OpenSAMLPerl
Descripción
Y
Guía de Usuario

Daniel García Franco

17 de enero de 2006

Resumen

Este documento trata de describir en qué consiste el Binding Perl para [**?**] **OpenSAMLPerl** así como hacer las veces de guía de referencia para desarrolladores.

Índice

1. Introducción **1**

2. Instalación **2**

3. Estructura de OpenSAMLPerl **2**

4. Ejemplo de Uso: Prueba de Validación **4**
4.1. Bateria de Pruebas 4
4.2. Código de las Pruebas 4
4.2.1. Código Java 4
4.2.2. Código Perl 9
4.3. Ejecución de la Prueba 13
4.4. Resultado de Ejecución de la Prueba 13
Bibliografía Y Referencias17

1. Introducción

OpenSAMLPerl, es un Binding Perl para la implementación en Java de la librería *OpenSAML*, el cual permite hacer uso de dicha librería, pero desde Perl, es decir, permite instanciar objetos Java de OpenSAML desde un un script Perl, así como invocar todos los métodos públicos que dicho objeto Java posea.

Es importante resaltar, que los objectos creado o instanciados, no son objetos Perl, sino objetos Java que pueden ser manejados desde Perl, por medio de referencias a los objetos Java reales. dichas referencias si son 'objetos' o mejor dicho, entidades de Perl.

La posibilidad de instanciar objetos Java desde código Perl, se la debemos a *Patrick LeBoutiller*, quien ha desarrollado el módulo Perl [**?**], y en el cual nos hemos basado para realizar la implementación del Binding.

2. Instalación

Para instalar este software, se necesitan:

- Perl. version 5.8.6 o superior.
- Inline. versión 0.44 o superior.
- Inline::Java. version 0.49 o superior.

También es necesario tener un Java SDK cuya versión sea 1.22 o superior. La cual se puede obtener de Sun Microsystems en *http://java.sun.com*

Sigua los siguientes pasos para descomprimir, desempaquetar y configurar el Binding.

1. %> bzip2 -d openSAMLPerl-0.1.tar.bz2
2. %> tar -xfv openSAMLPerl-0.1.tar

Tras lo cual, se encontrará con la siguiente estructura de directorios:

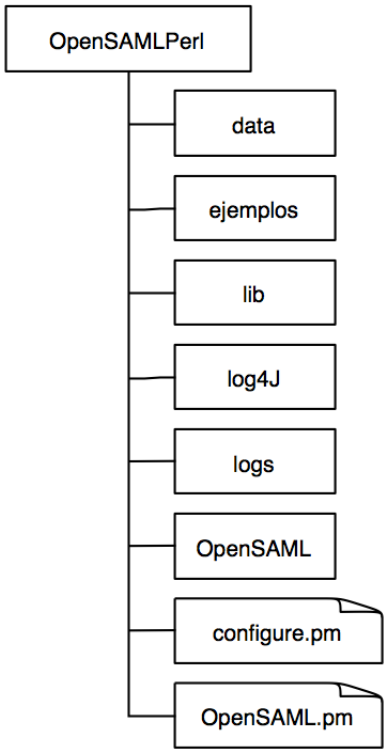

Figura 1: Estructura de directorios

Dentro del directorio *OpenSAMLPerl*, se encuentra un script, *configure.pl* que es el encargado de generar el fichero de *Global.pm*. Y para ello hace uso del path actual, es decir, del directorio donde se ha descomprimido y desempaquetado el Binding. Por tanto, es necesario, que antes de ejecutar el script *configure.pl* se mueva el directorio *OpenSAMLPerl* a su ubicación definitiva.

Para ejecutar *configure.pl*, pude hacerse de dos formas.

Haciendo uso del interprete Perl %> perl configure.pl

Dando permisos de ejecución %> chmod +x configure.pl && ./configure.pl

Tras la ejecución de *configure.pl* se ha de comprobar que el fichero *OpenSAML/Global.pm* se ha generado correctamente, para lo cual confirmaremos que a la variable `\$currentPath` se le ha asignado el path donde se ubica el Binding correctamente.

Una vez comprobado el fichero *OpenSAML/Global.pm*; podemos comprobar si todo funciona correctamente. Para realizar una comprobación del Binding, basta con crear un script Perl que haga uso del módulo *OpenSAML*, teniendo en cuenta, que si el Binding no se ha ubicado en alguno de los directorios en los que Perl busca por defecto, habrá que hacer uso del pragma `use lib "/path/ubicacion/Binding/"`. Para que Perl encuentre los módulos que forman el Binding.

3. Estructura de OpenSAMLPerl

Como se ha descrito en el punto anterior, el Binding OpenSAMLPerl está compuesto por seis directorios, un script de configuración, y por el módulo principal *OpenSAML*. A continuación describimos cual es el cometido y el contenido de los directorios.

data Este directorio, contiene los ficheros de configuración para [**?**][1], así como algunos esquemas xml de SAML.

ejemplos Este directorio contiene el fichero *SOAPBindingTest.pl*, el cual es una implementación en Perl del fichero Java SOAPBindingTest.java disponible con OpenSAML[2]

lib Este directorio es donde se encuentran todas las librerías Java que necesita OpenSAML así como el propio jar de OpenSAML.

Log4J Almacena las implementaciones de los módulos Perl que permiten crear y gestionar los objetos Log4J definidos en algunas clases de OpenSAML.

Log Directorio vacio donde se guardarán los log creados a partir de las configuraciones para *Log4J* disponibles en el directorio *data*.

OpenSAML Este directorio contine los módules Perl que se correcponden con las clases Java de OpenSAML.

[1] Quien este interesado en obtener más información a cerca de la configuración de OpenSAMLPerl para hacer uso de *Log4J*, así como detalles de la implementación, puede enviar un e-mail al autor de este documento, solicitandola

[2] En el siguiente punto se explicará con mayor detalle en que consiste el test, las modificaciones sufridas por eéste, y la codificación del mismo.

4. Ejemplo de Uso: Prueba de Validación

Para realizar las pruebas de validación del *binding*, y comprobar el correcto funcionamiento del mismo, se ha hecho uso de las pruebas desarrolladas en la versión 1.0 de OpenSAML. Dichas pruebas necesitan de un sistema Emisor/ Receptor de Autorización y Autenticación, ya que por si sólo, OpenSAML es simplemente una API que implementa la especificación SAML, la cual a su vez es un marco de trabajo para el intercambio de información segura. De forma que OpenSAML sólo se encarga de construir según el formato especificado por el Comité Técnico de Servicios de Seguridad [**?**] del consorcio [**?**], la información segura a intercambiar.

El sistema Emisor/Receptor de Autorización y Autenticación elegido para las pruebas del *binding* es [**?**], dicho sistema, posee una batería de pruebas basadas en las que posee OpenSAML en su versión 1.0. Estas pruebas tienen ligeras modificaciones sobre las de OpenSAML, pero aún así son validas para el propósito de testeo y validación del binding.

4.1. Bateria de Pruebas

La bateria de prueba original de AARR, modifica el fichero SOAPBindingTest de OpenSAML para que realice las siguientes peticiones[3]:

- Solicitud sobre el atributo eduPersonEntitlement que tiene el usuario Cándido Rodríguez del dominio us.es
- Solicitud de obtención de permiso para leer el recurso http://sugus.eii.us.es por el usuario anterior.
- Comprobación de autenticación de dicho usuario a través de una contraseña.

La bateria de pruebas del *binding*, consiste en reescribir el código Java del fichero SOAPBindingTest a Perl, de forma que donde se hacía uso de las primitivas Java de OpenSAML, ahora se hará uso de los paquetes Perl que se han desarrollado.

En el proceso de portar el código Java de SOAPBindingTest a Perl, se ha desechado la petición de comprobación de autenticación mediante el uso de contraseña, ya que supone el uso de varias clases de la API de Java, de forma directa, es decir, mediante la sintaxis de **Inline::Java**. Lo cual hace que el código Perl para la prueba sea menos legible.

En la prueba codificada en Java, se hace uso de un *framework* especifico para la realización de baterías de pruebas como es [**?**]. Aunque dicho *framework* también se encuentra disponible para Perl en forma de módulo cuyo nombre es [**?**], se ha decidido no hacer uso del mismo, con el objetivo de simplificar al máximo el código Perl resultante.

4.2. Código de las Pruebas

A continuación se muestra el código Java original de la prueba, desarrollado para AARR, y tras éste, el mismo código pero portado a Perl y con las modificaciones que ya han sido comentadas.

4.2.1. Código Java

```
package es.rediris.aarr.tests;

import junit.framework.TestCase;
```

[3]Para más información sobre la pruebas originales, dirigimos al lector al capítulo siete de la documentación de [AARR]

```
import org.apache.xml.security.signature.XMLSignature;

import java.io.FileInputStream;
import java.io.IOException;

import java.security.KeyStore;
import java.security.KeyStoreException;
import java.security.NoSuchAlgorithmException;
import java.security.UnrecoverableKeyException;

import java.util.Arrays;
import java.util.List;
import org.opensaml.QName;
import org.opensaml.SAMLAction;
import org.opensaml.SAMLAttributeDesignator;
import org.opensaml.SAMLAttributeQuery;
import org.opensaml.SAMLAuthenticationQuery;
import org.opensaml.SAMLAuthenticationStatement;
import org.opensaml.SAMLAuthorityBinding;
import org.opensaml.SAMLAuthorizationDecisionQuery;
import org.opensaml.SAMLConfig;
import org.opensaml.SAMLException;
import org.opensaml.SAMLNameIdentifier;
import org.opensaml.SAMLRequest;
import org.opensaml.SAMLResponse;
import org.opensaml.SAMLSOAPBinding;
import org.opensaml.SAMLSubject;
import org.opensaml.XML;


public class SOAPBindingTest extends TestCase {
  private String path = "data/org/opensaml/test.jks";
  private String alias = "mykey";
  private String password = "opensaml";
  private KeyStore ks = null;

  /**
   * Constructor for POSTProfileTest.
   * @param arg0
  */
  public SOAPBindingTest(String arg0) {
      super(arg0);
      System.out.println("Starting test");
  }

  public static void main(String[] args) {
```

```
  junit.textui.TestRunner.run(SOAPBindingTest.class);
}

/**
 * @see TestCase#setUp()
*/
protected void setUp() throws Exception {
    super.setUp();
    ks = KeyStore.getInstance("JKS");
  ks.load(new FileInputStream(path), password.toCharArray());

  SAMLConfig config = SAMLConfig.instance();
  config.setProperty("ssl-alias", alias);
  config.setProperty("ssl-keystore-pwd", password);
  config.setProperty("ssl-key-pwd", password);
}

/**
 * @see TestCase#tearDown()
*/
protected void tearDown() throws Exception {
    super.tearDown();
}

public void testSOAPBinding() throws Exception {
    SAMLAuthorityBinding binding = new SAMLAuthorityBinding(
        SAMLSOAPBinding.SAML_SOAP_HTTPS,
        "http://127.0.0.1:80",
        new QName(XML.SAML_NS, "AttributeStatement"));

  SAMLSOAPBinding b = new SAMLSOAPBinding();

  System.out.println("********* Attribute");
  SAMLRequest r = createAttributeQuery("Candido Rodriguez",
      "us.es",
      new String[] {"eduPersonEntitlement"}, false);
  getResponse(binding, b, r);
  System.out.println("--------------------------");

  System.out.println("********* AuthorizationDecision");
  SAMLRequest r4=createAuthorizationQuery(
      "Candido Rodriguez",
      "us.es",
      "http://sugus.eii.us.es",
      SAMLAction.SAML_ACTION_NAMESPACE_RWEDC_NEG,
      "Read",
      false);
```

```
    getResponse(binding,b,r4);
    System.out.println("--------------------------");

    System.out.println("********* Authentication");
    SAMLRequest r2=createAuthenticationQuery(
        "Candido Rodriguez",
        "us.es",
        SAMLAuthenticationStatement.AuthenticationMethod_Password,
        false);
    System.out.println("--------------------------");
    getResponse(binding,b,r2);
    System.out.println("--------------------------");
  }

  private void getResponse(
      SAMLAuthorityBinding binding,
      SAMLSOAPBinding b,
      SAMLRequest r) throws IOException, SAMLException {
    try {
      SAMLResponse r2 = b.send(binding, r);
      r2.toStream(System.err);
      System.err.println();
    }catch (SAMLException ex) {
      ex.printStackTrace();

      if ((ex.getException() != null) &&
          ex.getException() instanceof SAMLException) {
        ((SAMLException) ex.getException()).toStream(System.err);
      } else {
        ex.toStream(System.err);
      }

      throw ex;
    }
  }

  private SAMLRequest createAttributeQuery(String name,String domain,
      String[] attrs, boolean signed) throws SAMLException {
    List attrList = null;

    if (attrs != null) {
      SAMLAttributeDesignator[] attrArray =
          new SAMLAttributeDesignator[attrs.length];

      for (int i = 0; i < attrs.length; i++) {
        attrArray[i] = new SAMLAttributeDesignator(
            attrs[i], domain);
```

```
      }

      attrList = Arrays.asList(attrArray);
    }

    SAMLRequest r = new SAMLRequest(null,
        new SAMLAttributeQuery(
        new SAMLSubject(
        new SAMLNameIdentifier(name, domain, null),
        null, null, null),
        "http://www.foo.com/",
        attrList),
        null,
        null);

      if (signed) {
        try {
          r.sign(XMLSignature.ALGO_ID_SIGNATURE_RSA_SHA1,
              ks.getKey(alias, password.toCharArray()),
              Arrays.asList(ks.getCertificateChain(alias)));
        } catch (UnrecoverableKeyException e1) {
        } catch (KeyStoreException e2) {
        } catch (NoSuchAlgorithmException e3) {
        }
      }

      return r;
  }

  private SAMLRequest createAuthenticationQuery(String name,
      String domain,
      String authMethod,
      boolean signed) throws SAMLException {
    SAMLRequest r = new SAMLRequest(null,
        new SAMLAuthenticationQuery(new SAMLSubject(
        new SAMLNameIdentifier(name, domain, null), null, null,
        null), authMethod), null, null);

    if (signed) {
      try {
        r.sign(XMLSignature.ALGO_ID_SIGNATURE_RSA_SHA1,
            ks.getKey(alias, password.toCharArray()),
            Arrays.asList(ks.getCertificateChain(alias)));
        } catch (UnrecoverableKeyException e1) {
        } catch (KeyStoreException e2) {
        } catch (NoSuchAlgorithmException e3) {
      }
```

```
        }

        return r;
      }


      private SAMLRequest createAuthorizationQuery(String name,
          String domain, String uri, String actionUri,
          String actionData,
          boolean signed) throws SAMLException {
        SAMLAction action=new SAMLAction(actionUri,actionData);
        SAMLRequest r = new SAMLRequest(null,
            new SAMLAuthorizationDecisionQuery(
            new SAMLSubject(
            new SAMLNameIdentifier(name, domain, null),
            null, null, null),
            uri, Arrays.asList(new SAMLAction[]{action}), null),
            null,null);

          if (signed) {
            try {
              r.sign(XMLSignature.ALGO_ID_SIGNATURE_RSA_SHA1,
                  ks.getKey(alias, password.toCharArray()),
                  Arrays.asList(
                      ks.getCertificateChain(alias)));
              } catch (UnrecoverableKeyException e1) {
              } catch (KeyStoreException e2) {
              } catch (NoSuchAlgorithmException e3) {
            }
          }

          return r;
      }
    }
```

4.2.2. Código Perl

```
use OpenSAML;

use Inline Java      => 'STUDY',
           AUTOSTUDY => 1,
           STUDY     => ['java.util.ArrayList',
                                'java.lang.String',
                                'java.lang.System'];

use Inline::Java qw(caught);
```

```
 eval {&setUp();};
 if ($@) {
     if (caught("FileNotFoundException")){
         print "No se ha encontrado el fichero de configuración\n";
         print $@->getMessage()."\n";
     }else{
         die "Fallo algo y abortamos.\n Excepción en setUp()\n $@";
     }
 }

 eval { &testSOAPBinding(); };

 if ($@){
     print "testSOAPBinding ha fallado \n $@ \n";
     die;
 }



# - - - - -> Codificación de las Funciones <- - - - - #
sub setUp {
    my $alias = "mykey";
    my $password = "opensaml";
    my $config = new SAMLConfig();

    $config->setProperty("ssl-alias", $alias);
    $config->setProperty("ssl-keystore-pwd", $password);
    $config->setProperty("ssl-key-pwd", $password);
}

sub testSOAPBinding{
    my $qname = QName->new(XML::SAML_NS, "AttributeStatement");
    my $binding = SAMLAuthorityBinding->new(
      SAMLSOAPBinding::SAML_SOAP_HTTPS,
      "http://127.0.0.1",$qname);

    my $b = SAMLSOAPBinding->new();
    print "********* Attribute\n";
    my @array = ("eduPersonEntitlement");
    my $r = &createAttributeQuery("Candido Rodriguez", "us.es",
        @array, undef);
    &getResponse($binding, $b, $r);
    print "-------------------------\n";

    print "********* AuthorizationDecision\n";
    my $r4 = &createAuthorizationQuery(
        "Candido Rodriguez",
```

```
        "us.es",
        "http://sugus.eii.us.es",
        SAMLAction::SAML_ACTION_NAMESPACE_RWEDC_NEG,
        "Read", undef);
    &getResponse($binding, $b, $r4);
    print "-------------------------\n";

    print "********** Authentication\n";
    my $r2 = &createAuthenticationQuery("Candido Rodriguez", "us.es",
        SAMLAuthenticationStatement::AuthenticationMethod_Password,
        undef);
    print "-------------------------\n";
    &getResponse($binding, $b, $r2);
    print "-------------------------\n";
}

sub getResponse {
    my ($binding, $b, $r) = @_;
    my $r2;

    eval{
        $r2 = $b->send($binding, $r);
        print $r2->toString(). " \n";
    };

    if ($@){
        if(caught ('org.opensaml.SAMLException')){
            $@->printStackTrace();
            die "Fin de ejecución\n";
        }elsif (caught ('org.opensaml.MalformedException')){
            print "MalformedException_Mensaje".$@->getMessage()."\n";
        }else{
            print "Mensaje: " . $@->getMessage() . "\n";
            croak "No fue una excepción java:\n $@ \n";
        }
    }
    $@ =  undef;
}



# La función recibe 4 parámetros de entrada:
#  Scalar name:
#  Scalar domain:
#  Array[] Scalar attrs:
#  Scalar (boolean) signed:
```

```
sub createAttributeQuery {
    my ($name, $domain, @attrs, $signed) =@_;
    my $attrList = undef;
    my $samlRequest;
    my $samlAttributeQuery;
    my $samlSubject;
    my $samlNameIdentifier;

    if ( @attrs ){
        $attrList = new java::util::ArrayList();
        foreach my $aName (@attrs){
            my $aux;
            if ($aName){
                $aux = SAMLAttributeDesignator->new(
                    $aName, $domain);
                $attrList->add($aux);
            }
        }
    }

    $samlNameIdentifier = SAMLNameIdentifier->new(
        $name, $domain, undef);

    $samlSubject = SAMLSubject->new(
        $samlNameIdentifier, undef, undef, undef);

    $samlAttributeQuery = SAMLAttributeQuery->new(
        $samlSubject, "http://www.foo.com/",$attrList);

    $samlRequest = SAMLRequest->new(
        undef, $samlAttributeQuery, undef, undef);

    return $samlRequest;
}

sub createAuthenticationQuery{
    my ($name, $domain, $authMethod, $signed) = @_;
    my $request;
    my $nameIdentifier;
    my $subject;
    my $authenticationQuery;

    $nameIdentifier = SAMLNameIdentifier->new(
        $name, $domain, undef);

    $subject = SAMLSubject->new(
        $nameIdentifier, undef, undef, undef);
```

```
    $authenticationQuery = SAMLAuthenticationQuery->new(
        $subject, $authMethod);

    $request = SAMLRequest->new(
        undef, $authenticationQuery, undef, undef);

    return $request;
}

sub createAuthorizationQuery {
    my ($name,$domain,$uri,$actionUri,$actionData,$signed) = @_;
    my $action = SAMLAction->new($actionUri,$actionData);
    my $nameIdentifier = SAMLNameIdentifier->new($name,$domain,undef);
    my $subject = SAMLSubject->new($nameIdentifier,undef,undef,undef);
    my $arrayList = java::util::ArrayList->new();

    $arrayList->add($action);
    my $authorizationDecisionQuery=SAMLAuthorizationDecisionQuery->new(
        $subject, $uri, $arrayList, undef);

    my $request = SAMLRequest->new(
    undef, $authorizationDecisionQuery,undef,undef);

return $request;
}
```

4.3. Ejecución de la Prueba

Para realizar la ejecución de la prueba, se ha de lanzar el sistema AARR, el cual se queda a la escucha de peticiones. Una vez que está ejecutandose AARR, se ejecuta el 'script' Perl SOAPBindingTest.pl, el cual, haciendo uso de la API OpenSAMLPerl, la cual es el *binding* en si, realiza la conexión contra el sistema AARR y hace el envío de las peticiones que se describieron en el apartado **??** Batería de Pruebas.

Siendo el resultado obtenido, el que se muestra en el siguiente apartado.

4.4. Resultado de Ejecución de la Prueba

```
********* Attribute
<Response xmlns="urn:oasis:names:tc:SAML:1.0:protocol"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
InResponseTo="b0bdceb0e6ff3daa1fec2647b502fee7"
IssueInstant="2005-08-25T09:46:49.805Z"
MajorVersion="1"
MinorVersion="1"
ResponseID="d8ffb952ae4139db9886bb745b07e628">
    <Status>
```

```
        <StatusCode Value="samlp:Success">
        </StatusCode>
    </Status>
    <Assertion xmlns="urn:oasis:names:tc:SAML:1.0:assertion"
    AssertionID="ffc0325e79a1b6eddbee99162868179b"
    IssueInstant="2005-08-25T09:46:49.804Z"
    Issuer="My World"
    MajorVersion="1"
    MinorVersion="1">
        <Conditions NotBefore="2005-08-25T09:46:49.804Z">
        </Conditions>
        <AttributeStatement>
            <Subject>
                <NameIdentifier NameQualifier="us.es">
                    Candido Rodriguez
                </NameIdentifier>
            </Subject>
            <Attribute
             AttributeName="eduPersonEntitlement"
             AttributeNamespace="us.es">
                <AttributeValue>
                    urn:mace:rediris.es:entitlementForPAPIResources
                </AttributeValue>
            </Attribute>
        </AttributeStatement>
    </Assertion>
</Response>
-------------------------

********* AuthorizationDecision
<Response
 xmlns="urn:oasis:names:tc:SAML:1.0:protocol"
 xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
 xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
 InResponseTo="a0ec725e6ebfca37bd2e324a758ec5ec"
 IssueInstant="2005-08-25T09:46:51.127Z"
 MajorVersion="1"
 MinorVersion="1"
 ResponseID="a5eb4b08e929fbec786e6d12506503eb">
    <Status>
        <StatusCode Value="samlp:Success">
        </StatusCode>
    </Status>
    <Assertion
     xmlns="urn:oasis:names:tc:SAML:1.0:assertion"
     AssertionID="e970d22900c96f8534187535380c2154"
     IssueInstant="2005-08-25T09:46:51.127Z"
```

```
     Issuer="My World"
     MajorVersion="1"
     MinorVersion="1">
        <Conditions NotBefore="2005-08-25T09:46:51.127Z">
        </Conditions>
        <AuthorizationDecisionStatement
         Decision="Permit"
         Resource="http://sugus.eii.us.es">
            <Subject>
                <NameIdentifier NameQualifier="us.es">
                    Candido Rodriguez
                </NameIdentifier>
            </Subject>
            <Action
             Namespace="SAMLAction::SAML_ACTION_NAMESPACE_RWEDC_NEG">
             Read
            </Action>
        </AuthorizationDecisionStatement>
    </Assertion>
</Response>
-------------------------

********** Authentication
<Response
 xmlns="urn:oasis:names:tc:SAML:1.0:protocol"
 xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
 xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
 InResponseTo="d61b24349b553b3ae72095e4bc91e3ea"
 IssueInstant="2005-08-25T09:46:51.592Z"
 MajorVersion="1"
 MinorVersion="1"
 ResponseID="c33b4d4814c8d322fa9b3d007d42a322">
    <Status>
        <StatusCode Value="samlp:Success">
        </StatusCode>
    </Status>
    <Assertion
     xmlns="urn:oasis:names:tc:SAML:1.0:assertion"
     AssertionID="c23149893bf03a3f4075e8087a4c993c"
     IssueInstant="2005-08-25T09:46:51.592Z"
     Issuer="My World"
     MajorVersion="1"
     MinorVersion="1">
        <Conditions NotBefore="2005-08-25T09:46:51.588Z">
        </Conditions>
        <AuthenticationStatement
         AuthenticationInstant="1970-01-13T13:20:05.892Z"
```

```
         AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
            <Subject>
                <NameIdentifier NameQualifier="us.es">
                    Candido Rodriguez
                </NameIdentifier>
            </Subject>
            <SubjectLocality
             DNSAddress="150.214.142.245"
             IPAddress="150.214.142.245">
            </SubjectLocality>
        </AuthenticationStatement>
    </Assertion>
</Response>
-------------------------
```

Referencias

[1] Cándido Rodríguez Montes
Proyecto Fin de Carrera, Diseño de un AA-RR
http://www.rediris.es/app/aarr/

[2] INLINE::JAVA - WRITE PERL CLASSES IN JAVA.
http://search.cpan.org/ patl/Inline-Java-0.50/Java.pod

[3] JUNIT, TESTING RESOURCES FOR EXTREME PROGRAMING
http://www.junit.org/

[4] LOG4J PROJECT
http://logging.apache.org/log4j/docs/

[5] SECURITY SERVICES TECHNICAL COMMITTEE, SAML 1.0 SPECIFICATION SERVICE
http://www.oasis-open.org/committees/security/

[6] AN OPEN SOURCE SECURITY ASSERTION LANGUAJE IMPLEMENTACION
http://www.opensaml.org/

[7] OASIS SECURITY SERVICES (SAML) TC
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[8] SECURE SERVICE TECHNICAL COMMITTEE
www.oasis-open.org/commitees/security

[9] TEST::UNIT - THE PERLUNIT TESTING FRAMEWORK
http://search.cpan.org/ aspiers/Test-Unit-0.24/lib/Test/Unit.pm