OpenSAMLPerl
Description
&
User Guide

Daniel García Franco

January 20, 2006

Abstract

The aim of this document is describe the Binding Perl for **OpenSAML** [1] how it works and how it have been made. As well as supply a user guide for the people who want use the OpenSAML privitves from Perl.

Contents

1 Introduction

OpenSAMLPerl, it's a Binding Perl for the Java implementation of *OpenSAML* librabry, that makes possible to use that library from Perl, in others words, OpenSAMLPerl makes possible to create Java objects from inline Perl code, and invoke all the public methods methods of these Java objects.

It's important emphasize that the created objets aren't Perl objects, but Java objects that can be handled from inline Perl code using references that points to the real Java objects. The references that handle the Java objects belong to Perl, they are perl variables that point to the Java objects.

The Java objects creation is possible thanks to *Patrick LeBoutiller*, who has developed the Perl module **Inline::Java** [2]. That is the modele we have based the Binding implementation.

2 Install

To install the software is needed:

- Perl. version 5.8.6 or above.
- Inline. version 0.44 or above.
- Inline::Java. version 0.49 or above.

Also is needed Java SDK 1.22 or avobe. That can be downloaded from Sun Microsystems at *http://java.sun.com*
To uncompress, unpack and configure the Binding, follow those steps:

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

After that you can see that directory structure:
Inside the *OpenSAMLPerl* directory, there is the *configure.pl* script, that is used to make the *Global.pm* file. This script uses the current working directory, so we propose that before you execute the script, you move the directory *OpenSAMLPerl* to the final location.

The *configure.pl* script, can be execute in two ways.

By the Perl interpreter %> perl configure.pl

Granting execution mode %> chmod +x configure.pl && ./configure.pl

Once time the script has been executed, you should check that the *OpenSAML/Global.pm* file was well created. To do this you should check that the value of the variable `\$currentPath` is the path where the Binding is located.

Now you can check if all work well. You can make a Perl script that uses the *OpenSAML* Perl module. You should have in mind if you have located the Binding in a directory that Perl doesn't looked for the modules, you must use the Perl pragma `use lib "/path/to//BindingLocation/"` . To add the Binding's directory to the perl search path.

3 Strucuture of OpenSAMLPerl

As have been show in the above section, the Binding is compossed by six directories, a script to configure it and by the module *OpenSAML* (the OpenSAML.pm file), that is the most imporntant. Follow will be describe which is the task and what contain each one.

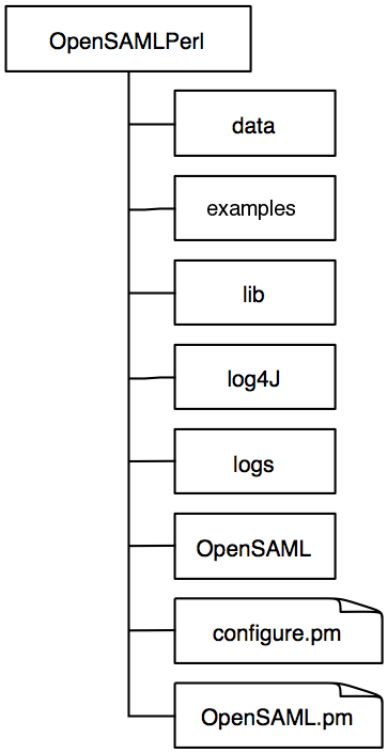

Figure 1: Directories Structure

data This directory contains the configuratin files for **Log4J** [3][1], and some SAML xml schemes.

examples This directory caontains the *SOAPBindingTest.pl* file, which is a Perl implementation of SOAPBindingTest.java, that it's available with OpenSAML.[2]

lib In this directory are stored all Java libraries that OpenSAML need, and also the OpenSAML jar.

Log4J In This directory are stored the Perl modules that implement Log4J, and that allow the creation and the managemenet the Log4J objects used in some OpenSAML classes.

Log This directory is empty, and it will store the different log produced by *Log4J*.

OpenSAML This directory contains the Perl Module that correspond with the OpenSAML Java classes.

[1] Who would like to obtain more information about the OpenSAMLPerl's configuration for *Log4J*, as so as to get more details about the implementation of this, can send an e-mail to author of this document.

[2] In the next section will be given details about the test, the modifications have been made and the changes added to this one.

4 OpenSAMLPerl Binding Documentation

The documentation for use the Binding, is the same tha the Javadoc of OpenSAML API, that one can be downloaded from the OpenSAML web site. But the Binding is based in **Inline::Java** Perl module, which make the interaction with Java.
You have in mine the types conversion when you pass Perl parameters to Java object methods. Although **Inline::Java** do the conversion by you, the arrays are deal in a differente way. You can pass any kinf od Perl scalar or any Java object to a method. The **Inline::Java** module will be automatically converted to the correct type:

```
use Inline Java => <<'END' ;
  class Pod_3_arg {
    public Pod_3_arg(){
    }
  }
  class Pod_3 {
    public int n ;

    public Pod_3(int i, String j, Pod_3_arg k) {
      n = i ;
    }
  }
  END

  my $obj = new Pod_3_arg() ;
  my $obj2 = new Pod_3(5, "toto", $obj) ;
  print($obj2->{n} . "\n") ; # prints 5
```

will work fine. These objects can be of any type, even if these types are not known to **Inline::Java**. This is also true for return types:

```
use Inline Java => <<'END' ;
  import java.util.* ;

  class Pod_4 {
    public Pod_4(){
    }

    public HashMap get_hash(){
      HashMap h = new HashMap() ;
      h.put("key", "value") ;

      return h ;
    }
```

```
    public String do_stuff_to_hash(HashMap h){
      return (String)h.get("key") ;
      }
    }
  END

  my $obj = new Pod_4() ;
  my $h = $obj->get_hash() ;
  print($obj->do_stuff_to_hash($h) . "\n") ; # prints value
```

Objects of types unknown to Perl can exist in the Perl space, you just can't call any of their methods. But it's possible to STUDY a class, so you can "teach"**Inline::Java** about that class, and so you can call the public methods o the class. You have to use the AUTOSTUDY option to tell **Inline::Java** that you wish to study all classes that it comes across:

```
use Inline Java => <<'END', AUTOSTUDY => 1 ;
import java.util.* ;

    class Pod_10 {
      public Pod_10(){
      }

      public HashMap get_hm(){
        HashMap hm = new HashMap() ;
        return hm ;
      }
    }
    END

   my $obj = new Pod_10() ;
   my $hm = $obj->get_hm() ;
   $hm->put("key", "value") ;
   my $val = $hm->get("key") ;
   print($val . "\n") ; # prints value
```

In this case **Inline::Java** intercepts the return value of the get_hm() method, sees that it's of a type that it doesn't know about (java . lang . HashMap), and immediately studies the class. After that call the java :: lang :: HashMap class is available to use through Perl.

You can also send, receive and modify arrays. This is done simply by using Perl lists:

```
use Inline Java => <<'END' ;
  import java.util.* ;
```

```
class Pod_6 {
  public int i[] = {5, 6, 7} ;

  public Pod_6(){
  }

  public String [] f(String a[]){
    return a ;
  }

  public String [][] f(String a[][]){
    return a ;
  }
}
END

my $obj = new Pod_6() ;
my $i_2 = $obj->{i}->[2] ; # 7
print($i_2 . "\n") ; # prints 7

my $a1 = $obj->f(["a", "b", "c"]) ; # String []
my $a2 = $obj->f([
                          ["00", "01"],
                          ["10", "11"],
                                          ]) ; # String [][]
print($a2->[1]->[0] . "\n") ; # prints 10
```

5 Examples of Use: Validation Tests

The test of OpenSAML version 1.0 have been used to do the validation test for the Binding, and to check that all work rigth. As OpenSAML is only an API that implements the SAML standar, to do the test is needed a system that interprets the SAML "message" that will be issued by the OpenSAMLPerl Binding. Therefore the test will need a Authorization and Authentication Request/Response system

The Authorization and Authentication Request/Response system choosed to do the Binding's test is **AARR** [4], that system, have a test pool based in the test pool proposed by OpenSAML in its version 1.0 The test pool of AARR have few chages from OpenSAML once.

5.1 Test Pool

The original test pool of AARR changes the SOPABindingTest of OpenSAML for the next requests[3]:

- Is requested the `eduPersonEntitlement` attribute that has the user `Cándido Rodríguez` from `us.es` domain.
- Request the permission to read the http://sugus.eii.us.es resource by the above user.
- Check the authentication of the above user by a password.

The test pool for the OpenSAMLPerl Binding lies in rewrite the Java code of SOAPBindingTest as Perl code, but now using the Perl modules of the Binding to create the Java object, instead of Java primitives.

5.2 Test Pool Code

The code of the original Java test pool is shown in this section, this code is from AARR. Also the Perl code make to the test pool is shown in the next section.

5.2.1 Java Code

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

import junit.framework.TestCase;

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;
```

[3]To get more information about the original pool test, see the seventh chapter of the AARR's documentation

```
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;
      }
    }
```

5.2.2 Perl Code

```
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;
 }



# - - - - -> Functions Code <- - - - - #
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;
}


# The function has four input parameters:
#  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;
}
```

5.3 Test Pool Execution

For the execution of the test pool, the AARR system must be running, when AARR is listening for requests, the SOAPBindingTest.pl test script must be executed.

The obtained results for the test pool are shown in the next section.

5.4 Obtained Results

```
********* 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>
-------------------------
```

References

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

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

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

[4] Cándido Rodríguez Montes, Diego R. López
AA-RR Design
http://www.rediris.es/app/aarr/AARR-design-v1.1.pdf

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

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