 SAML Privacy-Enhancing Profile

The SAML Privacy-Enhancing (PE) profile enables users to consume services through a web browser and use a client application for strong authentication. It provides a user-controlled data flow, which allows users to cancel the authentication process at any point in time. Furthermore, it provides detailed information about the participants and the revealed attributes to enable the user to give an informed consent for the authentication.

The profile is based on SAML Web Browser SSO and the profile defined in TR-03124-1. We extend the SAML AuthnRequest to include the additional information about the participants and the authentication (cf. Section 4). Most of this information (e.g., requested attributes and information about IdPs) is expressed using the standardized Metadata for SAML [CMPM05] and the SAML Metadata Extensions for Login and Discovery User Interface [Can12]. Our profile uses plain HTTP mechanisms rather than SOAP to simplify the integration in web applications and existing SAML libraries.

We note that our usage of SAML metadata to carry this information is motivated by the FutureID Authentication Request [BHS+14].

3.1 Setting

The setting comprises the following entities: (i) the Service Provider (SP), (ii) the User Agent (UA), (iii) the Enhanced Client Application (ECA), and, optionally, (iv) an Identity Provider (IdP). The SP provides a service like an online shop and requires a user authentication. The UA is running on the user platform and is usually a plain-vanilla web browser.

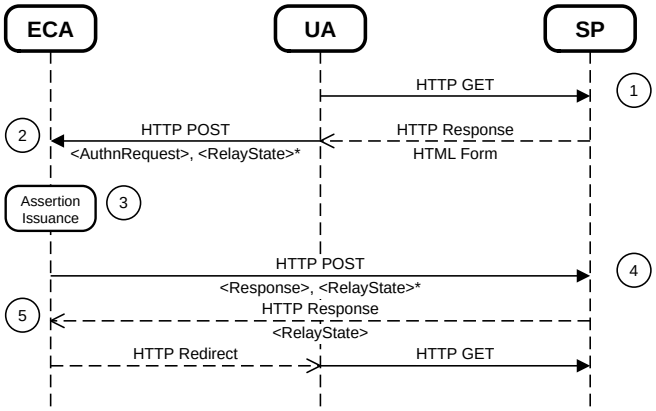

Figure 3: Protocol flow of the SAML PE profile.

The ECA is a universal application enabling authentication with various credentials and technologies. To be precise, it implements the user interaction, the authentication and transport protocols, the communication with credentials, and so forth. The ECA comprises a local HTTP-based interface which provides means to start authentication procedures or to fetch status information like supported application functionality. The assertion is optionally provided by an IdP, see below.

3.2 Protocol Flow

The protocol flow of the PE profile comprises five steps and is illustrated in Figure 3. In the following we describe the steps in detail.

Step 1 – Service Request First, the user navigates its UA to a protected service or resource. The authentication process begins.

Step 2 – Issuance Request The SP returns a HTML form to the UA which includes the extended `AuthnRequest` and an optional `RelayState`[1] [CHK+05, Section 3.5.3]. Both are forwarded to the ECA by submitting the HTML form via HTTP POST to the local HTTP-based interface[2].

The `AuthnRequest` represents the request from the SP to the IdP to issue an assertion about the user. It also includes detailed information of the participants and the authentication to enable the user to give an informed consent. The `RelayState` references to state

[1] Please note that the `RelayState` might cause severe security issues, therefore we recommend to protect it with confidentiality and integrity protection [HPM05, Section 6.4.6].

[2] `http://127.0.0.1:24727/eID-Client`

information stored at the SP to enable a redirect to the requested service or resources after the authentication.

Step 3 – User Consent and Assertion Issuance The ECA then presents all information about the authentication and involved participants to the user and prompts him or her to select an authentication method and/or an IdP. The ECA obtains this information from the proposed SAML `AuthnRequest` extension (cf. Section 4). Based on the displayed information the user is able to give an informed consent for the authentication.

Subsequently, the ECA fetches the assertion, i.e., SAML `Response`. Depending on the authentication method this can be done, e.g., by performing the Authentication Request Protocol [CKPM05, Section 3.4] with an IdP or some local assertion generation based on attributed-based credentials. The used protocol and authentication method is out of scope of the PE profile, so that a wide variety of methods can be supported and new methods can be added easily.

Step 4 – Assertion Delivery The ECA conveys the `Response` together with the `RelayState`, if received in Step 2, to the SP. The delivery of the assertion by the ECA is necessary to provide means for secure bindings (cf. Section 5), because the channel specific parameters can in general not be securely transferred from the ECA to the UA.

Step 5 – RelayState Processing In response to the successful verification of the assertion the SP returns a `RelayState` to the ECA. This value is either the same as the one sent in the previous step, or a preconfigured value from the SP in case no `RelayState` was given. The process continues only after a successful validation of the RelayState's integrity protection.

Finally, the ECA responds to the request from the UA in Step 2 with an HTTP redirect to the received `RelayState`. The UA follows the redirect and gets access to the protected resource.

4 User Consent

The user consent, as described in the Section 3.2, Step 3, is the core of our SAML PE profile. Before any personal information is revealed or any communication with further services takes place the user gets detailed information and is able to give an informed consent for the authentication. The user chooses (i) the IdP he or she would like to perform the authentication with, (ii) which user credential (e.g., hardware token, software certificate, username/password) he or she would like to use (with the chosen IdP) for authentication, and (iii) the attributes that will be disclosed to the SP; or (iv) to abort the authentication procedure.

In the following sections we describe how the information about the authentication process is transmitted to the client and encoded in the message SAML Authentication Request.

4.1 Information about the Service Provider

The information of the SP is defined by a `SPSSODescriptor` element (cf. Listing 1). The requested attributes are part of a `AttributeConsumingService` element and defined as a list of `RequestedAttribute` elements. To display this information in a user-friendly way, we include an `UIInfo` element as defined in [Can12]. The information is used to display (localized) information (e.g., name and description) about the SP and in particular about the requested attributes to the user. Each requested attribute is represented by a `RequestedAttributeInfo` element (cf. Appendix A), which in particularly contains at least one `Purpose` element in which the SP must give a reason why this attribute is necessary for the provided service. Additionally, a URL for more information may be provided in the `InformationURL` element. Optionally, the index of the `AttributeConsumingService` may be given, if different services have different purposes for the same attributes. Finally, the `SPSSODescriptor` element includes at least one `AssertionConsumerService` element which defines different consumer services at the SP. For example, the SP in Listing 1 accepts SAML Bearer assertions.

```
<md:SPSSODescriptor
  protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
  <md:Extensions>
    <mdui:UIInfo>
      <mdui:DisplayName xml:lang="en">SP1</mdui:DisplayName>
      <mdui:Description xml:lang="en">Description.</mdui:Description>
      <pe:RequestedAttributeInfo AttributeName="urn:oid:2.5.4.42">
        <pe:Purpose xml:lang="en">To call you.</pe:Purpose>
      </pe:RequestedAttributeInfo>
      <pe:RequestedAttributeInfo AttributeName="urn:oid:2.5.4.41">
        <pe:Purpose xml:lang="en">Enhanced user experience.</pe:Purpose>
      </pe:RequestedAttributeInfo>
    </mdui:UIInfo>
  </md:Extensions>
  <md:AssertionConsumerService
    index="0" isDefault="true" Location="https://sp1.example.com/saml"
    Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST">
  </md:AssertionConsumerService>
  <md:AttributeConsumingService index="0" isDefault="true">
    <md:ServiceName xml:lang="en">SP1</md:ServiceName>
    <md:RequestedAttribute
      Name="urn:oid:2.5.4.42" isRequired="true" FriendlyName="Forename"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
    </md:RequestedAttribute>
    <md:RequestedAttribute
      Name="urn:oid:2.5.4.41" isRequired="false" FriendlyName="Name"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
    </md:RequestedAttribute>
  </md:AttributeConsumingService>
</md:SPSSODescriptor>
```

Listing 1: Metadata of a SP. Namespaces: `md = urn:oasis:names:tc:SAML:2.0:metadata` is defined in [CMPM05], `mdui = urn:oasis:names:tc:SAML:metadata:ui` is defined in [Can12], `samlp = urn:oasis:names:tc:SAML:2.0:protocol` is defined in [CKPM05], and `pe = urn:oasis:names:tc:SAML:profile:privacy` is the namespace for the SAML PE profile.

4.2 Information about the Identity Providers

The information of an IdP is defined by a `IDPSSODescriptor` element (cf. Listing 2). To provide detailed information about the IdP to the user the `IDPSSODescriptor` element contains, in its extensions element, a `UIInfo` element. It includes information such as the name and a description of the IdP. One can imagine of including further details like terms of usage, data privacy statement, and so forth.

The `IDPSSODescriptor` element comprises at least one `SingleSignOnService` element, which is used to define different assertion issuance services at the IdP. The IdP in Listing 2 for instance issues SAML Bearer assertions.

Existing SAML metadata definitions do not allow to describe possible authentication

```
<md:IDPSSODescriptor
  protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
  <md:Extensions>
    <mdui:UIInfo>
      <mdui:DisplayName xml:lang="en">IdP1</mdui:DisplayName>
      <mdui:Description xml:lang="en">Description.</mdui:Description>
      <mdui:PrivacyStatementURL xml:lang="en">
        https://idp1.example.com/privstat.html
      </mdui:PrivacyStatementURL>
    </mdui:UIInfo>
  </md:Extensions>
  <md:SingleSignOnService
    Location="https://idp1.example.com/saml/remoteauth"
    Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST">
    <pe:AuthenticationOptions>
      <pe:AuthenticationOption
        index="0" Binding="urn:oid:1.3.162.15480.3.0.25">
        <pe:Accepts>
          <pe:CredentialList>
            <pe:CredentialEntry credentialType="eID-GOV-DE-v1.0"/>
            <pe:CredentialEntry credentialType="eID-gov-GB-v1"/>
          </pe:CredentialList>
        </pe:Accepts>
      </pe:AuthenticationOption>
      <pe:AuthenticationOption index="1"
        Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST">
        <pe:Accepts>
          <samlp:Scoping>
            <samlp:IDPList>
              <samlp:IDPEntry ProviderID="http://idp2.example.com"/>
            </samlp:IDPList>
          </samlp:Scoping>
        </pe:Accepts>
      </pe:AuthenticationOption>
    </pe:AuthenticationOptions>
  </md:SingleSignOnService>
</md:IDPSSODescriptor>
```

Listing 2: Metadata of an IdP. See Listing 1 for namespace definitions.

options at the IdP, which allows the user to authentication with different credentials. However, this information is crucial for the user to make an informed decision (i.e., to decide which user credential to use at which IdP). To provide this information, we define the new element `AuthenticationOptions` (cf. Appendix B), to be used in `SingleSignOnService`.

Every authentication option has an attribute `Binding` which defines the protocol of the authentication (e.g., TLS with X.509 client certificates or providing a SAML Bearer assertion from another IdP) and a list of accepted credential types (`CredentialEntry`) and/or IdPs (`IDPEntry`). Accepted IdPs are listed using the `Scoping` element.[3] The ECA must be able to obtain metadata for these IdPs as well (see Section 4.3). Accepted credentials are listed using the `CredentialList` element.

For example, the IdP in Listing 2 provides two authentication options: (i) Users may authenticate themselves using TLS with X.509 client certificates (provided by smart cards of different types) or (ii) they may authenticate themselves by presenting a SAML Bearer assertion issued by another IdP.

4.3 The SAML `AuthnRequest`

We extend the SAML `AuthnRequest` to provide the user with detailed information about the authentication. This information is provided as additional metadata.

The information, i.e. the metadata, of the SP is included by using an `EntityDescriptor` element. The `entityID` attribute of this element matches the `Issuer` of the request. The `EntityDescriptor` element contains the `SPSSODescriptor` element as described in Section 4.1.

The metadata of the IdP is included in the same way. The example in Listing 3 includes two additional `EntityDescriptor` elements which includes an `IDPSSODescriptor` element providing information about the IdPs.

Finally, the `Scoping` element is contained in the authentication request which contains the list of IdPs that are accepted by the SP. Please note that the SP might not accept assertions from all IdPs. However, the `AuthnRequest` must contain the metadata of all participants which might be involved in the authentication procedure. In the example in Listing 3 the SP only accepts assertions from the IdP 1. However, IdP 1 accepts assertions from IdP 2, thus, the metadata of IdP 2 must also be included in `AuthnRequest`. For privacy reasons it is very important that the SP resolves the transitive trust relation of the IdPs and provide all necessary metadata directly in the `AuthnRequest`.

```
<samlp:AuthnRequest
    IssueInstant="2014-04-22T12:00:00Z" Version="2.0"
    ID="b07b804c-7c29-ea16-7300-4f3d6f7928ad">
  <saml:Issuer>https://sp1.example.com/</saml:Issuer>
  <samlp:Extensions>
```

[3] We note that the `Scoping` element is defined in [CKPM05] and used in the `AuthnRequest` to specify accepted IdPs.

```
    <md:EntityDescriptor entityID="https://sp1.example.com/">
      <!-- Metadata of service provider 1 -->
    </md:EntityDescriptor>
    <md:EntityDescriptor entityID="http://idp1.example.com/">
      <!-- Metadata of identity provider 1 -->
    </md:EntityDescriptor>
    <md:EntityDescriptor entityID="http://idp2.example.com/">
      <!-- Metadata of identity provider 2 -->
    </md:EntityDescriptor>
  </samlp:Extensions>
  <samlp:Scoping>
    <samlp:IDPList>
      <samlp:IDPEntry ProviderID="http://idp1.example.com/"/>
    </samlp:IDPList>
  </samlp:Scoping>
</samlp:AuthnRequest>
```

Listing 3: A SAML AuthnRequest for the SAML PE profile. See Listing 1 for namespace definitions.

One can argue that the additional metadata of the SP and IdPs cause a lot of overhead. The amount of data can be reduced if the metadata is not included directly, but linked to a file stored at each IdP. However, then the ECA needs to fetch the metadata from each IdP which rise privacy issues.

5 Channel Binding

The commonly used SAML Web Browser SSO profile is susceptible to theft of the Bearer Token [HPM05]. In detail, if adversaries are able to steal the authentication assertion they can impersonate the user. To overcome this problem an assertion is bound to the service request as specified in the Holder-of-Key (HoK) profile for SAML. The user's web browser establishes a TLS [DR08] channel using a client certificate to the SP for requesting a resource. The browser uses the same certificate for the communication with the IdP to perform the user authentication. The IdP includes a reference of the certificate into the assertion. Thus, only the holder of the private key associated with the certificate is able to use the assertion at the SP for authentication. If an adversary steals the assertion, she cannot use it because she does not possess the required private key.

One of the key pillars is that the browser uses the same certificate for the TLS channel to the SP and to the IdP. In our scenario we face the challenge that we have two different applications, the browser, which requests the resource from the SP, and the ECA, which performs the user authentication and communicates with the IdP. To enable a channel binding as described in the HoK profile for SAML, we must use the same certificate in both applications and both TLS channels, respectively.

The simplest way would be to share the same key store for both applications. However, to provide enhanced privacy, we prefer ephemeral certificates which have a very short lifetime and are only used with a single SP. The certificates are self-signed and created on-

demand. The idea is that the ECA creates ephemeral certificates for each SP and makes them available to the browser for the certificate-based TLS authentication.

We propose to use a PKCS#11 [RSA97] module for web browsers, which allows the ECA to act as a cryptographic device which provides tokens to the browser, which in turn can be use for the certificate-based TLS authentication. In essence, when the browser establishes the TLS channel to the SP it queries all PKCS#11 modules for available tokens. The ECA application creates an ephemeral certificate and returns it to the browser. The same certificate is then used by the ECA for the TLS channel to the IdP.

6 Conclusion

We presented the SAML Privacy-Enhancing profile which supports common service usage through the web browser as well as local client applications to provide means for strong authentication. The profile is based on the existing SAML standard and extends the Authentication Request Protocol. This allows for an easy integration in existing SAML infrastructures. It provides detailed information for the user to give informed consent for the authentication. It also tackles privacy issues related to SAML by giving the user the control of the application flow. Furthermore, it supports secure channel binding to prevent Man-In-The-Middle attacks between the ECA and the SP.

A RequestedAttributeInfo

```
<element name="RequestedAttributeInfo">
  <complexType>
    <sequence>
      <element name="Purpose" type="md:localizedNameType"
               maxOccurs="unbounded" />
      <element name="InformationURL" type="md:localizedURIType"
               minOccurs="0" maxOccurs="unbounded" />
    </sequence>
    <attribute name="AttributeName" type="string" use="required" />
    <attribute name="AttributeConsumingServiceIndex"
               type="unsignedShort" />
  </complexType>
</element>
```

B AuthenticationOptions

```
<element name="AuthenticationOptions">
  <complexType>
    <sequence>
      <element name="AuthenticationOption" maxOccurs="unbounded"
               type="pe:AuthenticationOptionType" />
    </sequence>
  </complexType>
```

```
</element>
<complexType name="AuthenticationOptionType">
  <sequence>
    <element name="Accepts" type="pe:AcceptsType" />
  </sequence>
  <attribute name="index" type="unsignedShort" use="required" />
  <attribute name="isDefault" type="boolean" use="optional" />
  <attribute name="Binding" type="anyURI" use="required" />
</complexType>
<complexType name="AcceptsType">
  <choice>
    <element ref="samlp:Scoping" />
    <element name="CredentialList" type="pe:CredentialListType" />
  </choice>
</complexType>
<complexType name="CredentialListType">
  <sequence>
    <element name="CredentialEntry" type="pe:CredentialEntryType"
             maxOccurs="unbounded" />
  </sequence>
</complexType>
<complexType name="CredentialEntryType">
  <attribute name="CredentialType" type="anyURI" use="required" />
</complexType>
```

References

[AMHAJ+10] Joaquín Alcalde-Moraño, Jorge López Hernández-Ardieta, Adrian Johnston, Daniel Martinez, Bernd Zwattendorfer, Marc Stern, and John Heppe. D5.8.3b Interface Specification. STORK Deliverable, D5.8.3b, November 2010. https://www.eid-stork.eu/index.php?option=com_processes&Itemid=&act=streamDocument&did=1880.

[BHS+14] Bud P. Bruegger, Detlef Hühnlein, Johannes Schmlz, Monika Drabik, Nuria Ituarte, Eray Özmü, Jan Camenisch, Gregory Neven, Meiko Jensen, Jaap-Henk Hoepman, Heiko Ronagel, and Sebastian Kurowski. Reference Architecture. FutureID Deliverable, D21.4, Version 1.1, April 2014. http://futureid.eu/data/deliverables/year1/Public/FutureID_D21.04_WP21_v1.1_Reference%20Architecture.pdf.

[BSI14] Technical Guideline TR-03124-1 eID-Client – Part1: Specifications. Federal Office for Information Security, Version 1.1, 2014. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03124/TR-03124-1.pdf.

[Can12] Scott Cantor. SAML V2.0 Metadata Extensions for Login and Discovery User Interface Version 1.0. OASIS Standard, Apr 2012.

[CHK+05] Scott Cantor, Frederick Hirsch, John Kemp, Rob Philpott, and Eve Maler. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, Mar 2005.

[CKPM05] Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, Mar 2005.

[CMPM05] Scott Cantor, Jahan Moreh, Rob Philpott, and Eve Maler. Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, Mar 2005.

[DR08] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), Aug 2008. Updated by RFCs 5746, 5878, 6176.

[HCH+05] John Hughes, Scott Cantor, Jeff Hodges, Frederick Hirsch, Prateek Mishra, Rob Philpott, and Eve Maler. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, Mar 2005.

[HPM05] Frederick Hirsch, Rob Philpott, and Eve Maler. Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, Mar 2005.

[KS10] Nate Klingenstein and Tom Scavo. SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0. OASIS Standard, Aug 2010.

[RSA97] RSA Laboratories. PKCS #11: Cryptographic Token Interface Standard, Apr 1997.