Documentación técnica de ID Uruguay

Autor
Área de Seguridad de la información de Agesic

Fecha de creación
11/05/2023

Tipo de publicación
Guía técnica

Resumen

Documentación técnica de ID Uruguay y Usaurio.gub.uy

ID Uruguay: integración con SAML

¿Qué es?

ID Uruguay es un sistema que te permite, mediante una única cuenta, acceder a todos los trámites y servicios digitales del Estado sin necesidad de registros ni contraseñas adicionales.

La integración con ID Uruguay se puede lograr a través del protocolo SAML o el protocolo OpenID Connect, en su perfil Web SSO, y sus particularidades son el objeto de la presente documentación. Este documento explica la integración con SAML.

Terminología

A continuación, se definen los actores y elementos según la nomenclatura SAML y otros términos útiles.

Identity Provider (IDP): Es el proveedor de identidades, al cual se le delega la autenticación.

Service Provider (SP): Es la aplicación web/servicio de negocio que actuará como cliente, delegando la autenticación al*IDP*. O sea, es el sistema que se quiere integrar como, por ejemplo, el portal de un organismo o una empresa.

User ID (UID): Identificador de un usuario del sistema. Este será del tipo xx-yyy-zzzzzzz, donde xx es el código ISO del país, yyy el código del tipo de documento (dni, psp, ci) y z el numero de documento. Por ejemplo, un usuario con Cédula de Identidad uruguaya 1231231-4 tendrá como identificador uy-ci-12312314. Por mayor información sobre los tipo de usuarios que presentará el sistema, accedé a ID Uruguay

Assertion: Mensaje codificado en XML que se utiliza en el protocolo SAML para hacer pedidos y confirmaciones.

¿Cómo funciona?

El objetivo del sistema es delegar la autenticación en el IDP mediante el intercambio assertions. A los distintos mecanismos de comunicación se los conocen como bindings.

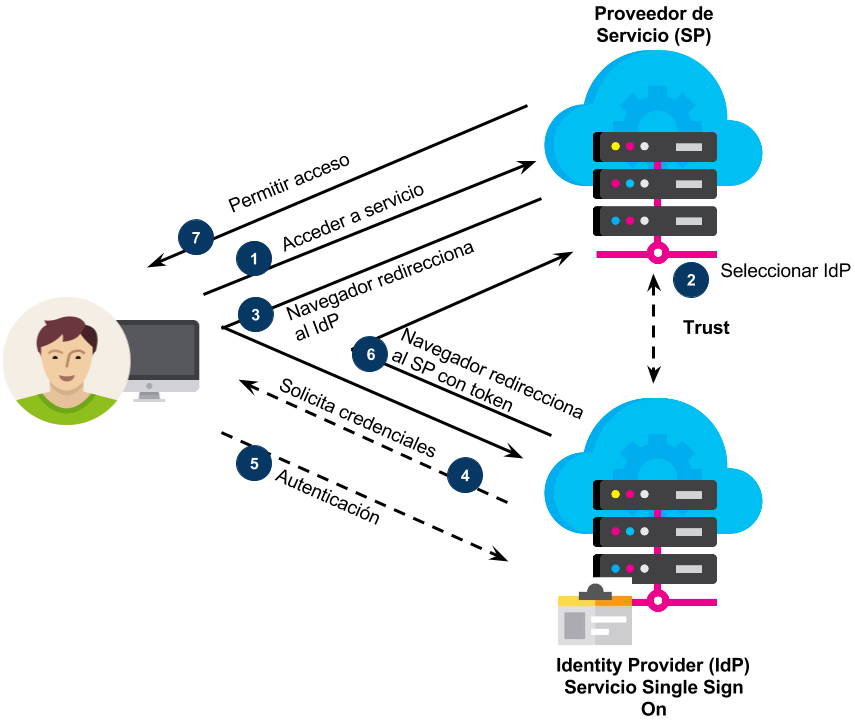
Sign On

El Sign On consiste en el pedido de autenticación del usuario por parte de un SP (aplicación) al IDP. En este sistema solo se autenticarán personas. Si la autenticación es exitosa, el IDP registra una sesión para ese usuario (en el contexto del explorador web) y comunica al *SP* el resultado de la autenticación, que podrá realizar un login local si el usuario cumple con las condiciones. En el caso ideal cuando el usuario se autentica de forma exitosa, se establece una sesión tanto en el *IDP* como en el *SP*.

Un Sign On se desarrolla en los siguientes pasos:

- (1) El usuario desea autenticarse a través de un SP (por ejemplo: portal web de un organismo), este enviará un Authentication Request Assertion al IDP, solicitando la autenticación del usuario que se encuentra utilizando el sistema. *
- El usuario entonces es redirigido al IDP (2) y (3) y éste se encarga de autenticarlo, con usuario y contraseña o con la Cédula Digital (4) y (5).
- Si la autenticación es exitosa o no, el IDP generará un Response Assertion que enviará al SP indicando el resultado de la autenticación (6) y (7).

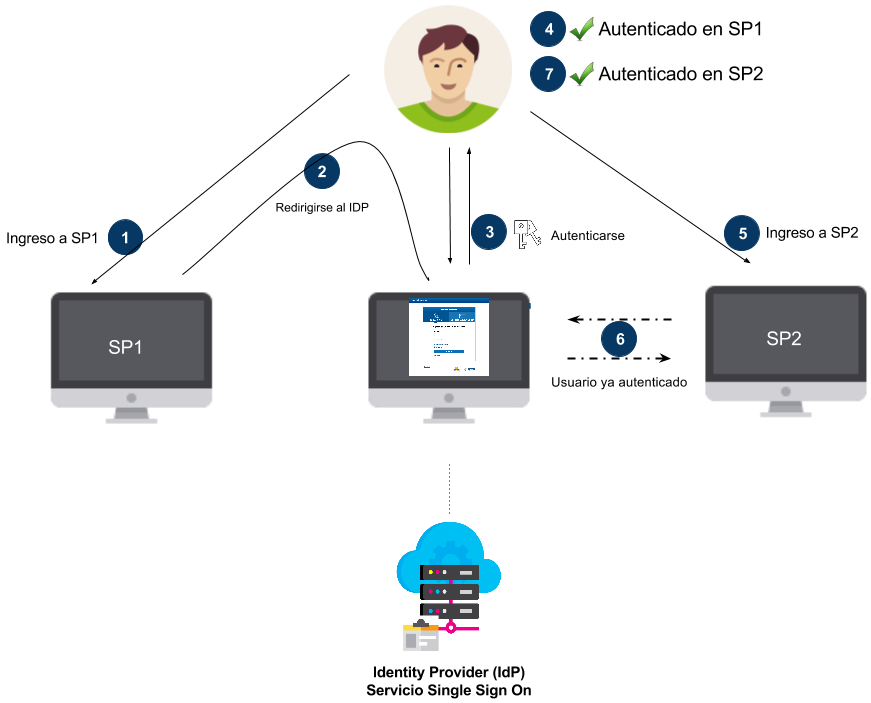
La siguiente imagen ilustra un caso genérico de autenticación utilizando el protocolo SAML.



Cuando el usuario desea desloguearse, lo solicitará en la aplicación en que se encuentre y la misma enviará al IDP un Logout Assertion. Luego de procesarla el IDP retornará un assertion con el resultado del deslogueo.

Single Sign On (SSO)

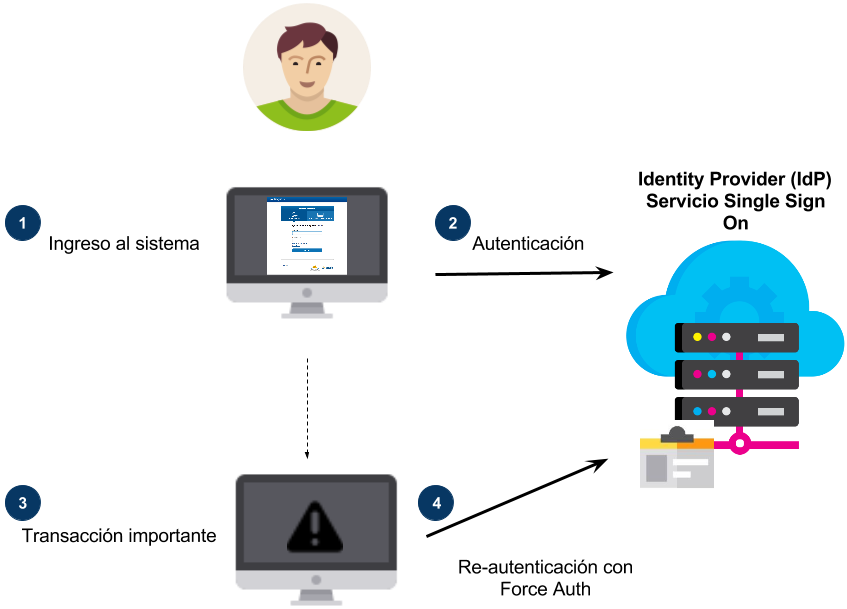
Ademas de la delegación de la autenticación, este sistema permite la implementación de un SSO. El Single Sign On consiste en que el usuario puede ingresar a varios servicios autenticándose una sola vez. Esto funciona gracias a que el IDP es compartido entre todos los SP.



Por lo tanto, si un usuario se autenticó para ingresar a, por ejemplo, SP1 y va a ingresar a SP2, al iniciar sesión en este último quedará autenticado sin tener que volver a ingresar las credenciales. Lo que pasa por detrás es que SP2 solicita la autenticación al IDP y este, como ya tiene una sesión activa, no le volverá a solicitar las credenciales.

Forzar Autenticación (Force Auth)

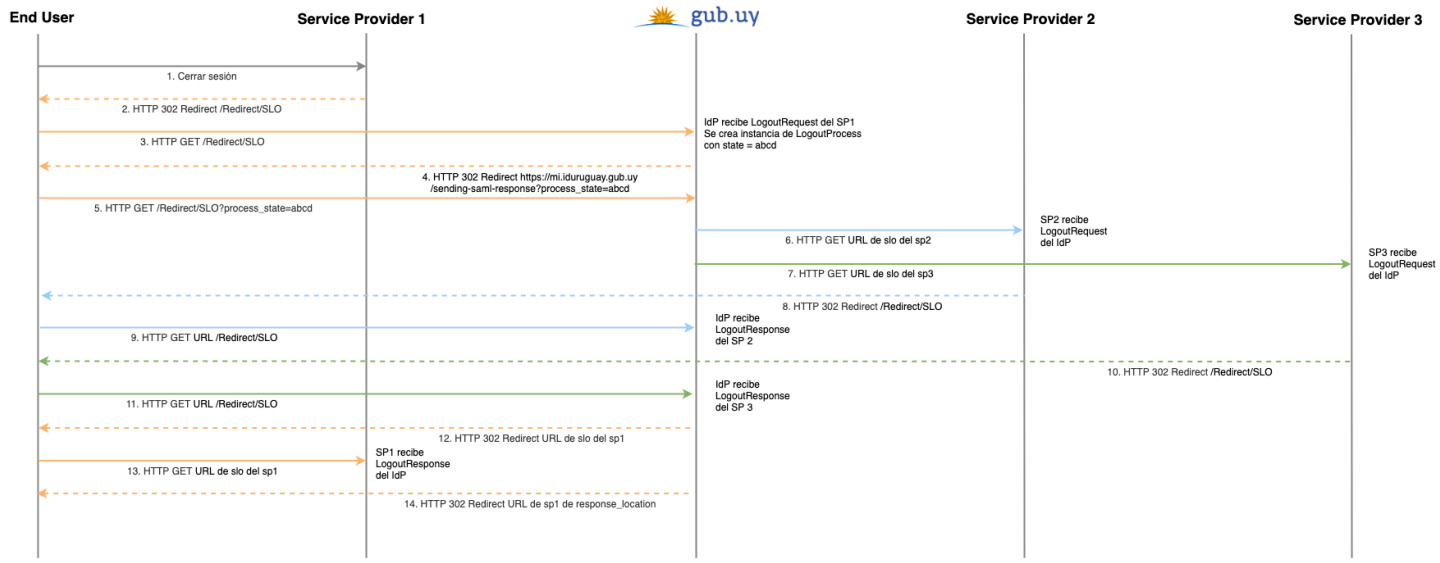
Si en el ejemplo del SSO, SP2 necesitase que el usuario confirme que es él quien vuelve a ingresar las credenciales, SP2 lo puede solicitar enviando nuevamente un Autentication Request con el atributo "*ForceAuth*" en el assertion. Este proceso es muy común cuando se va a acceder a funcionalidades o transacciones de alto riesgo para el SP y por lo tanto se quiere estar seguro de la vigencia del usuario que está utilizando el sistema. El usuario que se estuviera autenticando nuevamente con ForceAuth debe de ser el mismo que tiene la sesión activa, en caso contrario no podrá autenticarse. Si se desea autenticar un nuevo usuario, se deberá primero realizar un Single Logout.



Single Log Out (SLO)

Así como se implementa el SSO, análogamente se implementa el Single Logout. En un escenario donde se tienen varios SP autenticados con el mismo IDP; cuando el usuario se desloguea en alguno de ellos, este puede enviar un Single Logout Assertion al IDP, y este se encargará de solicitar el deslogueo local a cada uno de los SP en los que estaba autenticado, quienes deberán obligatoriamente aceptar ese pedido y efectivizar el cierre de la sesión.

SAML SLO



Registro de Sistema Informático para integrarse a ID Uruguay utilizando SAML

Para solicitar la integración, debe ingresar la solicitud en un trámite disponible acá:https://ioti.bpmgob.agesic.red.uy/simple/tramites/disponibles

Seleccionar la opción " Integracion ID Uruguay SamL 2.0 "
Para esto es necesario:

- Contar con una identificación digital de ID Uruguay al menos con nivel intermedio.
- Ingresar al trámite a través de la Red Uy (en otro caso por favor contactarnos).

Es necesario realizar el trámite para integrarse primero a nivel de testing y luego para producción ya que son conexiones y credenciales diferentes.
Para el caso de producción, el trámite debe ser realizado por el funcionario público responsable técnicamente del organismo interesado.
Por cualquier duda, por favor contactarse a soporte@agesic.gub.uy con copia a identificacion.electronica@agesic.gub.uy .

Implementación

El protocolo utilizado para delegar la autenticación será SAML (Security Assertion Markup Language) en su versión 2.0. A continuación, se ofrece una descripción de la implementación especifica del IDP y la información que se intercambia.

Datos recibidos de los assertions

Al solicitar al IDP que el usuario se autentico, este nos devolverá un assertion con información del usuario (atributos), si este se autenticó correctamente.

Los atributos recibidos en el assertion cuando el usuario se autenticó con la Cédula de Identidad Digital serán:

- **UID**: Identificador del usuario del tipo uy-dni-zzzzzzz, donde z es el número de la Cédula de Identidad.
- **x509Certificate**: Certificado en base64 y formato PEM incluido en la Cédula Digital.
- **Nombre completo**: Nombre completo encontrado en el certificado de la cédula.
- **País** / **documento**: uy.
- **Documento**: Número documento, igual a las zzzzzzzz del UID.
- **Tipo de documento**: dni
- **Certificado**: Este atributo siempre vendrá en true cuando la autenticación se realiza con cédula.

En cambio, cuando la autenticación es con usuario y contraseña:

- **UID**: Identificador del usuario del tipo xx-yyy-zzzzzzz donde xx es el código ISO del país, yyy el codigo del tipo de documento (dni, psp, ci) y z el número de documento.
- **Certificado**: Este campo ira en *true* si el usuario asocio la Firma Electrónica Avanzada a su usuario, dando garantías de que es quien dice ser. De lo contrario, se encontrará en *false* .
- **Presencial**: Este campo irá en *true* si el usuario se registro y presento la Cédula de Identidad presencialmente.
- **Primer nombre**: Primer nombre del usuario autenticado.
- **Segundo nombre** (opcional): Segundo nombre del usuario autenticado.
- **Primer apellido**: Primer apellido del usuario autenticado.
- **Segundo apellido**: Segundo apellido del usuario autenticado.
- **País del documento**: El mismo xx del UID.
- **Documento**: Las mismas z que el UID.
- **Tipo de documento**: Código definido por la UNAOID para el tipo de documento. En el caso de la Cédula de Identidad uruguaya, será *68909* .

Ejemplo del conjunto de atributos que recibirá el SP cuando la autenticación se realiza con Cédula Digital:

<saml2:AttributeStatement> <saml2:Attribute FriendlyName="Certificado" Name="Certificado" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">true</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="X509 Certificate" Name="X509Certificate" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
MIIFYTCCA0mgAwIBAgIQI4v+eauXgN9VXJVyswPMEjANBgkqhkiG9w0BAQsFADBtMTwwOgYDVQQD
EzNBdXRvcmlkYWQgQ2VydGlmaWNhZG9yYSBkZWwgTWluaXN0ZXJpbyBkZWwgSW50ZXJpb3IxIDAe
BgNVBAoTF01pbmlzdGVyaW8gZGVsIEludGVyaW9yMQswCQYDVQQGEwJVWTAeFw0xNTA1MjAxNDA4
NTBaFw0yMDA1MjAxNDA4NTBaMEMxCzAJBgNVBAYTAlVZMRQwEgYDVQQFEwtETkk0MjUwMjY0ODEe
MBwGA1UEAwwVTklDT0xBUyBQSVFVRVJFWiBSQU1BMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB CgKCAQEA0pJkfT6ZMQ8k373+4vB13nZ8NXX4oVizNH03kH3YhUN0RedRG2F//l6AOiltyjbY+T/7
jkte2Z2pAXzxTzd5erqSeqg5C/T4Ynq1mkHBMOAYCiw4oRQ6qJaPBKQEZ+32IH3J2H6lgtW2IIS6 ES7A7dOFE/wkCnE65OryesVqXng2DixSGxCWE/1KG0RFqOAxzgBicW6MZbZMqfX0XlON9pG4XYep
P4rqp0xHRucFl7Z7Zc3Yz9gSERea8BjGD6k7cxKMaPbPhQsTfABahX1xS0epzsz/GrcZ5LYWQXVi XTSQ7ohv/u167y5a1gQJq1qaJ1afyqqKMx17F+bux0LY1wIDAQABo4IBJTCCASEwCQYDVR0TBAIw
ADAOBgNVHQ8BAf8EBAMCBPAwHQYDVR0lBBYwFAYIKwYBBQUHAwIGCCsGAQUFBwMEMB0GA1UdDgQW
BBQ3uj603gPiyIbNyWQVs97HT2PM1TAfBgNVHSMEGDAWgBSfszQ8mzqpYBDKyFkEbe2tYixl/zBq
BgNVHSAEYzBhMF8GC2CGWoTirh2EiAUCMFAwTgYIKwYBBQUHAgEWQnd3dy51Y2UuZ3ViLnV5L2lu
Zm9ybWFjaW9uLXRlY25pY2EvcG9saXRpY2FzL2NwX3BlcnNvbmFfZmlzaWNhLnBkZjA5BgNVHR8E MjAwMC6gLKAqhihodHRwczovL2NhLm1pbnRlcmlvci5ndWIudXkvY3Jscy9jcmwuY3JsMA0GCSqG
Slb3DQEBCwUAA4ICAQB5gBDFHuvgNkAEIhdYxUAYDTvwRf6alnVn6ALccPerA7a2Q4cZQSZn2RZt yfH+0c01OMuDSonDXEPzIFmpYGy9lVZclk7j/+NNQCr+5MBaAzcgUGgHgT+xGkSxbGqGcg//x2dl
U3liX8MiL49/PA4mqmZaL5RZnPt0nH5oqRrrlJTtZTQBISCR+B6Ap/QF1bOCSEFd3rlpbxC3Fv+L XXmYU6S9RSglBrihth+HPpZ9zWyKlUepLJt78BoAkSJLH+E56rbyqbkC1zk2pw1GHJntyBu0j7DD
di69K4+Gulh1PBqfvEdbSPpDi6JuQSZZKG1IyhOLFm1Bm8szl6Rmxji64uf4sjZrK2JIrE4BaV9Z dvbdB10D/R3qCMg1I2scjCbogCwN+0e6dJO6BOWIEUc07YX6YH75JBQmOKdh9D7FzJM4fIDezxYg
wmxDogaifqx/2N4kTfZ0L1I5mN6L3Iqme7W6hlriCQei8tp43v13/6yltZLWk8V4xvIND6JGwEwC BnFLXMZk9c6TI3lhPXYSoZT2e6nenc3Cx8zZyyOZkIRHvCQolRebr7VZJ/4ZHuouUYatcri9w24m
QNi9nVCzDCsC6DuuRk6fT363XLPnS4/FHQVLHk81QC8Ak2DJXtoICO6wU8jeiPxmdzbWU8xXcebb c4WbE1qAgglHkYyoaQ==</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute
FriendlyName="UID" Name="uid" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">uy-dni-12312314</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Nombre Completo" Name="NombreCompleto" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">rodrigo perez

suarez</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Pais Documento" Name="PaisDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">uy</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Tipo Documento" Name="TipoDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">dni</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Documento" Name="Documento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">12312314</saml2:AttributeValue> </saml2:Attribute> </saml2:AttributeStatement>

Ejemplo de conjunto de atributos cuando la autenticación se realizó con usuario y contraseña:

<saml2:AttributeStatement> <saml2:Attribute FriendlyName="Certificado" Name="Certificado" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">false</saml2:AttributeValue></saml2:Attribute> <saml2:Attribute FriendlyName="Segundo Appelido" Name="SegundoApellido" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Suarez</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="UID" Name="uid" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">uy-ci-12312314</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Primer Nombre" Name="PrimerNombre" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Rodrigo</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Pais Documento" Name="PaisDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">uy</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Primer Appelido" Name="PrimerApellido" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Perez</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Tipo Documento" Name="TipoDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">68909</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Documento" Name="Documento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">12312314</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Presencial" Name="Presencial" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">false</saml2:AttributeValue> </saml2:Attribute> </saml2:AttributeStatement>

Niveles de confianza de los usuarios autenticados

Habrá cuatro tipos de niveles de confianza en los usuarios autenticados, según como se hayan realizado la autenticación o el registro.

1. **Usuarios autenticados con Cédula de Identidad Digital:** Son los usuarios que al momento de realizar la autenticación usaron la cédula electrónica ingresando su PIN. Tiene el nivel de confianza máximo que el sistema puede dar, ya que se autenticaron con algo que poseen (cédula) mas algo que saben (PIN). Para identificar un usuario de este tipo, se debe chequear el contexto de autenticación en el *response assertion* y que coincida con *<saml2:AuthnContext<saml2:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:**SmartcardPKI** </saml2:AuthnContextClassRef></saml2:AuthnContext>*. Adicionalmente el identificador será del tipo **uy-dni-zzzzzzzz** .
2. **Usuarios autenticados con usuario y contraseña que firmaron contrato con Firma Electrónica Avanzada:** Son los usuarios que se autorregistraron o se registraron presencialmente y firmaron un acuerdo con su Firma Electrónica Avanzada. Se chequea que la firma corresponda con el documento del UID, por lo que posee un alto nivel de confianza. Para identificar un usuario de este tipo, se debe chequear el contexto de autenticación en el response assertion y que coincida con *<saml2:AuthnContext><saml2:AuthnContextClassRef> urn:oasis:names:tc:SAML:2.0:ac:classes:**PasswordProtectedTransport** </saml2:AuthnContextClassRef></saml2:AuthnContext>* y el atributo ***Certificado*** en *true*.
3. **Usuarios autenticados con usuario y contraseña registrados presencialmente:** Usuarios que se registraron en una ventanilla para tal fin y presentaron documentación (documento de identidad nacional o pasaporte). Posee un nivel de confianza alto, ya que la información fue corroborada y chequeada por un operador de la plataforma. Para identificar un usuario de este tipo, se debe chequear el contexto de autenticación en el *response assertion* y que coincida con *<saml2:AuthnContext><saml2:AuthnContextClassRef> urn:oasis:names:tc:SAML:2.0:ac:classes:**PasswordProtectedTransport** </saml2:AuthnContextClassRef></saml2:AuthnContext>* y el atributo ***Presencial*** en *true*.
4. **Usuario autorregistrado:** Usuario que se registró él mismo en la consola de autogestión. No tiene garantías de que es quien dice ser, ya que no se presenta prueba de que el usuario es poseedor del documento de identidad ingresado. Puede servir para casos donde no se necesita tener garantías sobre la identidad o donde luego esta se verificará (por ejemplo, consulta de información pública o un sistema de agenda). Para identificar un usuario de este tipo se debe chequear el contexto de autenticación en el *response assertion* y que coincida con *<saml2:AuthnContext><saml2:AuthnContextClassRef> urn:oasis:names:tc:SAML:2.0:ac:classes:**PasswordProtectedTransport** </saml2:AuthnContextClassRef></saml2:AuthnContext>* y con ambos atributos ***Certificado*** y ***Presencial*** en false.

Perfil y bindings

ID Uruguay utiliza el perfil "Web SSO" de SAML 2.0, que está pensado para usarse en navegadores web. Cuando el usuario desea autenticarse en un servicio, este generará un assertion SAML y redirigirá al usuario al IDP junto con el assertion. La respuesta del IDP se realizará de igual forma, con redirecciones del explorador.

Para este perfil existen dos bindings para enviar los mensajes: ***HTTP-POST*** y ***HTTP-Redirect***.

El binding ***HTTP-POST*** consiste en poner el assertion en un campo de un Post http dirigido hacia la contraparte.

En el binding ***HTTP-Redirect*** se envía el mensaje a la contraparte como parámetro en la URL de un Redirect (302).

La otra gran diferencia entre estos dos bindings es que en el caso de HTTP-POST la firma del assertion se realiza dentro del xml del assertion, con firma estandar xml. En cambio, en HTTP-Redirect se codifica el assertion sin firma en un base 64. El assertion se incluye en el parámetro SAMLRequest; luego, en el parametro SigAlg irá el algoritmo de firma y, finalmente, en el parámetro Signature irá la firma de los últimos dos parámetros concatenados. Luego, se realizará un URL encoding para poder enviarlos como parámetros en la URL.

La realización de la firma de esa manera en el binding HTTP-Redirect se debe a que queda más liviana (de tamaño); la principal limitación de este binding esta en el tamaño maximo de caracteres que se pueden colocar en una URL de un pedido http. Es por esto que se recomienda el binding *HTTP-POST*.

La limitación del binding **HTTP-POST** es que algunos firewalls podrían bloquear los pedidos de este tipo, pero en la actualidad esto es muy improbable.

Si bien el uso de un binding u otro no siempre es posible debido a restricciones como el uso de HTTPS y el tamaño de las assertions, estos son independientes del comportamiento del protoclo SAML, estableciendo únicamente la forma en que se intercambian los mensajes entre *SP* e IDP, y no afectando en nada el comportamiento de los flujos de autenticación o de las sesiones.

Como la comunicación se realiza con redirecciones del explorador, no se necesita establecer nunca una comunicación directa entre el *SP* y el IDP.

Contenido de los assertions

Los assertions son mensajes xml. Se usarán cuatro tipos de assertions: Authentication Request, Response, Logout Request y Logout Response assertion.

Authentication Request Assertion

Este assertion se enviará desde el SP al IDP para solicitar una autenticación.

A continuación, se ofrece un ejemplo ilustrativo del assertion, que luego será explicado. En este caso, se usó el *HTTP-POST* binding; por eso, la firma esta incluida en el xml en el nodo *Signature*. En el caso del binding *HTTP-Redirect* la firma no sería incluida en el xml (si como parámetro como fue explicado).

<?xml version="1.0" encoding="UTF-8"?> <saml2p:AuthnRequest AssertionConsumerServiceURL="https://eid-sp.portal.gub.uy/v1.1/sp-idp/saml/acs" AttributeConsumingServiceIndex="0" Consent="urn:oasis:names:tc:SAML:2.0:consent:obtained" Destination="https://eid.portal.gub.uy/v1.1/idp/profile/SAML2/POST/SSO" ForceAuthn="false" ID="_d06b8cde-3bea-43d2-be35-ce2e6ff7f3f4" IssueInstant="2016-02-04T16:24:51.496Z" ProviderName="http://sp1.ejemplo.com/" Version="2.0" xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"> <saml2:Issuer xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">http://sp1.ejemplo.com/</saml2:Issuer> <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"> <ds:SignedInfo> <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/> <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/> <ds:Reference URI="#d06b8cde-3bea-43d2-be35-ce2e6ff7f3f4"> <ds:Transforms> <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/> <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/> </ds:Transforms> <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/> <ds:DigestValue>7JryUO+7m+3HdPwFq06q8UCg6PE=</ds:DigestValue> </ds:Reference> </ds:SignedInfo> <ds:SignatureValue>X4CUZFzK5US/NEDzBihn399PVKC7JGcYJgpI3v2sUcM5Yhha1qVOg780Pq SYtQM2WgJZH5/HiMtwtICyg+wN10+o4W0SxIPmgLbRON4lGEkhvVXKnclxh3kWDgRNs8CnblsAdXO DClyhaUjSU0NLGQnNrU3D8HNcHbd28M2DYeQFz1wQF+/C0C6thc3057JvA1Q/HBj0Vbz1+xpIfQbg qRXUUlm4cMfDYecY2JbgrgowSuqqUTpYaWNksqKNz/20PW8LAn5IffD6NUcQl9MN28MI5ucqL2MX8 zABWoGNuMKfHRjc6kxpoJYrqMcKCs3dNwKESxZDsdiTaG4jyOoFKA==</ds:SignatureValue> <ds:KeyInfo> <ds:X509Data> <ds:X509Certificate> MIIDZTCCAk2gAwIBAgIBAjANBgkqhkiG9w0BAQsFADBTMQswCQYDVQQGEwJGUjEQMA4GA1UECgwH R2VtYWx0bzEMMAoGA1UECwwDSUNTMSQwIgYDVQQDDBtSJkQgRGV2IFNlcnZpY2UgUHJvdmlkZXIg Q0EwHhcNMTQwMzE3MDkxOTM5WhcNMjQwMzE0MDkxOTM5WjBKMQswCQYDVQQGEwJGUjEQMA4GA1UE CgwHR2VtYWx0bzEMMAoGA1UECwwDSUNTMRswGQYDVQQDDBJTZXJ2aWNlIFByb3ZpZGVyIDEwggEi MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDhMiozJ+EMp+4FG+eUP/l7zsgKOt3fTBLcxaf/ 07u2OI+ahxlaPHret67RZPH9ll8JTbO6Moo8KTN1D5hL2zFyU3vcvjZTpKQskztvzvS7ER4zJypM sgOD3bHc7QsJ3xOyMwPRmaYn6dYOB754e5eb0Gp9l3gQVhX6kVKk4GlqYXYfqzdQOiaTXaqobic 0HgFHwa8RLJ6vHyK+jqCi4K04HwYoJr6MdX5UISjKaNCYpOspYsPRYLFy4IEwr3x81Dev/KZnTsA 8pBx7nYdzxcLvwnNid6UUWeggBvtt2qUaVjEeEjTsqR0kl/jSyZhPcP4PElmd4piCJUBRpGxxe0N AgMBAAGjTTBLMAkGA1UdEwQCMAAwHQYDVR0OBBYEFMLzZR7uSN43ZyGJwcBaFE1V65jxMB8GA1Ud IwQYMBaAFDnhHg0/Y0mVNnCnPHTKTv+dS8J4MA0GCSqGSIb3DQEBCwUAA4IBAQB7CZ9A/8LvkwOv ZMxiLoRzT1Uf5tRD6dx6e+ttfkgfvpXjF8xEwfcIBTAd3uawS8hSCTR72YhHE8BTfQGsZJ5kv9Ka 8qHMCxO+MAeRRqEtxZVoFtBVYw9ppolKS/YAD/VFFyiEgCMdGn3hFAkcjSETTNNjpXhQPrGYlvBl 8dVDuI8mMZy0YX0gTn+jODS5QQZx8602GWr1O4OEnXEDU4t6msx0lItTtGv9d78dmINcaya7CV6S

kls/N7aboeBKiNwj11ZAvbXbMjuML8YYqmXyHyo1yGf6xd/Vz8mWRo6fUHC+kNnyL4X/zndBueV9 IE4CW3nmVZZuUD07qjN8x7Zb</ds:X509Certificate> </ds:X509Data> </ds:KeyInfo> </ds:Signature> <saml2:Conditions xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"> <saml2:AudienceRestriction><saml2:Audience>http://sp1.ejemplo.com/</saml2:Audience></saml2:AudienceRestriction> </saml2:Conditions> <saml2p:RequestedAuthnContext> <saml2:AuthnContextClassRef xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"/> </saml2p:RequestedAuthnContext> </saml2p:AuthnRequest>

El nodo padre será *saml2p:**AuthnRequest*** y tendrá como atributos:

- ***AssertionConsumerServiceURL (ACS URL)***: En esta URL se recibirá el Authentication Response Assertion indicando el resultado de la autenticación. Debe coincidir con el ingresado en el formulario de alta, registrado previamente en el *IDP*. No se soporta la modificación dinámica del ACS URL.
- ***Destination***: URL a la que se enviará el Authentication Request. Se debe seleccionar la adecuada según se use el binding POST o Redirect. Por ejemplo, para el binding POST se usará: https://eid.portal.gub.uy/idp/profile/SAML2/POST/SSO.
- ***ForceAuthn***: Forzar autenticación. Si se pone "False", el usuario no deberá volver a ingresar las credenciales si ya se había autenticado, por ejemplo, a través de otro *SP* que también delegó la autenticación en este sistema. Si se pone en "True", el usuario debe volver a ingresar las credenciales aunque que ya lo hubiese hecho y su sesión estuviese vigente. Esto se usa cuando se desea forzar que el usuario se autentique efectivamente al momento de solicitar la autenticación.
- ***ID***: Numero aleatorio que identifica la transacción. Se recomienda el uso de un número aleatorio cuyo primer carácter sea '_' , debido a que el identificador debe ser de tipo "NCName". Debe coincidir con el que se encontrará en el Response Assertion.
- ***Issue Instant***: Timestamp de emisión del assertion.
- ***Provider Name***: Entity ID del *SP*. Debe coincidir con el ingresado en el formulario de alta y previamente registrado en el *IDP*.
- ***Version***: 2.0 .

Luego los hijos del nodo raíz serán cuatro: *Issuer*, *Signature*, *Conditions* y *RequestedAuthContext*.

- El nodo *Issuer* deberá contener el Entiy ID del *SP*, al igual que en el atributo Provider Name del nodo Raíz.
- El nodo *Signature* se usa en el caso del binding POST y contendrá la firma del XML estándar realizada con el certificado del *SP* con que se dio de alta. Los métodos de digest, canonicalización y transformación son los especificados en el ejemplo. Se debe incluir el certificado de la Persona Jurídica firmante del Assertion en el campo *<ds:x509Certificate>* .

Próximamente, se cambiará el agoritmo de hashing a SHA-256

- En *Conditions* se puede indicar a quién va dirigida la respuesta, la que deberá coincidir con el Entity ID del *SP*.
- En *RequestedAuthnContext*, se puede indicar cómo se desea que sea la autenticación. Este tiene un hijo *"AuthnContextClassRef"*, que al dejarlo vacío se indica que cualquier método es válido (Cédula de Identidad o contraseña), si se desea que la autenticación sea con usuario y contraseña se debe poner *urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport;* si se desea que sea con cédula, debe incluir *urn:oasis:names:tc:SAML:2.0:ac:classes:SmartcardPKI* .

Authentication Response Assertion

Este es el mensaje que envía el IDP al SP para expresar que se realizó una autenticación exitosa o que el usuario ya estaba autenticado en el IDP.

A continuación, un ejemplo ilustrativo del Authentication Response Assertion:

*<?xml version="1.0" encoding="UTF-8"?> <saml2p:Response xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" Destination="https://test-eid.pge.red.uy/v1.1/sp-idp/saml/acs" ID="_9124f5ed7d644263c042edc32e30de1a" InResponseTo="_d06b8cde-3bea-43d2-be35-ce2e6ff7f3f4" IssueInstant="2016-08-19T19:55:39.865Z" Version="2.0" xmlns:xs="http://www.w3.org/2001/XMLSchema"> <saml2:Issuer xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion" Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">https://test-eid.portal.gub.uy/idp</saml2:Issuer> <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"> <ds:SignedInfo> <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/> <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/> <ds:Reference URI="#_9124f5ed7d644263c042edc32e30de1a"> <ds:Transforms> <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/> <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"> <ec:InclusiveNamespaces xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#" PrefixList="xs"/> </ds:Transform> </ds:Transforms> <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/> <ds:DigestValue>PyaSdwfEkVEqbVS1/O6qYXuHlU4=</ds:DigestValue> </ds:Reference> </ds:SignedInfo> <ds:SignatureValue>crACRqSPMCPuJxsBtuY2FCe/qGNucM4AdpxV+awYAEa+uSzF yLsjDFTZmzvbxFTYr+bxFQM1T0zzz5somJNmdt95NGhGMaD9rUHJf/3ARuSdfsb6BBy 0yL8gTnIO7H7VaSYmQrYcVdbL3P3hE6A0kO7LzVDMVgjVk78sgTP+rfuOrXwA6FPmVH 8YxC3q0z9hl2Vd2rmLe9bB+g3XQXPOzP8Xge8XYfDuZQHZWZ6OW0i30TNHGaZJVMLik FzIc7oQ4opHP0G0GyQu0hz5rYsndjW4c1RP4y9QEuvbEBQcdq8imECEtjP0QIeNpQQa UuSwuXhOKlhX4V5tD4o1OaSkw==</ds:SignatureValue> <ds:KeyInfo><ds:X509Data><ds:X509Certificate> MIIGkDCCBHigAwIBAgIUAS/jVUldwEKKjbmRAZS36pQT3MkwDQYJKoZIhvcNAQEL
BQAwWjEdMBsGA1UEAxMUQ29ycmVvIFVydWd1YXlvIC0gQ0ExLDAqBgNVBAoMI0Fk
bWluaXN0cmFjacOzbiB </ds:X509Certificate></ds:X509Data></ds:KeyInfo> </ds:Signature> <saml2p:Status><saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/></saml2p:Status> <saml2:Assertion xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion" ID="_89a21c16e2fbc949e0dc59b35e4b9356" IssueInstant="2016-08-19T19:55:39.865Z" Version="2.0" xmlns:xs="http://www.w3.org/2001/XMLSchema"> <saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">https://test-eid.portal.gub.uy/idp</saml2:Issuer> <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"> <ds:SignedInfo> <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/> <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/> <ds:Reference URI="#_89a21c16e2fbc949e0dc59b35e4b9356"> <ds:Transforms> <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/> <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"> <ec:InclusiveNamespaces xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#" PrefixList="xs"/> </ds:Transform> </ds:Transforms> <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/> <ds:DigestValue>tF3/GucTBtzvUMM1nWld2G76Jwk=</ds:DigestValue> </ds:Reference> </ds:SignedInfo> <ds:SignatureValue>C4zTEWkxISwbcArkyvMU+axdPOXYQwlyJQd3NIDoDFqEjy3SK7lUEGuuQ 3UMT7OQ+B+/wY7AVcISYljgD9l9McJChTUFnWv3uQq4L5XSstK+gTrH/L7THru6GJEL0GCvoihs4 k4cmi2Aztpxtv8kO6Tv0C/z71iWiNmZ8FtGN1gP0/Df3Af9gyTStJD/hObQu92DLQQFwTbdqCpIz VGXyuzhpNiyD6ibO03neBcsOkPCw8hbAWfR7dCVUN9EOvh2ojxwHIbQeJGZmgfjatPaqvNANJX7J bl2/Mkef+OcwstOiRgJqy6R2k6EsaS4LhCHnOBNoX31+UJr5psibE0kwQ== </ds:SignatureValue> <ds:KeyInfo><ds:X509Data><ds:X509Certificate> MIIGkDCCBHigAwIBAgIUAS/jVUldwEKKjbmRAZS36pQT3MkwDQYJKoZIhvcNAQEL
BQAwWjEdMBsGA1UEAxMUQ29ycmVvIFVydWd1YXlvIC0gQ0ExLDAqBgNVBAoMI0Fk
bWluaXN0cmFjacOzbiB </ds:X509Certificate></ds:X509Data></ds:KeyInfo> </ds:Signature> <saml2:Subject> <saml2:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient" NameQualifier="https://test-eid.portal.gub.uy/idp">_825e118486db99f3106a5bc0829f732a</saml2:NameID> <saml2:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer"> <saml2:SubjectConfirmationData Address="10.255.15.54" InResponseTo="_d06b8cde-3bea-43d2-be35-ce2e6ff7f3f4" NotOnOrAfter="2016-08-19T20:00:39.865Z" Recipient="https://test-eid.pge.red.uy/v1.1/sp-idp/saml/acs"/> </saml2:SubjectConfirmation> </saml2:Subject> <saml2:Conditions NotBefore="2016-08-19T19:55:39.865Z" NotOnOrAfter="2016-08-19T20:00:39.865Z"> <saml2:AudienceRestriction> <saml2:Audience>http://sp1.ejemplo.com/</saml2:Audience></saml2:AudienceRestriction> </saml2:Conditions> <saml2:AuthnStatement AuthnInstant="2016-08-19T19:55:39.662Z" SessionIndex="5f784076c0c17264942d28ce9a7b8d9ad8891ee9b0527f07b7db4f1e7f6044d0"> <saml2:SubjectLocality Address="10.255.15.54"/> <saml2:AuthnContext> <saml2:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</saml2:AuthnContextClassRef> </saml2:AuthnContext> </saml2:AuthnStatement> <saml2:AttributeStatement> <saml2:Attribute FriendlyName="Certificado" Name="Certificado" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">false</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="UID" Name="uid" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">UY-ci-12312314</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Primer Nombre" Name="PrimerNombre" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Rodrigo</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Pais Documento" Name="PaisDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">UY</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Primer Appelido" Name="PrimerApellido" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Perez</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Tipo Documento" Name="TipoDocumento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">68909</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Documento" Name="Documento" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">12312314</saml2:AttributeValue> </saml2:Attribute> <saml2:Attribute FriendlyName="Presencial" Name="Presencial" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"> <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">false</saml2:AttributeValue> </saml2:Attribute> </saml2:AttributeStatement> </saml2:Assertion> </saml2p:Response>*

El Authentication Response Assertion es el mensaje que envía el *IDP* al *SP* en respuesta a un pedido de autenticación. Este indica si el usuario se autenticó con éxito o no y, en caso afirmativo, indica sus datos.

Se forma con un nodo padre *saml2p:Response* con los siguientes atributos:

Destination: Contiene la URL hacia la cual es dirigido este assertion.

ID: Un identificador del assertion. Se recomienda el uso de un número aleatorio cuyo primer caracter sea '_' , ya que el identificador debe ser de tipo "NCName".

InResponseTo: ID del Authentication Request Assertion al que se está respondiendo.

IssueInstant: Fecha de emisión del Assertion.

Version: 2.0 .

El nodo raíz está compuesto por los nodos:

Issuer con el Entity ID del IDP (https://eid.portal.gub.uy/idp)

Signature con la firma xml de todo el Response.

Status con el resultado de la autenticación, si fue exitosa o no.

En caso de ser exitosa <saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
En caso contrario <saml2p:Status><saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Responder"><saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:AuthnFailed"/>
</saml2p:StatusCode></saml2p:Status> .

Assertion con los atributos del sujeto autenticado.

En el nodo Assertion tendremos la información sobre la autenticación. Tiene como atributos:

- Un *ID*, la *fecha de emisión* y la *versión* de forma análoga al nodo raíz.
- También de forma análoga al nodo raíz tiene la firma (*Signature*) con alcance restringido al nodo Assertion.

En el nodo Subject se indica el *ID del sujeto* dependiendo el formato elejido en el formulario podrá ser *urn:oasis:names:tc:SAML:2.0:nameid-format:transient* o *urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified* . Con *transient* se devuelve un identificador temporal, mientras que con *unspecified* se devuelve el UID en el formato real *PP-TD-NNNNNNNN* (País-Tipo-Número de documento). *El NameID debe de ser guardado por lo menos en la sesión para poder solicitar un deslogueo global*. De todas maneras, el UID es enviado como atributo en otro nodo. También se incluye información del ID del Authentication Request, hasta cuándo es válido el assertion y la URL a donde se lo está enviando. Estos últimos datos sirven para que el SP pueda cerrar el Authentication Response contra el Request originalmente realizado, tanto en tiempos de validez como contra el ID originalmente enviado.

- El nodo ***Conditions*** indica la fecha desde que es válido el assertion y la fecha hasta la que es valida el assertion en los atributos *NotBefore* y *NotOnOrAfter* respectivamente. En *Audience* se indica el Entity ID del SP al que va dirigido el assertion, y cualquier identificador diferente al propio debería ser descartado.

AuthnStatement tiene como atributo el *SessionIndex*, que representa el ID de la sesión. Es importante guardar este identificador ya que será el que usaremos para solicitar el deslogueo global. Otro atributo es la fecha de emisión del assertion.

En Subject Locality irá la IP desde donde es enviado el assertion (el IDP). Dentro del AuthnStatement irán varios nodos; particularmente importante es *AuthnContext*, ya que contiene el método por el cual se autenticó el usuario. Sus valores posibles son los mismos que se podían utilizar para pedir un método particular en el Authentication Response, es decir: *urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport* si se hizo login con usuario y password, y *urn:oasis:names:tc:SAML:2.0:ac:classes:SmartcardPKI* si se hizo login con Cédula de Identidad Digital.

Logout Request

El Logout Request Assertion es utilizado en dos momentos: cuando un SP solicita al IDP el deslogueo global y cuando el IDP debe solicitar al resto de los SP (los que no hicieron el request anterior) que realicen el deslogueo local.

El Logout Request se envía con el binding Redirect, por lo que el assertion se pone en un paramerto *SAMLRequest* y se complementa con los parametros *SigAlg (*donde irá el algoritmo de firma) y *Signature* (donde irá la firma en sí).

<?xml version="1.0" encoding="UTF-8"?> <saml2p:LogoutRequest Destination="https://eid.portal.gub.uy/idp/profile/SAML2/Redirect/SLO" ID="_98eb3adc-dca5-4be7-869d-43bcaa3b0615" IssueInstant="2016-02-04T16:26:13.257Z" Version="2.0" xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"> <saml2:Issuer xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">http://sp1.ejemplo.com/</saml2:Issuer> <saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified" xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">uy-ci-42502648</saml2:NameID> <saml2p:SessionIndex>5f784076c0c17264942d28ce9a7b8d9ad8891ee9b0527f07b7db4f1e7f6044d0</saml2p:SessionIndex> </saml2p:LogoutRequest>

El logout request assertion está compuesto por un nodo raíz "saml2p:LogoutRequest", que tendrá como atributos:

Destination, el cual contendrá el endpoint del *IDP* donde se solicitará el deslogueo (URL), en el caso donde el *SP* solicita el deslogueo global; en el caso donde el *IDP* está solicitando el deslogueo local, se pondrá en Destination el endpoint "Single Logout Location" que fue ingresado en el formulario y hacia donde se redirigirá el LogoutRequest.

ID: Número aleatorio que identifica la transacción.

Issue Instant: Timestamp de emisión del assertion.

Version: 2.0 .

El nodo raíz tendrá tres subnodos: Issuer, NameID y SessionIndex.

- En el nodo Issuer irá el Entity ID del *SP*. Debe coincidir con el ingresado en el formulario de alta y previamente registrado en el *IDP*.
- En NameID se debe incluir el mismo identificador recibido del sujeto autenticado que vino en el nodo Subject del Authentication Response Assertion.

En el nodo SessionIndex irá el mismo identificador de sesión obtenido en el Authentication Response Assertion. En el caso de los Request enviados por el IDP (para deslogueo local), no será incluido.

Logout Response Assertion

El logout response es enviado como respuesta a un Logout Request. Al igual que en los LogoutRequest, se usará en dos casos: cuando los SP deben responderle al IDP sobre el deslogueo local exitoso y cuando el IDP le debe avisar el resultado de este al SP que hizo el request de deslogueo global original.

<?xml version="1.0" encoding="UTF-8"?> <saml2p:LogoutResponse xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" Destination="https://eid.portal.gub.uy/sp-idp/saml/sls" ID="_4c03f43d74b4948622322e8fcbc84fc0" InResponseTo="_98eb3adc-dca5-4be7-869d-43bcaa3b0615" IssueInstant="2016-02-04T16:24:01.998Z" Version="2.0"> <saml2:Issuer xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion" Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">idp.portal.gub.uy</saml2:Issuer> <saml2p:Status> <saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/> </saml2p:Status> </saml2p:LogoutResponse>

El nodo raíz de un Logout Response tendrá los atributos comunes a los assertions de respuesta: Destination, ID, InResponseTo, IssueInstant y Version.

- ***Destination***: Contiene la URL hacia donde es dirigido este assertion. En el caso de que estemos respondiendo a un pedido de deslogueo local por parte del *IDP* será el endpoint de Logout del *IDP*. En caso de que sea el *IDP* que nos estará respondiendo a nuestro pedido de autenticación global, será la URL ingresada en el formulario *Single Logout Response Location*
- ***ID***: Un identificador del assertion.
- ***InResponseTo***: ID del Logout Request al que se está respondiendo.
- ***IssueInstant***: Fecha de emisión del Assertion.
- ***Version***: 2.0 .

El nodo raíz tendrá dos subnodos:

Issuer: De ser el logout response del SP al IDP, en este campo deberá ir el Entity ID del SP. Debe coincidir con el ingresado en el formulario de alta y previamente registrado en el IDP. De lo contrario, será el Entity Id del IDP.

Status: Indicará el resultado del deslogueo. Habra dos codigos: *Success* en caso exitoso y *PartialLogout* que se envía al responderle al SP que inició el pedido de deslogueo global bajo la condición de que no todos los SP comunicaron el resultado del deslogueo local de forma exitosa (se asume que el SP que inicia el deslogueo global se desloguea localmente de forma exitosa).

Solicitud de cliente SAML 2.0 TEST de ID Uruguay:acceder a la solicitud Integracion ID Uruguay SAML 2.0

ID Uruguay: integración con OpenID Connect

Objetivo

El objetivo de este documento es brindar una descripción detallada de cómo funciona el protocolo OpenID Connect 1.0 en ID Uruguay y cómo es posible operar como un cliente de este para obtener información verificada de la ciudadanía de manera segura, tanto para la aplicación cliente como para los usuarios.

Terminología

La terminología utilizada en este documento responde a la detallada por las especificaciones de OpenID Connect 1.0 y OAuth 2.0. No obstante, se brindarán definiciones para algunos términos fundamentales como "roles", "claims", "scopes" y "tokens".

¿Qué es ID Uruguay?

Es la plataforma de autenticación implementada por Agesic para centralizar cuentas de usuarios y facilitar el acceso web a los servicios digitales del Estado. Esto quiere decir que, una vez registrado en ID Uruguay, un usuario podrá ingresar a los servicios vinculados a la cuenta sin necesidad de nuevos registros ni contraseñas adicionales.

¿Qué es OpenID Connect 1.0?

Es un protocolo de identidad simple y de estándar abierto creado sobre el protocolo OAuth 2.0, el cual permite a aplicaciones Cliente *(Relaying Party)* verificar la identidad de un usuario basado en la autenticación realizada por este en un Servidor de Autorización *(OpenID Provider)*, así como también obtener información personal del usuario mediante el uso de una *API REST*.

¿Por qué OpenID Connect 1.0 en ID Uruguay?

OpenID Connect 1.0 es un protocolo que se encuentra implementado en múltiples lenguajes de programación y está siendo utilizado por la mayoría de los sitios y aplicaciones que requieren el manejo de un usuario, existiendo muchos proveedores (Google, Facebook, Linkedin, etc) y aún muchos más Clientes.

Su principal funcionalidad es permitir a usuarios de un sitio (Cliente) iniciar sesión o registrarse autenticándose en otro sitio (OpenID Provider) que ya contiene sus datos.

Entre sus principales ventajas se encuentran:

Acelera el proceso de registro

La mayoría de los portales del Estado (y otros sitios) solicitan a sus usuarios que completen un formulario de registro con información que, por lo general, comprende el mismo conjunto de datos (primer nombre, primer apellido, correo electrónico, etc.). Con OpenID Connect, los usuarios pueden proveer esta información con un solo clic. De este modo, el proceso de registro es más simple, rápido y seguro.

Reduce la frustración de administrar diferentes contraseñas

La mayoría de los usuarios que utilizan servicios web deben recordar diferentes nombres de usuario y contraseñas para iniciar sesión en cada uno de ellos. Recordar estas combinaciones puede ser una tarea muy tediosa, pero utilizar las mismas credenciales implica un problema de seguridad importante. Con OpenID Connect, es posible iniciar sesión o registrarte en todos estos sitios manteniendo solo una cuenta (la de ID Uruguay).

Mejora el control sobre la identidad electrónica

Cada vez que un usuario inicia sesión en un aplicación externa utilizando OpenID Connect, deberá dar su consentimiento explícito de qué datos quiere compartir con la aplicación. De este modo, su cuenta ID Uruguay servirá como una identidad centralizada que podrá utilizarse de manera controlada en muchos sitios de internet.

Minimiza los riesgos de seguridad asociados a las contraseñas

Muchos usuarios utilizan la misma contraseña en múltiples sitios. De este modo, si alguno de estos fuera atacado y las contraseñas quedaran comprometidas, un hacker podría obtener acceso a todos los sitios que comparten esta credencial. Con OpenID Connect, si ocurriera una situación que comprometa las credenciales, basta con restablecer la contraseña en un lugar, ID Uruguay.

Roles del flujo de autenticación OpenID Connect 1.0

Los roles que participan en un flujo de autenticación OpenID Connect 1.0 son los siguientes:

- **End-User:** Participante humano capaz de autorizar el acceso a un recurso protegido validando su identidad.
- **Relaying Party (RP):** Aplicación Cliente que solicita la autenticación de un End-User a un OpenID Provider con el fin de poder acceder a recursos protegidos en nombre del usuario autenticado. En este caso, será la aplicación web/mobile que quiere integrarse con ID Uruguay para obtener información del usuario (claims).
- **OpenID Provider (OP):** Servidor de autenticación capaz de autenticar usuarios y proveer información sobre estos y el proceso de autenticación a un Relying Party. En este caso, será ID Uruguay.

Descripción general

En términos generales, el protocolo OpenID Connect 1.0 en ID Uruguay se puede describir a través de los pasos listados a continuación:

1. Un Cliente registrado (RP) envía un pedido de autenticación a ID Uruguay (OP).

2. El OP autentica al usuario y obtiene su autorización para compartir ciertos datos (claims) con el RP.

3. El OP responde con un ID Token y un Access Token al RP.

4. El RP utiliza el Access Token para solicitar información sobre el usuario al UserInfo Endpoint.

5. El UserInfo Endpoint retorna un listado de claims del usuario.

Finalmente, la aplicación Cliente (RP) podrá crear una sesión para el usuario autenticado o registrarlo en su base de datos con los claims obtenidos.

* **NOTA:** la autenticación en OpenID Connect puede llevarse a cabo de tres formas: *Authorization Code Flow, Implicit Flow o Hybrid Flow*. El mecanismo seleccionado determina como el *ID Token* y *Access Token* son retornados al Cliente y esto puede hacer que ciertos pasos sean modificados y/o nuevos pasos añadidos. En particular en ID Uruguay se implementa el *Authorization Code Flow*, que será descrito en detalle más adelante.

Tokens

En esta sección se presentan los diferentes *tokens* que participan en el proceso de autenticación y autorización de OpenID Connect 1.0 y, por lo tanto, en la implementación de ID Uruguay.

ID token

Un *ID token* es un JSON Web Token (JWT) que contiene información sobre el proceso de autenticación del *End-User* en el servidor de autenticación (OP).

Access token

Los *Access tokens* son credenciales emitidas por el servidor de autenticación (OP) para un cliente (RP) y tienen como fin permitirles el acceso a recursos protegidos. Un *Access Token* es un *string opaco* que representa el acceso a ciertos datos y puede ser utilizado por un tiempo limitado.

Refresh token

Los *Refresh tokens* son credenciales emitidas por el servidor de autenticación (OP) para un cliente (RP) y tienen como fin la obtención de nuevos *Access tokens* cuando estos expiran o se vuelven inválidos.

Claims y Scopes

Los *JWT* contienen *claims*, campos de información (tales como nombre o e-mail) sobre una entidad (típicamente un usuario) y metada adicional.

OpenID Connect 1.0 define un conjunto standard de *claims*, que incluyen nombre, correo electrónico y género, entre otros. Y los agrupa en *scopes*, que permiten a un RP definir qué tipo de información desea obtener sobre un usuario.

Por su parte, ID Uruguay toma como base los *claims* y *scopes* de OpenID Connect 1.0 y define los siguientes, que podrán ser solicitados en un *Authentication Request*:

Claims y Scopes

Nombre	Claims	Descripción
personal_info	nombre_completo, primer_nombre, segundo_nombre, primer_apellido, segundo_apellido, uid, rid	Nombres y apellidos del usuario, identificador y el nivel de registro de identidad digital. Este último puede ser alguno de los siguientes valores: [0,1,2,3] correspondiendo a los niveles Muy Bajo, Bajo, Medio y Alto, respectivamente.
profile	name, given_name, family_name	Nombre completo, nombre(s) y apellido(s) respectivamente.
document	pais_documento, tipo_documento, numero_documento	Información sobre el documento del usuario.
email	email, email_verified	Correo electrónico y si el mismo está verificado.
auth_info	rid, nid, ae	Datos de registro y autenticación del ciudadano en formato URN correspondientes a la Política de Identificación Digital.

De este modo, un RP podrá obtener los datos incluidos en los *scopes* que solicite, previa autorización del *End-User* (propietario de dichos datos).

* **NOTA:** el scope "profile" presenta información de nombres y apellidos del ciudadano de manera distinta a "personal_info". Este scope puede ser de ayuda para usar integraciones standard. Si se necesita la información del ciudadano en forma más modular, se recomienda usar el scope "personal_info".

ACR - Authentication Context Class reference

Authentication Context Class corresponde a un conjunto de métodos o procedimientos de autenticación que se consideran equivalentes entre sí en un contexto particular.

Los valores acr definidos en ID Uruguay son: "urn:iduruguay:nid:0, urn:iduruguay:nid:1, urn:iduruguay:nid:2 y urn:iduruguay:nid:3". Estos valores se corresponden con los *niveles de seguridad en la identidad digital* de la Política de Identificación Digital. Los nid son una correspondencia entre el *RID* del usuario en gub.uy (procedimiento de registro de identificación digital) y el *AE* utilizado (proceso de autenticación electrónica).

La solicitud de acr_values se realiza en la *Authentication request* y el acr satisfecho por el OP al autenticar al usuario es retornado en el *ID Token*. De no haber sido posible cumplir con lo solicitado, igualmente se retornará el acr satisfecho.

Si el *(Relaying Party)* envía un acr que no se corresponde a ninguno de los mencionados anteriormente, el OP responderá a la authentication request con error "invalid_request" y error_description "The request is otherwise malformed".

AMR - Authentication Methods References

Authentication Methods References es un JSON array of strings que corresponden a identificadores de métodos de autenticación usados en la autenticación.

Estos valores son retornados al *(Relaying Party)* en el *ID Token*.

Los valores amr definidos en ID Uruguay son: "urn:iduruguay:am:password, urn:iduruguay:am:totp, urn:iduruguay:am:ci, urn:iduruguay:am:idp:ae:0, urn:iduruguay:am:idp:ae:1, urn:iduruguay:am:idp:ae:2, urn:iduruguay:am:idp:ae:3".

Los valores con formato "urn:iduruguay:am:idp:ae:X" indican que el usuario se autenticó utilizando un Proveedor de Identidad, y los métodos que utilizó se corresponden con el *AE* (proceso de autenticación electrónica) X.

Integración OpenID Connect con ID Uruguay

El objetivo de esta sección es describir el procedimiento para registrarse y poder operar como *Relaying Party* en el proceso de autenticación/autorización OpenID Connect 1.0 de ID Uruguay

Registro de Sistema Informático para integrarse a ID Uruguay utilizando Open ID Connect

Para solicitar la integración, debe ingresar la solicitud en un trámite disponible acá: https://ioti.bpmgob.agesic.red.uy/simple/tramites/disponibles
Seleccionar la opción " Integracion ID Uruguay OpenID Connect"
Para esto es necesario:

- Contar con una identificación digital de ID Uruguay al menos con nivel intermedio.
- Ingresar al trámite a través de la Red Uy (en otro caso por favor contactarnos).

Es necesario realizar el trámite para integrarse primero a nivel de testing y luego para producción ya que son conexiones y credenciales diferentes.
Para el caso de producción, el trámite debe ser realizado por el funcionario público responsable técnicamente del organismo interesado.
Por cualquier duda, por favor contactarse a soporte@agesic.gub.uy con copia a identificacion.electronica@agesic.gub.uy .

Autenticación utilizando Authorization Code Flow

El *Authorization Code Flow* es uno de los tres posibles flujos de autenticación provistos por el protocolo OpenID Connect 1.0. En este, un RP redirige al *End-User* al *Authorization Endpoint* del OP, el cual lleva a cabo su autenticación y autorización. Si el resultado es exitoso, el OP retorna al RP un *Authorization Code*, que podrá ser utilizado (dentro de un período de tiempo) para obtener un *ID Token* y *Access Token* desde el *Token Endpoint*. Finalmente, el *Access Token* obtenido puede ser utilizado para conseguir *claims* sobre el usuario en el *Userinfo Endpoint*. El diagrama a continuación ilustra el flujo descrito.

* **NOTA:** este flujo de autenticación tiene como beneficio que ningún token se encuentra expuesto al User Agent y, de este modo, a posibles aplicaciones maliciosas que lo controlen. Por ello, es apropiado para Clientes que puedan mantener de manera segura una clave secreta, típicamente aplicaciones con un Backend de datos.

Authorization Endpoint (/oidc/v1/authorize)

El *Authorization Endpoint* lleva a cabo la autenticación y autorización del *End-User*. Para invocarlo se debe enviar un *Authentication request*, que será respondido por un *Authentication response*. Ambos pedidos son detallados a continuación.

Authentication request

Un *Authentication request* es un pedido HTTP con ciertos parámetros que sirve para solicitar la autenticación de un *End-User* en ID Uruguay.

Este pedido puede llevarse a cabo empleando los métodos HTTP GET o HTTP POST definidos en el RFC 2616. Si se utiliza el método HTTP GET, los parámetros deberán ser serializados empleando URI Query String Serialization. En caso de utilizar el método HTTP POST, los parámetros deberán ser serializados utilizando Form Serialization.

A continuación, se muestra una tabla con los parámetros aceptados y una breve descripción de cada uno:

Authentication request

Parámetro	Tipo	Descripción
scope	Requerido	Siempre debe incluirse "openid". Adicionalmente, se pueden incluir los scopes descritos en la sección "Claims y Scopes" separados por espacios. Ej: "openid personal_info email".
response_type	Requerido	Valor que determina el tipo de flujo de autenticación a utilizar. En caso del *Authorization Code Flow*, es valor es "code".
client_id	Requerido	Identificador del cliente provisto al momento del registro.
redirect_uri	Requerido	URI a donde debe ser enviada la respuesta. Esta debe ser una de las registradas al momento de darse de alta como cliente.
state	Recomendado	Valor opaco para mantener el estado entre el pedido y la respuesta. Será retornado al cliente junto con el código de autorización o error

Parámetro	Tipo	Descripción
nonce	Opcional	String opaco utilizado para asociar la sesión de un Cliente con un *ID Token* y mitigar *replay attacks*.
prompt	Opcional	Lista de valores de cadena ASCII delimitados por un espacio, sensibles a minúsculas y mayúsculas, que especifica si el servidor de autorización solicita al usuario final la reautenticación y consentimiento. Los valores definidos son: `none, login y consent.`
acr_values	Opcional	Lista de strings sensibles a minúsculas y mayúsculas, separados por espacios y en orden de preferencia, correspondientes a los nombrados en la sección "acr - Authentication Context Class Reference".

Ejemplo de Authentication request con HTTP GET

El RP retornará un HTTP 302 Redirect.

HTTP/1.1 302 Found Location: https://auth-testing.iduruguay.gub.uy/oidc/v1/authorize? response_type=code &scope=openid%20personal%20email &client_id=ID_CLIENTE &state=STRING_RANDOM &redirect_uri=https%3A%2F%2Fclient.org%2F

El navegador realiza el pedido:

GET /oidc/v1/authorize? response_type=code &scope=openid%20personal%20email &client_id=ID_CLIENTE &state=STRING_RANDOM &redirect_uri=https%3A%2F%2Fclient.org%2F Host: https://auth-testing.iduruguay.gub.uy

Ejemplo de Authentication request con HTTP POST

POST /oidc/v1/authorize HTTP/1.1 Host: auth-testing.iduruguay.gub.uy Content-Type: application/x-www-form-urlencoded
response_type=code&scope=openid%20personal%20email&client_id=ID_CLIENTE&state=STRING_RANDOM&redirect_uri=https%3A%2F%2Fclient.org%2F

Authentication Response

El *Authentication Response* es el mensaje retornado por el *Authorization Endpoint* del OP en respuesta a un *Authentication request* enviado por el RP.

La respuesta incluye los parámetros "code" y "state", codificados con el formato *"application/x-www-form-urlencoded"* y añadidos a la *redirect_uri* especificada al enviar el *Authentication request.* La tabla a continuación presenta una breve descripción de los parámetros mencionados:

Authentication Response

Parámetro	Tipo	Descripción
code	Requerido	Código de autorización generado por el OP. Puede ser utilizado una única vez para obtener un *ID Token* y *Access Token*. Expira en 10 minutos.
state	Requerido si fue envíado	El valor exacto recibido del RP en el parámetro `state` del *Authentication Request*.

Ejemplo de respuesta exitosa

HTTP/1.1 302 Found Location: https://client.org/? code=SplxlOBeZQQYbYS6WxSbIA &state=STRING_RANDOM

Si el *request* falla debido a que el parámetro *redirect_uri* es vacío, inválido o no coincide con ninguna de las URI de redirección configuradas al momento del registro o si el parámetro *client_id* es vacío o inválido, el OP mostrará al *End-User* una pantalla de error y no redireccionará el *User-Agent* a la URI inválida.

Por otra parte, si el *End-User* rechaza el request o si la autenticación falla por razones diferentes a las antes mencionadas, el OP informa al RP añadiendo a la URI especificada por el parámetro *redirect_uri*, los siguientes parámetros codificados con el formato *"application/x-www-form-urlencoded"*:

Ejemplo de respuesta exitosa

Parámetro	Tipo	Descripción
error	Requerido	Un código de error de los descritos en OpenID Connect 1.0 u OAuth 2.0
error_description	Opcional	Descripción del error que provee información para ayudar a los desarrolladores a entender el error ocurrido.
state	Requerido si fue envíado	El valor exacto recibido del RP en el parámetro "state" del *Authentication Request*.

Ejemplo de respuesta con error

HTTP/1.1 302 Found Location: https://client.org/? error=invalid_request &error_description= Unsupported%20response_type%20value &state=STRING_RANDOM

Token Endpoint (/oidc/v1/token)

Para obtener un *ID Token*, un *Access Token* y opcionalmente un *Refresh Token*, el RP envía un *Token Request* al *Token Endpoint* del OP, quien responderá con un *Token Response*.

Token Request

Un RP realiza un *Token Request* presentando su código de autorización (`code`) ante el *Token Endpoint* del OP. Este *request* debe implementarse utilizando el método HTTP POST, y debe contener los siguientes parámetros serializados como Form Serialization:

Token Request

Parámetro	Tipo	Descripción
grant_type	Requerido	Tipo de credenciales a presentar. Debe ser "authorization_code".
code	Requerido	Código de autorización emitido por el OP, previamente tramitado en el *Authentication Endpoint.*
redirect_uri	Requerido	URI a donde debe ser redirigido el *User Agent* con la respuesta (*Token Response*). Debe ser una de las URIs configuradas al momento del registro del RP.

Adicionalmente a estos parámetros, el _Token Request_ debe contener las credenciales de autenticación provistas al momento del registro (*client_id y client_secret*) siguiendo el esquema de autenticación HTTP Basic Auth.

Ejemplo de Token Request HTTP POST

Si `client_id = 123456789` y `client_secret = 0Pg8RabLluvuoG3`, un ejemplo de *Token Request* es:

POST /token HTTP/1.1 Host: https://auth-testing.iduruguay.gub.uy Content-Type: application/x-www-form-urlencoded Authorization: Basic MTIzNDU2Nzg5OjBQZzhSYWJMbHV2dW9HMw==
grant_type=authorization_code&code=SplxlOBeZQQYbYS6WxSbIA &redirect_uri=https%3A%2F%2Fclient.org%2F

En caso de que se quiera obtener un nuevo Access Token (Refresh), se puede utilizar el Token Endpoint enviando un Token Request que con los parámetros:

Ejemplo de Token Request HTTP POST

Parámetro	Tipo	Descripción
grant_type	Requerido	Tipo de credenciales a presentar. Debe ser `refresh_token`
refresh_token	Requerido	`refresh_token` obtenido en el *ID Token* anterior.

Token Response

Luego de recibido y validado un *Token Request* del RP, el OP procede a responder con un *Token Response* que incluye los siguientes parámetros codificados como `application/json`:

Token Response

Parámetro	Tipo	Descripción.
access_token	Requerido	*Access Token* emitido por el OP.
token_type	Requerido	Tipo de token. Será siempre "Bearer".
id_token	Requerido	*ID Token* asociado a la sesión de autenticación.
expires_in	Recomendado	Tiempo de vida del *Access Token* en segundos. Valor por defecto 60 minutos.
refresh_token	Opcional	*Refresh Token* que puede ser utilizado para obtener nuevos *Access Tokens.*

Ejemplo de Token Response exitoso

HTTP/1.1 200 OK Content-Type: application/json Cache-Control: no-store Pragma: no-cache { "access_token": "bf6ab0e8fcef4ec7be5cfbfecb520c7f", "token_type": "bearer", "refresh_token": "6859d02ddb794e66b71321b587046344", "expires_in": 3600, "id_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzc yI6ICJodHRwOi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4Mjg5 NzYxMDAxIiwKICJhdWQiOiAiczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZ fV3pBMk1qIiwKICJleHAiOiAxMzExMjgxOTcwLAogImlhdCI6IDEzMTEyODA5Nz AKfQ.ggW8hZ1EuVLuxNuuIJKX_V8a_OMXzR0EHR9R6jgdqrOOF4daGU96Sr_P6q Jp6IcmD3HP99Obi1PRs-cwh3LO-p146waJ8lhehcwL7F09JdijmBqkvPeB2T9CJ NqeGpe-gccMg4vfKjkM8FcGvnzZUN4_KSP0aAp1tOJ1zZwgjxqGByKHiOtX7Tpd QyHE5lcMiKPXfEIQILVq0pc_E2DzL7emopWoaoZTF_m0_N0YzFC6g6EJbOEoRoS K5hoDalrcvRYLSrQAZZKflyuVCyixEoV9GfNQC3_osjzw2PAithfubEEBLuVVk4 XUVrWOLrLl0nx7RkKU8NXNHq-rvKMzqg" }

* **NOTA:** para evitar que el *User-Agent* almacene en caché información sensible. El *Token Response* incluye los Headers HTTP "Cache-Control: no-store" y "Pragma: no-cache".

Si el *Token Request* es inválido o no pudo ser autorizado, el OP crea una respuesta con los siguientes parámetros codificados como "application/json" y un código de error "HTTP 400 Bad Request":

Ejemplo de Token Response exitoso

Parámetro	Tipo	Descripción.
error	Requerido	Un código de error de los descritos en OAuth 2.0.
error_description	Opcional	Descripción del error que provee información para ayudar a los desarrolladores a entender el error ocurrido.

Userinfo Endpoint (/oidc/v1/userinfo)

El *UserInfo Endpoint* es un *endpoint* protegido que retorna *claims* sobre el *End-User* autenticado. Para obtener estos *claims*, el RP envía un *UserInfo Request* acompañado por un *Access Token* obtenido a través del flujo de autenticación OpenID Connect. Los *claims* se retornan como clave-valor en la respuesta con formato JSON.

UserInfo Request

Un *UserInfo Request* puede ser implementado utilizando los métodos HTTP GET y HTTP POST, y en cualquier caso deberá incluir el header HTTP `Authorization` (definido por HTTP 1.1 siguiendo el esquema `Bearer` OAuth 2.0 Bearer Token Usage para incluir el *Access Token* previamente obtenido.

Ejemplo de UserInfo Request HTTP GET

Si `SlAV32hkKG` es el *Access Token* obtenido, un UserInfo Request podría ser:

GET /userinfo HTTP/1.1 Host: https://auth-testing.iduruguay.gub.uy Authorization: Bearer SlAV32hkKG

UserInfo Response

El *UserInfo Response* se retorna como respuesta a un *UserInfo Request* y devuelve los *claims* solicitados como atributos de un JSON. Por razones de privacidad, el OP puede elegir no retornar alguno de los *claims* solicitados (si, por ejemplo, no fue acordado al momento del registro).

El *UserInfo Response* siempre incluirá el campo "sub" (subject). Este, debe ser exactamente igual al campo `sub` del ID Token recibido. Si no coinciden, la respuesta debe considerarse inválida.

Ejemplo de UserInfo Response exitoso.

HTTP/1.1 200 OK Content-Type: application/json { "sub": "248289761001", "email": "juan@email.com", "primer_nombre": "Juan", "segundo_nombre": "José", "primer_apellido": "Perez", "segundo_apellido": "Martinez" }

Si ocurre algún tipo de error, el *UserInfo Endpoint* retorna un *Error Response* tal y como lo especifica OAuth 2.0 Bearer Token Usage

Ejemplo de UserInfo Response con error.

HTTP/1.1 401 Unauthorized WWW-Authenticate: error="invalid_token", error_description="The Access Token expired"

JWKS Endpoint (/oidc/v1/jwks)

El estándar JSON Web Key (JWK) define una manera consistente de poder representar una clave criptográfica en formato JSON. *JSON Web Key Set (JWKS)* es, justamente, un set o conjunto de JWKs.

JWKS Endpoint expone las claves y algoritmos que el OP usa y puede ser útil a los RP para poder verificar la autenticidad de los tokens emitidos.

Este endpoint es parte del estándar Open ID Connect Discovery.

JWKS Request

En ID Uruguay, se puede acceder al *JWKS Endpoint* empleando el método HTTP GET

Ejemplo de JWKS Request HTTP GET

GET /oidc/v1/jwks HTTP/1.1 Host: https://auth-testing.iduruguay.gub.uy Content-Type: application/json

JWKS Response

El *JWKS Response* es una lista codificada en formato "application/json" con las JWKs que el servidor implementa.

Cada JWK tiene un identificador único ("kid"). Cada JWT contiene este identificador en el parámetro "header" para indicar que JWK usa.

La lista de parámetros completos son:

JWKS Response

Parámetro	Descripción
kty	Identifica la familia del algoritmo criptográfico utilizado.
alg	Identifica el algoritmo utilizado. ("RS256", "HS256").

Parámetro	Descripción
use	Identifica el uso previsto de la clave pública. Indica si se usa para cifrar datos ("enc") o para verificar la firma ("sig")
kid	Identificador único.
n	El módulo de la clave (2048 bit). Codificado en Base64.
e	El exponente de la clave (2048 bit). Codificado en Base64.

Ejemplo de JWKS Response exitoso.

HTTP/1.1 200 OK Content-Type: application/json { "keys": [{ "kty": "RSA", "alg": "RS256", "use": "sig", "kid": "84361b05da05b6f656f7e13e575f8431", "n": "wYtA6llGGvE28lbJwXJ4Ks03IS9Y4tAJv_fyBollr7XTnNujGJ6N4easerObmD8iVI5TCQHCuh8z9HahtDVW_Ci8PHeokGgGcBDBLi0t4J9It_6mhVzWoGNN3HlzzJTVFIOnTykCzVkhEEoEPKL1SsDdv3NOffgLTo("e": "AQAB" }] }

Logout Endpoint (/OIDC/V1/LOGOUT)

Cuando un proceso de cierre de sesión es iniciado en el SP, este podría querer solicitar que también se produzca en el OP.

Este endpoint forma parte del estandard OpenID Connect Session Management, el cual propone cierto mecanismo para llevar a cabo el cierre de sesión del ciudadano en el OP.

Este endpoint solo puede ser invocado mediante el método `HTTP GET`.

Adicionalmente, puede ser descubierto desde la sección OpenID Configuration Endpoint como "end_session_endpoint".

Los parámetros necesarios son:

Logout Endpoint (/OIDC/V1/LOGOUT)

Parámetro	Tipo	Descripción.
id_token_hint	Requerido	Corresponde al "id_token" obtenido en el mecanismo de inicio de sesión del SP. Identifica al ciudadano y cliente en cuestión y valida la integridad del SP por el hecho de su posesión, ya que fue intercambiado de forma segura.
post_logout_redirect_uri	Opcional	URL a la cual será redireccionado el SP luego que el logout en el OP finalice exitosamente. Esta URL debe existir en la configuración que mantiene el OP del SP; si no existe o no es exactamente igual, será redireccionado al inicio del OP.
state	Opcional	Valor opaco para mantener el estado entre el pedido y la respuesta. Será retornado como parámetro en la "post_logout_redirect_uri" enviada.

CASO RECOMENDADO OpenID Configuration Endpoint V2:

Documento JSON fue actualizado para corregir desviaciones del protocolo OpenID Connect detactadas en la versión anterior que se expone más abajo, disponible en la ruta formada por la concatención del Issuer (https://auth-testing.iduruguay.gub.uy/oidc/v2/.well-known/openid-configuration) al string "/.well-known/openid-configuration".

https://auth-testing.iduruguay.gub.uy/oidc/v2/jwks

En este caso fue eliminado el campo x5c del jwks para solucionar un problema de compatibilidad con algunas soluciones, y se ajustó el wellknown para que coincidiera con el del token JWT en caso de presentar problemas con este campo “x5c” que se encuentra en el Endpoint V1 entonces se puede utilizar esta versión del Endpoint.

OpenID Configuration Endpoint Legacy

Hay un documento JSON disponible en la ruta formada por la concatención del Issuer (https: //auth-testing.iduruguay.gub.uy/oidc/v 1 al string "/.well-known/openid-configuration".

Este documento describe la configuración del proveedor OpenID ID Uruguay.

Este endpoint es parte del estándar OpenID Connect Discovery 1.0.

OpenID Provider Configuration Request

El documento de configuración debe ser consultado utilizando el método HTTP GET a la ruta especificada anteriormente.

OpenID Provider Configuration Response

La respuesta es un conjunto de Claims acerca de la configuración del proveedor OpenID ID Urguay, incluyendo todos los endpoint necesarios y la ubicación de la clave pública.

Claims que retornan varios valores, son representados como arreglos JSON.

Ejemplo de OpenID Provider Configuration Response exitoso.

HTTP/1.1 200 OK Content-Type: application/json { "issuer": "https://auth-testing.iduruguay.gub.uy/oidc", "authorization_endpoint": "https://auth-testing.iduruguay.gub.uy/oidc/authorize", "token_endpoint": "https://auth-testing.iduruguay.gub.uy/oidc/token", "userinfo_endpoint": "https://auth-testing.iduruguay.gub.uy/oidc/userinfo", "end_session_endpoint" : "https://auth-testing.iduruguay.gub.uy/oidc/logout", "id_token_signing_alg_values_supported": ["HS256", "RS256"], "jwks_uri": "https://auth-testing.iduruguay.gub.uy/oidc/jwks", "scopes_supported": ["openid", "personal_info", "profile", "email", "document", "auth_info"], "response_types_supported": ["code"], acr_values_supported: ["urn:iduruguay:nid:0", "urn:iduruguay:nid:1", "urn:iduruguay:nid:2", "urn:iduruguay:nid:3"], subject_types_supported: ["public"], "claims_parameter_supported": true, "claims_supported": ["nombre_completo", "primer_nombre", "segundo_nombre", "primer_apellido", "segundo_apellido", "uid", "name", "given_name", "family_name", "pais_documento", "tipo_documento", "numero_documento", "email", "email_verified", "rid", "nid", "ae"], "service_documentation": "https://centroderecursos.agesic.gub.uy/" }

Solicitud de cliente OpenID Connect TEST de ID Uruguay: acceder a la solicitud Integracion ID Uruguay OpenID Connect

Descripción técnica de ID Uruguay

Los servicios digitales que ofrece el Estado uruguayo requieren diferentes niveles de seguridad y garantías de identidad, de acuerdo a la criticidad de la información que manejan o a las acciones que pueden sucederse una vez consumidos por la ciudadanía.

ID Uruguay tiene un esquema llamado Single Sign On, que tiene como objetivo la formación de una federación de identidades de los diferentes organismos en torno a un único proveedor de identidades (Identity Provider - IdP). El objetivo es que las personas cuenten con un único conjunto de credenciales y un único punto para autenticarse. Una vez que la persona se crea un usuario, automáticamente puede ir a cualquiera de los organismos que lo integran sin necesidad de volver a iniciar sesión o proveer credenciales adicionales, haciendo que el esquema de autenticación sea totalmente centrado en la ciudadanía.

Objetivo

El propósito de este artículo es describir las características de Usuario.gub.uy para dar un panorama funcional de la aplicación.

Se detallan qué necesidades de los organismos cubre la plataforma una vez realizada la integración, así como también los requerimientos que se deben cumplir para concretarla. Adicionalmente, se realiza una introducción a los distintos procedimientos que deben respetar los organismos que deseen integrarse con ID Uruguay.

Descripción general

ID Uruguay es un producto que brinda servicios de autenticación a todos aquellos Proveedores de Servicio (SP, por sus siglas en inglés) que tengan el interés de unirse a la federación de identidades mediante una integración técnica.

Muchas veces el concepto de "autenticación" está relacionado con la "autorización", debido a que para la mayoría de los casos es transparente para las personas. Dado que este producto se limita exclusivamente a la autenticación de personas, es necesario diferenciar los siguientes conceptos:

- **Registro:** es el proceso en el cual una persona se registra en el sistema. Condición necesaria para la autenticación con usuario y contraseña.
- **Autenticación:** es un procedimiento por el cual se verifica que una persona es quien dice ser.
- **Autorización:** es el paso siguiente a la autenticación y consiste en asegurar que cierta funcionalidad es solamente accesible a aquellos personas que tienen los permisos correspondientes.

Por lo tanto, ID Uruguay finaliza su rol una vez que autentica a la persona y le comunica al SP que esta es quien dice ser mediante el protocolo seguro de integración utilizado. Como consecuencia, el SP otorga los accesos que considere pertinentes para cada caso en particular en forma autónoma.

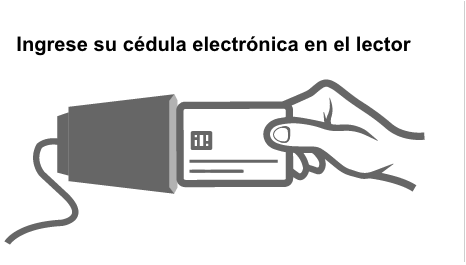
Para utilizar ID Uruguay, cada persona debe registrarse. A excepción de la autenticación con Cédula de Identidad Digital, que es un caso particular de autenticación con certificado digital y no requiere que la persona se registre.

Cada persona que complete el registro tendrá su cuenta asociada a credenciales básicas (usuario y contraseña) y, por lo tanto, puede establecer una sesión autenticándose con ellas. Adicionalmente, pueden firmar digitalmente un formulario para acceder a un nivel superior que garantiza su identidad dentro del sistema.

Proceso de autenticación

La autenticación de las personas se puede realizar por dos vías: autenticación mediante el uso de la Cédula Digital o autenticación mediante usuario y contraseña.

Autenticación mediante Cédula de Identidad Digital



La ciudadanía puede utilizar su Cédula de Identidad con chip para autenticarse en la plataforma. Para ello, es necesario contar con un lector de tarjetas inteligentes, donde se inserta la cédula, y colocar el PIN del documento una vez solicitado. El resto de los componentes necesarios para utilizar este medio se instalan automáticamente en el navegador de la persona la primera vez que se utilice el servicio.

Este método otorga a la identidad digital de la persona un nivel de seguridad superior, ya que dos factores para verificar su identidad: algo que tiene (cédula) y algo que sabe (PIN).

Autenticación mediante usuario y contraseña

Otra forma de autenticación es mediante un usuario y contraseña, obtenidos al momento del registro. Existen diferentes vías de registro, como se detallan a continuación.

Registro presencial: la persona se dirige en forma presencial a un Agente de Registro y este, a través de una interfaz especialmente delegada para él, lo registra en el sistema.

Autoregistro: la persona ingresa a la página de registro de ID Uruguay y sus datos. Se le envía una confirmación al correo electrónico ingresado y al aceptarla, se crea la cuenta básica del usuario en el IdP.

Certificado: una vez activada la cuenta, es posible firmar un contrato mediante Firma Digital Avanzada con el fin de asegurar una garantía de identidad de certificado, equivalente a la presencial.

Para completar el registro, se aceptan todos los documentos nacionales de identidad emitidos por países de América del Sur. Para el resto del mundo, es obligatorio el uso de pasaporte.

Garantías de identidad

La modalidad de autoregistro ofrece comodidad, pero no otorga demasiadas garantías respecto a la identidad de la persona dueña de la cuenta de usuario básica. Es por esto que, además de los datos ingresados, el sistema asigna automáticamente dos campos más que indican el nivel de validación de identidad con el que se cuenta en un momento dado:

- **Certificado:** si se realizó validación de identidad firmando el contrato de habilitación de usuario con la cédula de identidad.
- **Presencial:** si se realizó validación de identidad de forma presencial.

Acceder a más información sobre las garantías de identidad.

Requerimientos de integración con Usuario gub.uy

Los organismos que tengan el interés de utilizar Usuario gub.uy deberán adoptar una serie de consideraciones. En esta sección se presenta un resumen de los cambios que implica para los organismos el uso de Usuario gub.uy.

Definición de los Agentes de Registro

Se encarga de realizar el registro presencial de las personas, así como también elevar la identidad de los usuarios a un nivel presencial, en el caso de que se trate de un autoregistro.

Además de registrar usuarios, las personas con el rol de Agentes de Registro están capacitadas para evacuar dudas de la ciudadanía sobre este tema.

El organismo interesados deberá definir al menos una persona como Agente de Registro. El procedimiento para la creación de los Agentes de Registro es el siguiente:

- El organismo firma un contrato de adhesión en el cual se acuerdan las responsabilidades del organismo con respecto a la creación, uso y eliminación de personas con el rol de Agente de Registro.
- El responsable de la integración del organismo debe solicitar al responsable de la integración por parte de Agesic que se le asigne el rol de Agente de Registro a las personas designadas, indicando el documento de identidad de cada una de ellas.
- Las personas designadas deben tener una garantía de identidad presencial o certificada para poder ser dados de alta en el rol de Agente de Registro.
- El responsable de la integración de Agesic solicita los permisos correspondientes y notifica al organismo cuando el rol quede asignado a las personas indicadas.

Definición de usuarios aceptados

Son todas las personas que poseen un documento de identidad nacional emitido por algún país de América del Sur o los que cuenten con pasaporte de cualquier país del mundo. Además, cada organismo puede definir si aplica controles adicionales para el uso de su aplicación.

Mesa de Ayuda de Nivel 1

Todo organismo que desee integrarse al sistema necesita contar con personal capacitado para dar soporte de primer nivel, tanto a los usuarios como a los Agentes de Registro, en caso de que fuera necesario.

Integración de aplicaciones

Para concretar la integración con el sistema es necesario que los proveedores de servicio adapten sus aplicaciones según los requerimientos tecnológicos de Usuario gub.uy. Acceder a mayor información sobre la integración técnica con servicio de autenticación.

Marco jurídico

Para ejecutar el proceso de integración con la plataforma se debe firmar un Contrato de Adhesión que podrá encontrar con nombre "Habilitación de Usuario gub.uy.pdf".

Vías de soporte y capacitación

Manual para Agentes de Registro

Se cuenta con un Manual de Agentes de Registro donde se detallan las funcionalidades de la consola para administración de usuarios.

Preguntas frecuentes

Las FAQ son preguntas frecuentes acerca de todo lo relacionado con el sistema.Acceder a las preguntas frecuentes de Usuario gub.uy

Flujo de soporte al sistema de Usuario gub.uy

El flujo siempre comienza con alguna consulta disparada por los ciudadanos. Si estos se encuentran en el ámbito de una aplicación de los organismos, su consulta entra por medio de su mesa de ayuda. En el caso en que la consulta surja desde la pagina de Usuario gub.uy, el canal de resolución es por Atención a la Ciudadanía.

Mesa de Ayuda de organismo

Una vez aquí, o bien se resuelve el problema o se escala hacia el soporte de Agesic en caso de fallas técnicas o a Atención a la Ciudadanía en caso de que la Mesa de Ayuda del organismo no cuente con los recursos como para resolver el problema.

Atención a la Ciudadanía

Este canal recibe tanto consultas desde las Mesas de Ayuda de los organismos como de la ciudadanía. En caso de que alguna entrada se clasifique como "falla técnica", se deriva hacia el Soporte de Agesic; lo mismo sucede ante problemas no detectados como técnicos, pero de alta complejidad y a los que no se les encuentre solución.

Soporte Agesic

A este nivel deben llegar fallas exclusivamente técnicas de funcionamiento del sistema, a las cuales se les busca solución directamente con el responsable del servicio en caso de ser necesario.

Caso particular de consultas de Agentes de Registro

Para el caso particular de consultas que salgan directamente de los Agentes de Registro, el canal de entrada debe ser la Mesa de Ayuda de su organismo; luego, el flujo sigue el cauce natural explicado anteriormente.

Políticas de gestión de cuentas

- El usuario tiene 30 días a partir del registro para activar la cuenta utilizando el enlace de activación que es enviado a su correo electrónico. Luego de vencido el plazo, el enlace deja de funcionar y la cuenta se elimina; por lo tanto, si desea registrarse, debe repetir el procedimiento de registro.
- Políticas de contraseña: Debe tener entre 8 y 20 caracteres y no puede contener el nombre de usuario.

Tutorial sobre integración de aplicaciones .NET al IDP

Esta guía describe cómo integrar una aplicación web desarrollada sobre la plataforma Microsoft .NET con la solución de SSO Web ofrecida como servicio por el Estado uruguayo a todos sus organismos.

Esta guía permitirá a un desarrollador con conocimientos en ASP.NET modificar sus aplicaciones .NET existentes, de forma de incorporar un esquema de autenticación federado en ella, utilizando el Estándar de federación de identidades SAML V2.0.

Una implementación de referencia puede encontrarse enGithub

Antecedentes

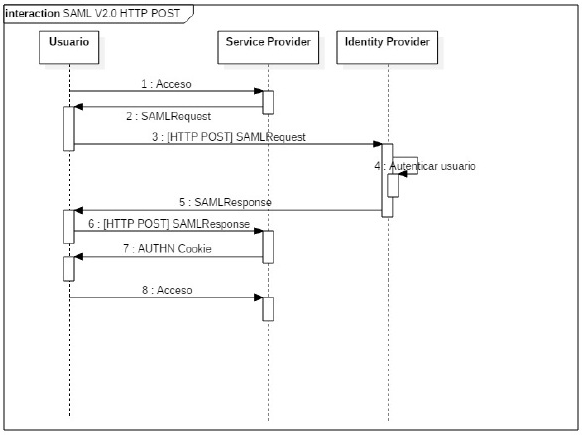
El estándar SAML se compone de varios capítulos en los cuales se definen, entre otros elementos, los formatos que deben seguir las aserciones que son generadas por los miembros de una federación. Y también los protocolos de comunicación que deben utilizarse para implementar un esquema de autenticación federado.

Dentro de la federación, se definen dos roles fundamentales:

- Identity Provider: Es un servicio proveedor de identidades centralizado capaz de autenticar al usuario y generar una aserción firmada. Esta aserción es luego enviada al servicio que desea autenticar a sus usuarios de forma federada, el cual confía en el proveedor de identidades. El estado uruguayo provee un proveedor de identidades en el cual deberán confiar las aplicaciones que participen de la federación.
- Service Provider: Es la aplicación (.NET en esta guía) que delega la autenticación de sus usuarios en el proveedor de identidades. El proveedor de identidades le indicará quién es el usuario autenticado y la aplicación deberá confiar en esta información.

Esta guía indica los pasos a seguir para convertir una aplicación .NET en un Service Provider que autentique a sus usuarios con el Identity Provider del Estado uruguayo, posibilitando la implementación de escenarios de Single Sign On entre aplicaciones del Estado, así como delegar los mecanismos de autenticación en un servicio que se ocupará de obtener las pruebas de identidad necesarias.

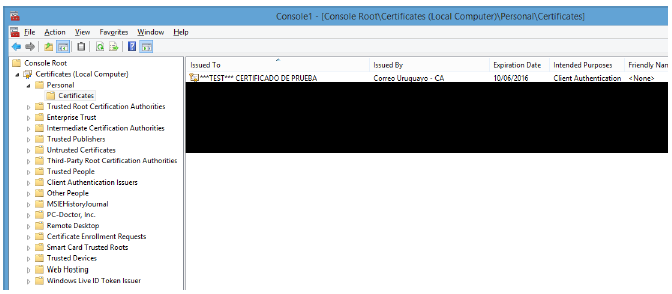
A continuación, se incluye un diagrama indicando la interacción que deberá darse entre el Service Provider y el Identity Provider para la autenticación de un usuario. Este diagrama se realiza en base a un binding definido en el estándar SAML V2.0, denominado HTTP-POST. Si bien otros binings son también soportados por el estándar, este es el recomendado para la autenticación federada en la plataforma.



Requisitos

Los componentes utilizados en esta guía requieren que la aplicación se encuentre desarrollada sobre la versión 3.5 del Microsoft .NET Framework o superior. Asimismo, deberá encontrarse publicada sobre HTTPS.

El Service Provider deberá contar con un certificado de servicio emitido por una CA autorizada, el cual deberá encontrarse instalado en el Certificate Store del equipo donde se encuentra la aplicación, incluyendo la clave privada. A efectos de esta guía, se asume que el certificado se encuentra instalado en el store Personal del equipo (LocalMachine / My). A continuación, se incluye una captura del Certificate Store:

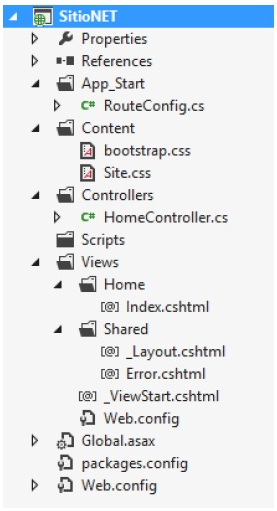


La cuenta configurada para la ejecución del sitio web deberá tener acceso a la clave privada del certificado para poder realizar la firma de lo SAML Requests enviados al IDP.

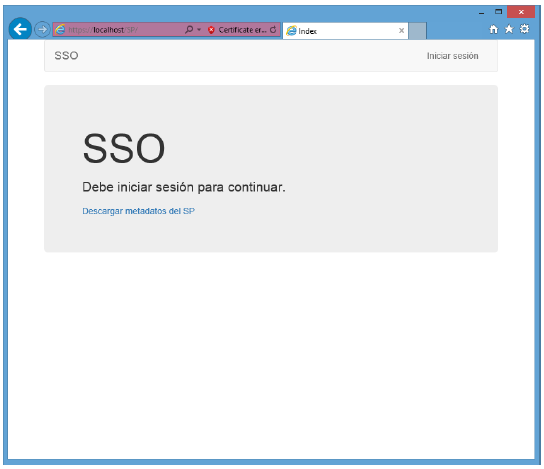
Aplicación .NET

Para la elaboración de esta guía, se asume una aplicación .NET existente desarrollada sobre ASP.NET MVC 4, la cual cuenta con un único controlador (Home Controller) que despliega en la página de inicio un aviso de login.

Los conceptos detallados en esta guía aplican tanto a una aplicación MVC como a una WebForms, ya que se utilizan componentes de ASP.NET que son comunes a ambos frameworks.



En el diagrama anterior se pueden observar los componentes del sitio inicial, el cual cuenta con la siguiente página de inicio:

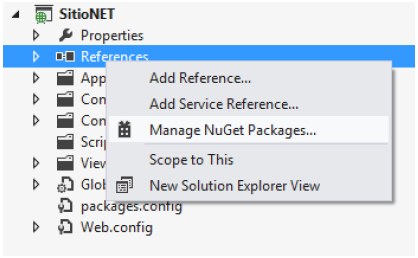


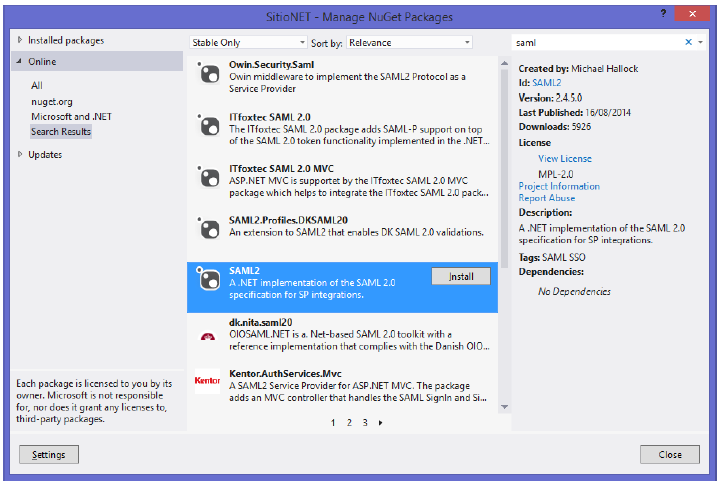
El código fuente de la solución de inicio se encuentra disponible junto con esta guía, de modo de permitir seguir los pasos detallados en ella sobre esta aplicación de ejemplo.

Integración con el Identity Provider de la plataforma

1. Incorporar componente SAML V2.0

El primer paso para integrar el proveedor de identidades de la plataforma es utilizar el gestor de paquetes NuGet para agregar un componente al proyecto que permite trabajar con los protocolos definidos en SAML V2.0. A continuación, se indica cómo realizar esta operación:





2. Modificar archivo de configuración

Agregar las siguientes líneas de configuración en el web.config:

<configSections> <section name="saml2" type="SAML2.Config.Saml2Section, SAML2" /> </configSections>

Reemplazar la sección <system.webServer> con el siguiente XML:

```
<system.webServer>
 <validation validateIntegratedModeConfiguration="true" />
 <handlers>
  <remove name="SAML2.Protocol.Saml20SignonHandler" />
  <remove name="SAML2.Protocol.Saml20LogoutHandler" />
  <remove name="SAML2.Protocol.Saml20MetadataHandler" />
  <add name="SAML2.Protocol.Saml20SignonHandler" verb="*" path="Login.ashx" type="SAML2.Protocol.Saml20SignonHandler, SAML2" />
  <add name="SAML2.Protocol.Saml20LogoutHandler" verb="*" path="Logout.ashx" type="SAML2.Protocol.Saml20LogoutHandler, SAML2" />
  <add name="SAML2.Protocol.Saml20MetadataHandler" verb="*" path="Metadata.ashx" type="SAML2.Protocol.Saml20MetadataHandler, SAML2" />
 </handlers>
</system.webServer>
```

Agregar las siguientes líneas de configuración en el web.config, reemplazando los textos entre paréntesis rectos por los propios del servicio a integrar:

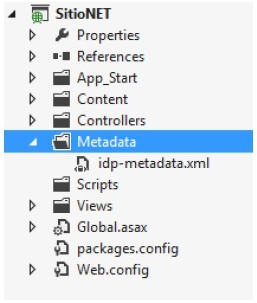
```
<saml2>
 <allowedAudienceUris>
  <audience uri="[Identificación servicio]" />
 </allowedAudienceUris>
 <serviceProvider id="[Identificación servicio]" server="[Url del servicio]">
  <signingCertificate storeName="My" storeLocation="LocalMachine" findValue="[Subject del certificado a usar por el Servicio]" x509FindType="FindBySubjectName" />
  <endpoints>
   <endpoint localPath="[URL Servicio]/Login.ashx" type="SignOn" redirectUrl="~/" />
   <endpoint localPath="[URL Servicio]/Logout.ashx" type="Logout" redirectUrl="~/" />
   <endpoint localPath="[URL Servicio]/Metadata.ashx" type="Metadata" />
  </endpoints>
  <nameIdFormats allowCreate="true">
  <add format="urn:oasis:names:tc:SAML:1.1:nameid-format:transient"/>
  </nameIdFormats>
 </serviceProvider>
 <identityProviders metadata="Metadata">
  <add id="[Entity ID del IdP]" default="true" omitAssertionSignatureCheck="true">
   <endpoints>
    <endpoint type="Logout" url="[Endpoint de SLO del IdP]" binding="Redirect" />
   </endpoints>
  </add>
 </identityProviders>
 <metadata>
  <contacts>
   <contact type="Administrative" company="" givenName="" surName="" email="" phone="" />
  </contacts>
 </metadata>
 <actions>
  <clear/>
  <action name="SetSamlPrincipal" type="SAML2.Actions.SamlPrincipalAction, SAML2" />
  <action name="UIDParserAction" type="[Namespace del Sitio].UIDParserAction, [Nombre del sitio]" />
  <action name="Redirect" type="SAML2.Actions.RedirectAction, SAML2" />
 </actions>
</saml2>
```

Modificar la sección “authentication” dentro de <system.web> para configurar autenticación usando SAML v2.0.

<authentication mode="Forms"> <forms loginUrl="~/Login.ashx" timeout="2880" /> </authentication>

3. Incorporar metadatos del servicio

Crear una carpeta dentro del proyecto denominada “Metadata” e incluir dentro de ella el archivo descriptor del Identity Provider entregado por Agesic (idp-metadata.xml).



4. [MVC] Modificar rutas configuradas

Si se trata de un proyecto MVC, es necesario modificar las rutas para permitir acceder a los endpoints definidos para la comunicación utilizando SAML-P: Login.ashx, Logout.ashx y Metadata.ashx.

Incorporar las siguientes líneas en el archivo RouteConfig.cs dentro de la carpeta App_Start, al comienzo del método RegisterRoutes:

```
routes.IgnoreRoute("Login.ashx");
routes.IgnoreRoute("Logout.ashx");
routes.IgnoreRoute("Metadata.ashx");
```

5. Incorporar action para la obtención del UID

A continuación, se incluye el código fuente de una clase llamada UIDParserAction, que deberá incorporarse dentro de la carpeta App_Code del sitio, la cual permite obtener el nombre del usuario a partir del atributo SAML “UID”, e incorporarlo a la identidad de la sesión actual.

```
public class UIDParserAction : IAction
{
 private string _name = "UIDParserAction"; private const string UID_ATTRIBUTE_NAME = "uid";
 public string Name
 {
 get { return _name; }
 set { _name = value; }
 }
 public void SignOnAction(AbstractEndpointHandler handler, HttpContext context, Saml20Assertion assertion)
 {
  if (!Saml20Identity.Current.HasAttribute(UID_ATTRIBUTE_NAME))
  {
  throw new ArgumentException("El SAML Assertion no contine un atributo uid.");
  }
  if (!Saml20Identity.Current[UID_ATTRIBUTE_NAME].Any(x => x.AttributeValue.Length > 0))
  {
   throw new FormatException("El SAML assertion no contiene un valor para el atributo uid.");
  }
  string uid = Saml20Identity.Current[UID_ATTRIBUTE_NAME].FirstOrDefault(x => x.AttributeValue.Length > 0).AttributeValue.FirstOrDefault();
  FormsAuthentication.SetAuthCookie(uid, false);
 }
```

```
 public void LogoutAction(AbstractEndpointHandler handler, HttpContext context, bool IdPInitiated)
 {
  FormsAuthentication.SignOut();
 }
}
```

Esta clase es incorporada como Action en el archivo de configuración del punto 2. Es importante asegurar la coincidencia de los nombres para que la aplicación funciones correctamente.

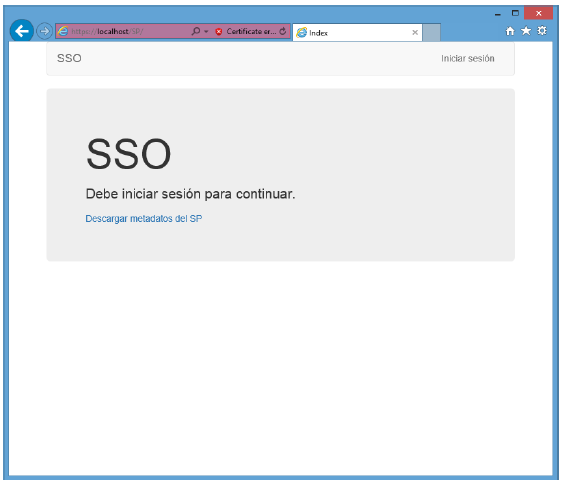
6. Obtener metadatos del servicio local y enviar a Agesic

Ejecutar la aplicación web. Desde ella, se podrá acceder a la URL /Metadata.ashx, la cual generará un archivo descriptor xml que podrá ser descargado localmente, y enviado a Agesic para que se configure el service provider dentro de la federación.

Mientras el SP no sea dado de alta, no será posible enviar SAML Requests al IdP, ya que este rechazará las solicitudes.

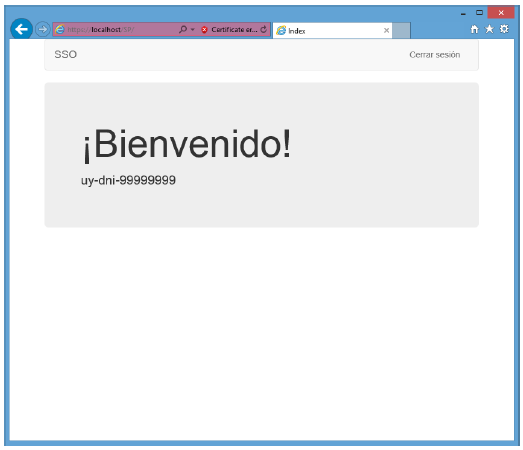
7. Acceder al servicio y probar la integración

Una vez que el SP fue dado de alta en el IdP, se podrá probar el Single Sign-On y Single Logout desde ella. A continuación, se incluyen capturas con ejemplos de estos flujos.



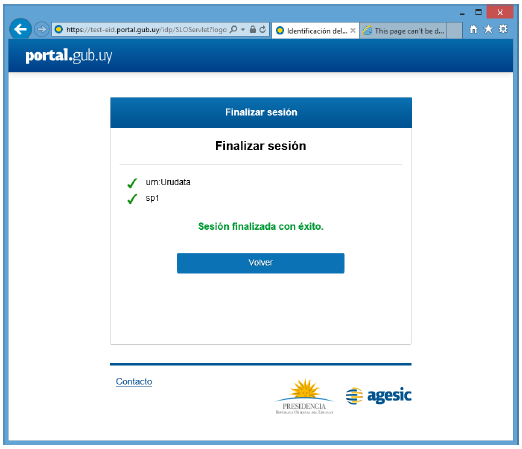
Al hacer clic en “Iniciar sesión”, se reenviará automáticamente al IdP del Estado. Si el usuario ya tenía una sesión abierta con el IdP, no se solicitará su reautenticación y será retornado a la pantalla de inicio con un SAML Assertion generado por el IdP. De lo contrario, se mostrará la siguiente pantalla:

Luego de autenticarse, se redirigirá al usuario automáticamente a la home del sitio con el Assertion generado por el IdP. A partir de él, se obtendrá el nombre de usuario dentro del atributo UID:



Para acceder al nombre de usuario, la aplicación utiliza el mecanismo habitual de ASP.NET (HttpContext.Current.User.Identity), con lo cual la autenticación por SAML es transparente para la aplicación. Los flujos posteriores de autorización serán los ya implementados por la aplicación, trabajando con el nombre de usuario retornado por el IdP.

El usuario podrá cerrar sesión desde la aplicación, la cual envía un request de Single Logout al IdP que cierra sus sesiones en todas las aplicaciones federadas donde haya sesiones abiertas. Asimismo, el logout puede ser iniciado desde otra aplicación, generando los mismos resultados.



Anexo I - Diagnóstico de problemas

Si existieran problemas con la federación, es posible habilitar el logging de eventos para poder diagnosticarlos. Este logging permitirá acceder a los mensajes enviados y recibidos con el IdP, así como eventos a internos de cada uno de los flujos.

1. Crear clase EventLogger

Crear un archivo EventLogger.cs en la carpeta App_Code del sitio web, con el siguiente contenido:

```
public class EventLoggerFactory : ILoggerFactory
{
 private static readonly EventLogger logger = new EventLogger();
 public IInternalLogger LoggerFor(Type type)
 {
  return logger;
 }
 public IInternalLogger LoggerFor(string keyName)
 {
  return logger;
 }
}

public class EventLogger : IInternalLogger
{
 private const string LOGGER = "SAML2";
 public EventLogger()
 {
  if (!EventLog.SourceExists(LOGGER)) EventLog.CreateEventSource(LOGGER, "Application");
 }
 private void WriteLog(string severity, string message, Exception ex)
 {
  string innerMessage = message;
  if (ex != null) string.Concat(innerMessage, "\r\n", ex.ToString()); EventLog.WriteEntry(LOGGER, string.Format("{0} - {1} - {2}", severity, DateTime.Now.ToString("dd/MM/yyyy HH:mm:ss"), innerMessage));
 }
 public void Debug(object message, Exception exception)
 {
  WriteLog("DEBUG", message.ToString(), exception);
 }
 public void Debug(object message)
 {
  WriteLog("DEBUG", message.ToString(), null);
 }
 public void DebugFormat(string format, params object[] args)
 {
  if (args.Length > 0) WriteLog("DEBUG", string.Format(format, args), null);
 }
 public void Error(object message, Exception exception)
 {
  WriteLog("ERROR", message.ToString(), exception);
 }
 public void Error(object message)
 {
  WriteLog("ERROR", message.ToString(), null);
 }
 public void ErrorFormat(string format, params object[] args)
 {
  if (args.Length > 0) WriteLog("ERROR", string.Format(format, args), null);
 }
 public void Fatal(object message, Exception exception)
 {
  WriteLog("FATAL", message.ToString(), exception);
 }
 public void Fatal(object message)
 {
  WriteLog("FATAL", message.ToString(), null);
 }
 public void Info(object message, Exception exception)
 {
  WriteLog("INFO", message.ToString(), exception);
 }
 public void Info(object message)
 {
  WriteLog("INFO", message.ToString(), null);
 }
 public void InfoFormat(string format, params object[] args)
 {
  if (args.Length > 0) WriteLog("INFO", string.Format(format, args), null);
 }
 public bool IsDebugEnabled
 {
  get { return true; }
 }
 public bool IsErrorEnabled
 {
  get { return true; }
 }
 public bool IsFatalEnabled
 {
  get { return true; }
 }
 public bool IsInfoEnabled
 {
  get { return true; }
 }
 public bool IsWarnEnabled
```

```
{
 get { return true; }
}
public void Warn(object message, Exception exception)
{
 WriteLog("WARN", message.ToString(), null);
}
public void Warn(object message)
{
 WriteLog("WARN", message.ToString(), null); }
 public void WarnFormat(string format, params object[] args) { if (args.Length > 0) WriteLog("WARN", string.Format(format, args), null);
}
}
```

2. Configurar EventLogger en el web.config

Incorporar la siguiente línea dentro del tag <saml2> del archivo de configuración:

<logging loggingFactory="[Namespace del sitio].EventLoggerFactory, [Nombre del sitio]" />

3. Acceder al Event Log

Luego de realizar esta configuración, se podrá acceder al EventLog de Windows (eventvwr.msc) para encontrar las entradas de log, dentro de “Application”.

Recursos y materiales técnicos ID Uruguay y Usuario.gub.uy

Información técnica ID Uruguay y Usuario.gub.uy

Información Técnica de Cédula de Identidad con chip

Introducción

En el presente documento se incluye toda la información técnica de la cédula de identidad uruguaya. La cédula de identidad con chip se comenzó a emitir en 2015. En noviembre de 2022 se actualizó el plástico y esto implicó algunos cambios. Ente otros, es posible firmar e identificarse digitalmente igual que como el modelo anterior, pero por proximidad a utilizando el protocolo PACE sobre NFC. En los últimos capítulos de este documento se agrega información relativa a este cambio (MOC cédula 2015, modificaciones para el MOC cédula 2022 y NFC – PACE para cédula 2022).

A continuación, se explica cómo distinguir las versiones de la cédula IAS CLASSIC v4 (2015) e IAS CLASSIC v5 (2022).

IAS CLASSIC v4:

Software Version

The command returns the software version in TLV format as follows:

Tag	Length	Meaning	Display Value	ASCII Value
C0h	0Eh	Applet Label	"IAS Classic v4"	49 41 53 20 43 6C 61 73 73 69 63 20 76 34h
C1h	07h	Applet Version	4.0.x.y	34 2E 30 2E x 2E yh

The DO with tag C0h contains the product reference.

The DO with tag C1h contains the product version.

The software version is an example and will evolve as the product evolves.

EL DO con el tag C0h contiene la referencia del producto

El DO con el tag C1h contiene la versión del producto

La versión de software expuesta en esta imagen es un ejemplo e irá evolucionando a medida que lo haga el prooducto.

IAS CLASSIC v5:

Software Version

The command returns the software version in the TLV format as follows:

Tag	Length	Meaning	Display Value	ASCII Value
C0h	0Eh	Applet Label	IAS Classic Applet V5	49h 41h 53h 20h 43h 6Ch 61h 73h 73h 69h 63h 20h 76h 35h
C1h	07h	Applet Version	M.m.r.p.c	M: major version m: minor version r: release version ('0-9') p:product c: configuration. 'C' for Full, 'O' for Compact. 35h 2Eh 32h 2Eh 30h 2Eh 41h 2Eh 4Fh (Compact configuration on MultiappV5.0) 35h 2Eh 32h 2Eh 30h 2Eh 41h 2Eh 43h (Full configuration on MultiappV5.0)

The DO with tag C0h contains the ASCII value of the product reference.

The DO with tag C1h contains the ASCII value of the product version.

EL DO con el tag C0h contiene el valor ASCII de la referencia del producto

El DO con el tag C1h contiene el valor ASCII de la versión del producto

Resumen de las respuestas de las versiones:

Operation	APDU	Response	Analysis
Select IAS Classic instance	00A40400 0C A0000000184000000163 4200	9000	
Send the get data TAG 7F30	00CA7F30 Le=00	**Response on IAS Classic V5 –> Tarjetas duales** 7F301B C00E 4941532043 6C6173736963207635 C109 352E322E302E412E*XX* **Response on IAS Classic V4 –> Tarjetas hibridas** 7F3019 C00E 4941532043 6C6173736963207634 C107 342E302E302E41	**Response on IAS Classic V5 –> Tarjetas duales** 4941532043 6C6173736963207635 352E322E302E412E*XX* → 5.2.0.A.*?* **Response on IAS Classic V4 –> Tarjetas hibridas** 4941532043 6C6173736963207634 342E302E302E41 → 4.0.0.A

¿Qué es un APDU y por qué es relevante?

El Application Protocol Data Unit (APDU) es la unidad de comunicación entre un lector de tarjetas inteligentes y una tarjeta inteligente (en inglés *smart card)*. Dado que la Cédula de Identidad Digital es, en esencia, una *smart card* conformante con el estándar ISO 7816, esta es la unidad lógica utilizada para comunicarse con ella a bajo nivel.

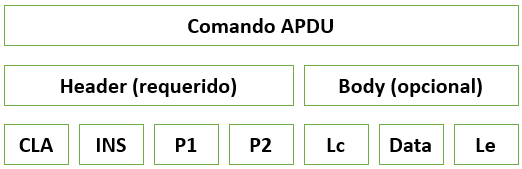
Si bien existen otras vías para comunicarse con la Cédula de Identidad Digital, como drivers PKCS#11, plug-ins, bibliotecas, etc., todas estas vías se implementan utilizando APDU, es decir, son *wrappers*. Poder interactuar con las aplicaciones de la Cédula de Identidad Digital a través de APDU tiene la ventaja de que otorga la máxima flexibilidad a nivel de plataformas en las que se puede implementar la interacción. Y, además, hay operaciones como el *Match On Card (MOC)* para las que no se cuenta con una biblioteca de más alto nivel, por lo que solo pueden realizarse a través de esta vía.

Estructura de un APDU

Hay dos tipos de APDU: comandos y respuestas. Los comandos APDU los envía el lector a la tarjeta, mientras que las respuesta APDU las envía la tarjeta al lector.

La estructura de un APDU está definida en los estándares ISO/IEC 7816.

Estructura de un comando APDU



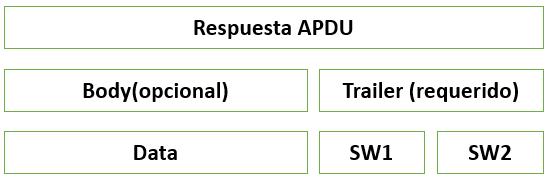
La trama APDU de tipo comando consta de los siguientes campos:

- **CLA:** Byte de clase
- **INS:** Byte de instrucción
- **P1, P2:** Parámetros
- **Lc:** tamaño del bloque de datos
- **Data**
- **Le:** Tamaño de la respuesta esperada

Los cuatro primeros son obligatorios, mientras que los relacionados con los datos y la respuesta esperada son opcionales. A partir del byte de instrucción, la tarjeta sabe qué es lo que se le pide.

Contienen una cabecera obligatoria de 4/5 bytes, y entre 0 y 255 bytes de datos.

Estructura de una respuesta APDU



La trama APDU respuesta consta de los campos:

- **Data**
- **SW1, SW2:** Palabra de estado, donde se codifica el estado de la operación (correcta, error criptográfico, error general). Una vez más, los datos son opcionales, pero el código de estado es obligatorio.

Contienen una palabra de estado obligatoria de 2 bytes y entre 0 y 255 bytes de datos.

Casos de APDU

Existen cuatro casos definidos para comandos APDU donde la estructura varía de la siguiente forma:

1. El largo Lc es nulo; por lo tanto, los campos Lc y data van vacíos. El largo Le es también nulo; por lo tanto, el campo Le va vacío. Por consecuencia, el campo Body es vacío.
2. El largo Lc es nulo; por lo tanto, los campos Lc y data van vacíos. El largo Le no es nulo; por lo tanto, el campo Le está presente. Por consecuencia, el campo Body es el Le.
3. El largo Lc no es nulo; por lo tanto, el campo Lc está presente y define el largo de campo Data también presente. El largo Le es nulo; por lo tanto, el campo Le es vacío. Por consecuencia, el Body contiene al campo Lc seguido del campo data.
4. El largo Lc no es nulo; por lo tanto, el campo Lc está presente y define el largo del campo Data también presente. El largo Le no es nulo; por lo tanto, el campo Le esta presente. Por consecuencia, el Body consiste en el campo Lc seguido del campo Data y del campo Le.

Caso 1: No incluye data, No requiere respuesta.	CLA	INS	P1	P2		
Caso 2: No incluye data, Requiere respuesta.	CLA	INS	P1	P2	Le	
Caso 3: Incluye data, No requiere respuesta.	CLA	INS	P1	P2	Lc	Data
Caso 4: Incluye data, Requiere respuesta.	CLA	INS	P1	P2	Lc	Data	Le

Formato TLV para datos

Para ciertos tipos de operaciones, la información dentro del campo Data en los comandos y respuestas APDU está representada en formato TLV "Tag Length Value" (Tipo Longitud Valor).

Este formato permite organizar mejor la información y tiene la siguiente lógica:

- **Tag:** Representa la información contenida en el campo Value.
- **Length:** Largo en bytes de la información contenida en el campo Value.
- **Value:** Información.

Ejemplo de un TLV que contiene el número de documento de la persona obtenido como parte del caso de uso de extracción de los datos de identificación:

Tag	Length	Value
5F01	09	NumeroDeDocumento

Como se ve en el ejemplo, el TAG 5F01 representa al número de documento que tiene largo en bytes 09.

El campo Length viene habitualmente en un byte, pero cuando el tamaño del Value es mayor a 0x7F (127), en el campo Length el primer bit será 1 y los restantes indican el tamaño en bytes restantes del campo. Esto se traduce a que será 0x80 + tamaño restante de LENGTH. Entonces:

Si 0 <= LENGTH < 0x7F(127) => LENGTH 1 byte = XX

Si 0x7F < LENGTH < 0xFF(255) => LENGTH 2 byte = 81 XX

Si 0xFF < LENGTH < 0xFFFF(65535) => LENGTH 3 byte = 82 XX XX

Por ejemplo, en el caso de obtener la imagen de la persona, el campo Length contendrá un byte con el número 0x82 seguido del largo representado en 2 bytes; por ejemplo, T= 0x3F01 L= 0x8223FE D= 9214 bytes(0x23FE). También ocurre esto cuando se envían 42 o más minucias al match on card.

Comandos APDU y casos de uso

En esta sección se presentan los comandos APDU que luego serán utilizados en conjunto para la construcción de casos de uso como por ejemplo extraer los datos de identificación del documento.

Selección del applet de firma IAS - SELECTIAS

El comando select selecciona el Applet IAS de firma por su AID (Application Identifier) dentro del eID.

Es precondición para cualquier otro comando APDU descripto en este documento que se haga un select del Applet IAS antes.

La estructura de un comando de selección de aplicación por su AID es la siguiente:

	CLA	INS	P1	P2	Lc	Data
Caso 3: Incluye data, No requiere respuesta.	00	A4	04	00	Lc	AID

El comando APDU que selecciona el applet de firma IAS:

CLA	INS	P1	P2	Lc	Data
00	A4	04	00	0C	A0 00 00 00 18 40 00 00 01 63 42 00 00

IMPORTANTE: El comando selectIAS se debe ejecutar al inicio antes que cualquier otro comando

Selección de un archivo por el ID de archivo - Selectfile

La información contenida en el documento como por ejemplo los datos de identificación o certificado de la persona se encuentra distribuida en archivos y cada archivo se identifica por un ID.

Para realizar la lectura de esta información se debe seleccionar el archivo por su correspondiente ID utilizando el comando APDU*selectFile*.

Luego de realizada la selección del archivo mediante su ID, el comando APDU*readBinary* se envía para extraer los datos. Para el envío del comando*readBinary* se debe conocer el tamaño del archivo a leer.

Este dato necesario para la lectura del archivo se encuentra en lo que se denomina FCI Template. El FCI Template de cada archivo se obtiene de la respuesta APDU al comando*selectFile*.

Por ejemplo, para leer los datos del certificado del usuario debemos seleccionar primero el archivo correspondiente y obtener su FCI Template.

El FCI Template contiene los siguientes datos de relevancia entre otros:

- Nombre del archivo.
- Tamaño del archivo.

El tamaño del archivo es necesario para realizar la lectura de la información utilizando el comando*readBinary*.

Entonces, si se quisieran leer los datos del certificado digital contenido en el eID se deben realizar los siguientes pasos:

1. Obtener el archivo FCI Template del certificado a través de su ID y la operación*selectFile*.
2. Extraer del FCI Template obtenido en el comando APDU respuesta al comando*selectFile* el tamaño en bytes del certificado.
3. Conociendo el tamaño del archivo que contiene el certificado se ejecuta el comando APDU*readBinary* para obtener la información del certificado.
4. Estructura del comando *selectFile*:

Caso 4:
Incluye data,
Requiere respuesta.

CLA	INS	P1	P2	Lc	Data	Le
00	A4	00	00	02	FileID	Le

Ejemplo de un comando de *selectFile* para seleccionar el FCI Template del certificado.

CLA	INS	P1	P2	Lc	Data	Le
00	A4	00	00	02	B0 01	Le

En la respuesta viene el archivo FCI Template correspondiente al certificado. El ID del archivo que contiene al certificado es **"B001"** como se ve en el ejemplo.

0	6Fh	Tag del FCI Template
1	L	Largo de FCI Data
2	81h	Tag de largo de archivo
3	02h	Largo del tamaño del archivo
4–5	Tamaño de archivo	Valor del tamaño del archivo
6	82h	FDB Tag
7	01h	Largo del FDB
8	FDB	Valor del FDB
9	83h	Tag del ID de archivo
10	02h	Largo del archivo ID
11–12	ID Archivo	Valor del ID archivo
13	8Ah	Tag de “Life Cycle Status byte for file"
14	01h	Largo de “Life Cycle Status byte for file”
15	Var.	Valor de “Life Cycle Status byte for file”
16	8Ch	Tag de atributos de seguridad
17	L	Largo de atributos de seguridad
18	AMB	Modo acceso a bytes
19–(18+X)	SCBs	Condición de seguridad bytes (X)

Como se ve en la imagen, el FCI template contiene el tamaño del archivo a leer (offset 4-5), necesario y suficiente para su lectura.

Lectura de un binario - Readbinary

La lectura de un binario se realiza sobre un archivo previamente seleccionado con el comando APDU *selectFile*.

Cada comando *readBinary* puede leer como máximo 0xFF bytes de un archivo, si el tamaño del archivo es superior a 0xFF se deberán enviar tantos comandos APDU *readBinary* como sean necesarios.

Por ejemplo, si el tamaño del certificado a leer (obtenido del FCI Template del archivo) es de 3.000 bytes, se deberán enviar 3000/255 + 1 comandos de *readBinary* para completar la lectura del certificado. En este caso, 12.

Estructura del comando *readBinary*:

Caso 2:
No Incluye data,
Requiere respuesta.

CLA	INS	P1	P2	Le
00	B0	P1	P2	Le

Donde P1 P2 es el offset en hexadecimal donde empezar a leer datos.

Ejemplo de un comando *readBinary* obteniendo los primeros 255 bytes (0xFF) del archivo seleccionado:

CLA	INS	P1	P2	Le
00	B0	00	00	FF

Para los siguientes READ_BINARY se debe sumar 0xFF al offset y continuar leyendo datos, por ejemplo si L es el largo del archivo en hexadecimal:

00 B0 00 FF Le=FF

00 B0 01 FE Le= FF

00 B0 02 FD Le=FF

.

.

.

00 B01 L1 L2 Le=L3

Donde:

L1 || L2 = 0xFF x cociente(L/0xFF)

L3 = resto(L/0xFF)

Verificación de PIN - VERIFYPIN

La operación de verificación de PIN es requisito previo a las operaciones de firma.

Caso 3:
Incluye data,
No requiere respuesta.

CLA	INS	P1	P2	Lc	Data
CLA	20	00	P2	Lc	Pin de Verificación

- **CLA:**
  - 00h Transmisión en plano
  - 0Ch Transmisión sobre canal seguro
- **INS:**
  - 20h Fijo para la operación de verificación.

- **P1:**
  - 00h Modo verificación
- **P2:**
  - 11h Para global PIN
- **Lc:**
  - 0Ch Siempre espera 12 bytes de largo, se agregan 0's al final como padding.
- **Data:**
  - El PIN va en codificado en ASCII y largo 12 bytes.

Ejemplo de un comando APDU para verificar el PIN 1234:

CLA	INS	P1	P2	Lc	Data
00	20	00	11	0C	31 32 33 34 00 00 00 00 00 00 00 00

Validar PIN verificado - ISVERIFIEDPIN

Valida si el PIN se encuentra verificado.

Caso 1:
No Incluye data,
No requiere respuesta.

CLA	INS	P1	P2	Le
CLA	20	00	P2	00

- **CLA:**
  - 00h Transmisión en plano
  - 0Ch Transmisión sobre canal seguro
- **INS:**
  - 20h Fijo para la operación de verificación.
- **P1:**
  - 00h Modo verificación
- **P2:**
  - 11h Para global PIN
- **Le:**
  - 00h No espera respuesta pero se debe enviar Le o Lc con 00h.
- **Data:**
  - Ausente.

Ejemplo de un comando APDU para verificar si el PIN se encuentra verificado:

CLA	INS	P1	P2	Le
00	20	00	11	00

Validación de la huella digital de la persona MATCH ON CARD

Realiza la operación de *Match On Card.* Validación 1 a n comparando las minucias extraídas por un lector de huellas versus las minucias almacenadas en el chip del documento electrónico.

Las minucias deben ser extraídas conforme al estándar **ISO/IEC 19794-2** Compact Card, sin cabezales, es decir, solamente la información de las minucias.

En este formato, cada punto característico de la huella dactilar se corresponde a una minucia, que a su vez es codificada en 3 bytes: uno para la coordenada X, otro para la coordenada Y, y un tercero para el tipo de punto (valle, bifurcación, etc) y el ángulo de inclinación de su característica. En ese formato, un conjunto de minucias debe ser entonces siempre de una cantidad de bytes múltiplo de 3, de la forma:

X1|**Y1**|**T1**|**X2**|**Y2**|**T2**|....|**Xn**|**Yn**|**Tn** *(el | marca la división de bytes)*

Las minucias deben ordenarse primero *ascendente por la coordenada Y* y luego, en caso de dos minucias con igual Y, *ascendente por la coordenada X;* de lo contrario, el match fallará con error 6CXX (siendo XX la cantidad de intentos restante) o 6A80, que indica error de formateo de datos. Se recomienda tener especial cuidado con lenguajes de programación que manejen bytes con signo como Java para la extracción de minucias. En esos casos, cualquier comparación susceptible al signo (< o >) debe ser hecha enmascarando los bytes con un *bitwise AND* (AND bit a bit, & en Java y C/C++) con 0xFF, lo cual fuerza a que sean considerados como bytes sin signo.

El largo máximo de las minucias soportado es de 192 bytes, es decir, 64 minucias de 3 bytes cada una.

Estructura del comando APDU para la operación *Match On Card.*

Caso 3:
Incluye data,
No requiere respuesta.

CLA	INS	P1	P2	Lc	Data
00	21	00	21	Lc	TLV Data

Tag	Length	Value
7F2E	L	TLV Minucias

Tag	Length	Value
81	L	Minucias

- **CLA:**
  - 00h Transmisión en plano
  - 0Ch Transmisión sobre canal seguro
- **INS:**
  - 21h Fijo para la operación de *Match on Card.*
- **P1:**
  - 00h Fijo para la operación de *Match on Card.*
- **P2:**
  - 21h Fijo para la operación de *Match on Card.*

- **Lc:**
  - Largo en bytes del TLV para validación *Match on Card.*
- **Data:**
  - TLV para la validación *Match on Card* que contiene las minucias extraídas de una huella.

Ejemplo de un comando APDU para verificación *Match On Card*:

CLA	INS	P1	P2	Lc	Data
00	21	00	21	Lc	72 FE 4F 81 4D 761460f21474971...

Importante: La cédula se bloquea luego de cinco intentos consecutivos de match on card sin éxito, devolviendo el error 0x6984

Las minucias para la versión 3 de los especímenes se solicitan a identificacion.electronica@agesic.gub.uy

Extracción de los datos de identificación de la persona

Los datos de identificación de la persona dentro del documento son los que muestra el plástico (menos la imagen de la huella e imagen de la firma). Ver Cédula de Identidad Digital.

Estos datos se encuentran en archivos que deberán ser leídos con las operaciones *selectFile* y *readBinary*.

La información obtenida de las respuestas APDU a los comandos *readBinary* para cada archivo está codificada en formato TLV.

A continuación, se presentan las operaciones de *selectFile* necesarias para la obtención de la información de identificación y la especificación de los TLV correspondientes a cada archivo.

Los datos del campo value se encuentran codificados en formato ASCII con excepción de la fecha de expedición.

selectFile del archivo que contiene el número de documento, ID 7001:

CLA	INS	P1	P2	Lc	Data	Le
00	A4	04	00	02	70 01	Le

Información en formato TLV obtenida luego de las operaciones de *readBinary*:

Tag	Length	Value
5F01	09	NumeroDeDocumento

selectFile del archivo que contiene los datos biográficos, ID 7002:

CLA	INS	P1	P2	Lc	Data	Le
00	A4	04	00	02	70 02	Le

Información en formato TLV obtenida luego de las operaciones de *readBinary*:

Tag	Length	Value
1F01	L	PrimerApellido
1F02	L	SegundoApellido
1F03	L	Nombres
1F04	03	Nacionalidad (ISO 3166)
1F05	08	FechaDeNacimiento (DDMMAAA)
1F06	L	LugarDeNacimiento (Ciudad/Pais en ISO 3166)

selectFile del archivo que contiene la imagen en formato JPG, ID 7004:

CLA	INS	P1	P2	Lc	Data	Le
00	A4	04	00	02	70 04	Le

Información en formato TLV obtenida luego de las operaciones de *readBinary*: (Length = 2 bytes)

Tag	Length	Value
3F0182	L	ImagenJPG

selectFile del archivo que contiene el MRZ, ID 700B:

CLA	INS	P1	P2	Lc	Data	Le
00	A4	04	00	02	70 0B	Le

Información en formato TLV obtenida luego de las operaciones de *readBinary*:

Tag	Length	Value
7F01	L	MRZ

Firma Digital

Antes de ejecutar alguna operación de Firma Digital se debe haber ejecutado la operación de verificación de PIN

La operación de Firma Digital consta del cifrado de un hash utilizando la clave privada del documento digital y un algoritmo seleccionado. Los algoritmos soportados para la operación de Firma Digital son RSA y ECDSA, aunque actualmente las claves de la Cédula Digital son RSA de 2048 bits.

De forma macro, los pasos para realizar la operación de firma son los siguientes:

1- Se realiza un hash del mensaje a firmar, el hash puede ser realizado de tres formas:

- Externo al eID (el más utilizado).
- Utilizando el eID.
- Parcialmente externo y utilizando el eID.

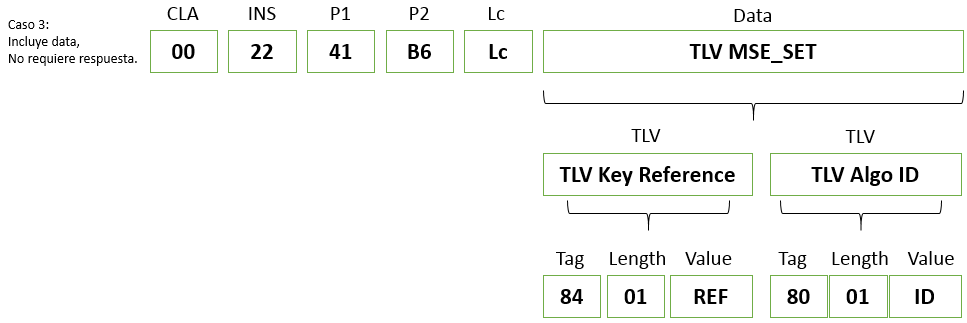
2. Se selecciona el algoritmo de firma y hash utilizando el comando APDU *MSE_SET_DST*.

3. Se envía el hash al documento eID mediante el comando APDU *PSO_HASH*.

4. El hash es cifrado con la clave privada del documento eID utilizando el algoritmo y parámetros seleccionados en los pasos anteriores mediante el comando APDU *PSO–Compute Digital Signature*.

5. Se obtiene el hash cifrado como resultado del paso anterior.

Comando APDU *MSE_SET_DST*:

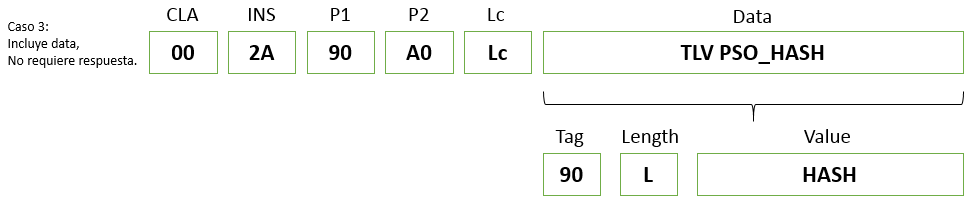


AlgoIDs:

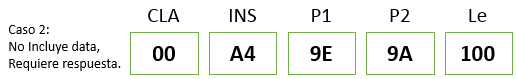
01, 02, 03 = No hash
32, 35 = SHA224
41, 42, 45 = SHA256
52, 55 = SHA384
62, 65 = SHA512
x1=RSA padding ISO9796-2
x2=RSA padding PKCS#1v1.5
x3=RSA padding RFC2409
x5=RSA PSS

Por ejemplo, para utilizar RSA con hash SHA-256 y padding PKCS#1v1.5 utilizaremos el AlgoID=42

Comando APDU *PSO_HASH*: (Hash realizado fuera de la tarjeta)



Comando APDU *PSO–Compute Digital Signature*:



Como la clave utilizada es RSA de 2.048 bits, se espera un resultado de largo 256 bytes = 0x100.

Ejemplo de Firma Digital

A continuación, se presentan como ejemplo los comandos APDU enviados para la Firma Digital de un hash de un mensaje generado de forma externa con el algoritmo SHA-256.

Mensaje a firmar: "Ejemplo de firma en APDU utilizando el nuevo documento eID"
HASH en formato hexadecimal del mensaje con el algoritmo **SHA256:** "A3D00CBE708B435D6E7B898770378FD54319B2FD7571C769DB414094E7008624".

MSE_SET_DST:

CLA	INS	P1	P2	Lc	Data
00	22	41	B6	06	84 01 01 80 01 02

PSO_HASH:

CLA	INS	P1	P2	Lc	Data
00	2A	90	A0	20	A3D00CBE708B435D6E7B898770…

PSO-Compute Digital Signature:

CLA	INS	P1	P2	Le
00	A4	9E	9A	100

Hash cifrado con la clave privada del documento de identidad digital obtenido como resultado al comando PSO-CDS: "A91283AA1239213...".

Repositorios públicos de ejemplo

Existen estos dos desarrollos públicos en Github que utilizan APDU.

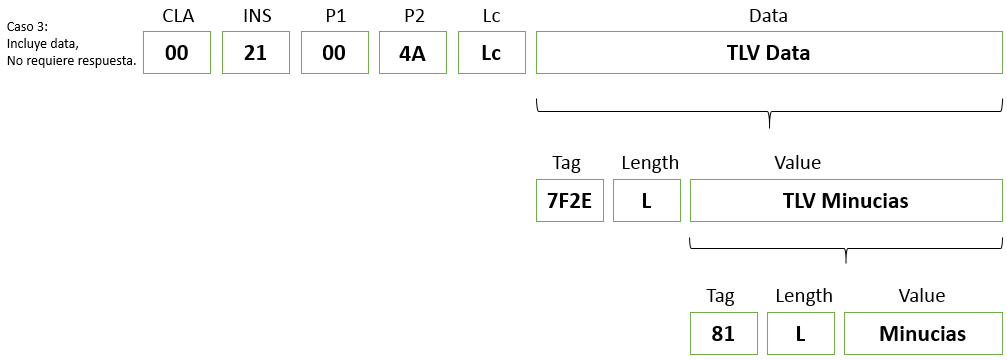
Diferentes servicios con APDU

Acceder a un ejemplo de uso con interfaz gráfica, lee los datos públicos de la CI y los muestra en pantalla.

Match on Card Cédula 2022 (cambios)

Validación de la huella digital de la persona Match On Card para las nuevas eID (Dual interface)

Estructura del comando APDU para la operación *Match On Card*.



- **CLA:**
  - 00h Transmisión en plano
  - 0Ch Transmisión sobre canal seguro
- **INS:**
  - 21h Fijo para la operación de *Match on Card*.
- **P1:**
  - 00h Fijo para la operación de *Match on Card*.
- **P2:**
  - 4Ah Fijo para la operación de *Match on Card*.
- **Lc:**
  - Largo en bytes del TLV para validación *Match on Card.*
- **Data:**
  - TLV para la validación *Match on Card* que contiene las minucias extraídas de una huella.

Ejemplo de un comando APDU para verificación *Match On Card*:

PACE - NFC Cédula 2022

PACE es una "súper aplicación" que controla (permite, deniega) la selección o acceso a las aplicaciones en función de la configuración estática de PACE (listas de restricciones) y el entorno de tiempo de ejecución.

Por lo tanto, dependiendo de la interfaz de comunicación actual utilizada (contacto/sin contacto) y la configuración de las listas de restricción de PACE, una aplicación determinada puede seleccionarse libremente, es decir SIN autenticación PACE, o su selección puede requerir tener una autenticación PACE.

Para las nuevas tarjetas MAV5.0 entregadas al Ministerio del Interior de Uruguay, este mecanismo será utilizado únicamente en la interfaz sin contacto y protegerá el acceso a la aplicación IAS. A través de la interfaz de contacto, se podrá acceder a todas las aplicaciones sin restricción de PACE.

Configuración de PACE en las aplicaciones

Interfaz sin contacto		
Aplicación	**Descripción**	**Listas de Restricción**
IAS Classic	Autenticación PACE es requerida para la selección/acceso a la instancia de IAS Classic. Nota: la selección debe ser realizada en claro (sin PACE secure messaging)	CL AL2
eTravel	Autenticación PACE no es requerida para la selección/acceso a la instancia de eTravel.	CL AL1
Card Manager	Autenticación PACE no es requerida para la selección/acceso a la instancia de Card manager.	CL AL1

Objeto de datos '01FC' en listas sin contacto

Las listas de restricciones y autorizaciones sin contacto se almacenan en un objeto de datos con la etiqueta '01FC' en el administrador de tarjetas que utiliza el siguiente formato:

Length of CFG and AID | CFG value | AID value | {L1 CFG1 AIDal1} … {Ln CFGn AIDaln}

Dónde:

• AID es el applet AID

• CFG se codifica como en la siguiente tabla:

CFG	Simple Coding On 1 Byte	Extended Coding On 2 Bytes
CL AL 1	xxxx xxx1	'80' xxxx xxx1
CL AL 2	xxxx xx1x	'80' xxxx xx1x
CL RL 3	xxxx x1xx	'80' xxxx x1xx
CL RL 4	xxxx 1xxx	'80' xxxx 1xxx
CL RL 5	xxx1 xxxx	'80' xxx1 xxxx
CL AL 6	xx1x xxxx	'80' xx1x xxxx

Etiqueta de codificación para '01FC':

01FC 7F	
0A 01 A0000002471001	→ eTravel
09 01 A000000151000000	→ Card Manager
0D 02 A00000001840000001634200	→ IAS

El objeto de datos se crea y actualiza a través del comando PUT DATA, descrito en el manual de referencia de MultiApp ID.

Este comportamiento de PACE cumple con los establecido en el documento ICAO TR-03110 v2.10-Parte 2.

Configuración PACE para tarjetas MAV5 (Interfaz Dual)

La nueva tarjeta MAV5.0 estará protegida mediante el algoritmo PACE**id-PACE-ECDH-GM-AES-CBC-CMAC-256** basado en una curva **elíptica NIST P-384 (secp384r1).**

Durante la personalización, los siguientes datos son personalizados utilizando comandos PUT DATA:

- Contraseñas (MRZ, PIN)
- Parámetros de información del dominio PACE
- Información PACE
- Configuración compatible con la llave de sesión (3DES, AES128, AES192, AES256)

PaceInfo y PaceDomainParameterInfo se ensamblan al final de la personalización de PACE para formar el archivo EF.CardAccess.

El esquema de autenticación PACE basado en el PIN “Contraseña” nunca transmite el PIN en sí mismo (ya sea en claro o encriptado). La información que se transmite es el resultado de una función unidireccional utilizando el PIN y números aleatorios. Por lo tanto, no es posible consultar una sesión de PACE basada en el PIN para recuperar el valor del PIN.

Personalización eléctrica de PACE

En el producto MAV5.0, PACE se personaliza a través de la aplicación Global Dispatcher Perso, con la APDU "PUT DATA", la APDU "END PERSO" debe ejecutarse al final dentro del contexto global (GApplet seleccionado).

Una vez que se ejecuta "END PERSO", se crea automáticamente "EF.cardAccess", en ese momento, el ciclo de vida de PACE se cambiará a**Personalizado**.

Aplicaciones utilizadas para la personalización de PACE:

Aplicación	Instancia AID
Global Dispatcher Perso	A000000018100203000000000000000
GApplet	A000000018100203000000000000001

Flujo de personalización

El flujo para personalizar PACE para cédulas de identidad electrónicas uruguayas es el siguiente:

1. Instanciar el subprograma Global Dispatcher Perso (GDP)

2. Instanciar GApplet

3. Escriba la lista de restricciones de PACE en el Administrador de tarjetas

4. Seleccione el contexto de la aplicación GApplet y autentíquese en GDP

5. Personalizar GApplet

6. Terminar la personalización: esto activará PACE.

La nueva cédula electrónica chip 2022 cuenta con capacidades de lectura segura por NFC utilizando el protocolo PACE para más información puede remitirse a los siguientes documentos:

- Technical Guideline TR-03110. Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token – Part 2: Protocols for electronic IDentification, Authentication and trust Services (eIDAS). Version 2.21. Date: 21. December 2016
- Technical Guideline BSI TR-03111. Elliptic Curve Cryptography. Version 2.10. Date: 2018-06-01
- Doc 9303 - Documentos de viaje de lectura mecánica. Parte 11: Mecanismos de seguridad para los MRTD