

LDAPPortalAuthentication+and+RequestUserInfoFromXSP

If you have a default Portal with an XML-file authentication and you want to change it to LDAP authentication read this. I also explain how it is possible to request the ID, user and role from an XSP page.

There are certainly other ways to make it, but here I explain an simple an easy way (you won't have to decrypt password or anything like that).

To authenticate I use this manner :

I use the parameters (user and password from the login form) to connect to the ldap server. If the login succeed that means that the password is correct for the username. So then I just get the user information and set them!

Part1 : authentication with LDAP

Sitemap configuration :

```
...
<map:pipeline internal-only="true">
  <!-- This is the Cocoon Demo Portal Pipeline -->
  <map:match pattern="sunrise-*">
    <!-- authentication -->
    <map:match pattern="sunrise-authuser">
      <map:generate src="ldap.xml"/>
      <map:transform type="ldap">
        <map:parameter name="rootdn" value="{request-param:name}"/>
        <map:parameter name="password" value="{request-param:password}"/>
      </map:transform>
      <map:transform type="xslt" src="ldap.xsl"/>
      <map:transform src="styles/sunrise-user.xsl">
        <map:parameter name="use-request-parameters" value="true"/>
      </map:transform>
      <map:serialize type="xml"/>
    </map:match>

    <map:match pattern="sunrise-changeuser">
      ...
    </map:match>
  </map:match>
</map:pipeline>
...
```

LDAP.XML page :

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<authentication xmlns:ldap="http://apache.org/cocoon/LDAP/1.0">
  <ldap:execute-query>
    <ldap:initializer>com.sun.jndi.ldap.LdapCtxFactory</ldap:initializer>
  <!-- enter your own ldap server -->
    <ldap:serverurl>ldap://CORPROOT.NET</ldap:serverurl>
    <ldap:authentication>simple</ldap:authentication>
    <ldap:version>3</ldap:version>
    <ldap:port>389</ldap:port>
    <ldap:scope>SUBTREE_SCOPE</ldap:scope>
  <!-- enter your own searchbase & filter -->
    <ldap:searchbase>DC=corproot,DC=net</ldap:searchbase>
    <ldap:filter>(&!(objectclass=User)(cn=TFR*))</ldap:filter>
    <ldap:deref-link>TRUE</ldap:deref-link>
    <ldap:count-limit>0</ldap:count-limit>
    <ldap:time-limit>0</ldap:time-limit>
    <ldap:show-attribute>TRUE</ldap:show-attribute>
    <ldap:doc-element>users</ldap:doc-element>
    <ldap:row-element>user</ldap:row-element>
    <ldap:error-element>ELEMENT</ldap:error-element>
  <!-- enter your own attributes -->
    <ldap:attribute>cn</ldap:attribute>
    <ldap:attribute>co</ldap:attribute>
    <ldap:attribute>sn</ldap:attribute>
    <ldap:attribute>employeeID</ldap:attribute>
    <ldap:attribute>title</ldap:attribute>
    <ldap:attribute>company</ldap:attribute>
    <ldap:attribute>givenname</ldap:attribute>
    <ldap:attribute>mail</ldap:attribute>
    <ldap:attribute>streetAddress</ldap:attribute>
    <ldap:attribute>postalCode</ldap:attribute>
    <ldap:attribute>st</ldap:attribute>
    <ldap:attribute>mobile</ldap:attribute>
  </ldap:execute-query>
</authentication>

```

Now we need to have the same tags than in the demo file : sunrise-user.xml so I pass my xml trough an xsl :

LDAP.XSL file :

```

<?xml version="1.0"?>

<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="authentication">
    <authentication>
        <xsl:apply-templates select="users"/>
    </authentication>
</xsl:template>

<xsl:template match="users">
    <users>
        <xsl:apply-templates select="user"/>
    </users>
</xsl:template>

<xsl:template match="user">
    <user>
        <name><xsl:value-of select="cn"/></name>
        <password>testa</password>
        <role>admin</role>
        <title><xsl:value-of select="title"/></title>
        <firstname><xsl:value-of select="givenName"/></firstname>
        <lastname><xsl:value-of select="sn"/></lastname>
        <company><xsl:value-of select="company"/></company>
        <street><xsl:value-of select="streetAddress"/></street>
        <zipcode><xsl:value-of select="postalCode"/></zipcode>
        <city><xsl:value-of select="st"/></city>
        <country><xsl:value-of select="co"/></country>
        <phone><xsl:value-of select="mobile"/></phone>
        <fax></fax>
        <email><xsl:value-of select="mail"/></email>
        <bankid></bankid>
        <bankname></bankname>
        <accountid></accountid>
    </user>
</xsl:template>

</xsl:stylesheet>

```

Now that the xml result (to see the result you can add a logger in the pipeline just after the ldap.xml :

```

<map:transform type="log">
    <map:parameter name="logfile" value="UserList.log"/>
    <map:parameter name="append" value="NO"/>
</map:transform>

```

Now we apply the demo xsl (sunrise-user.xml) and then we serialize the result in xml.

That's it for the authentication.

Part2 : Getting user info from XSP

Since it's not possible to use <xsp-session:getxml> I had to find an alternative.

So I use <xsp-session:get-attribute-names/> to get all session parameters, then I make a java function to parse it. There certainly must be easier ways but this way works. Well, you first need to add jakarta ORO libs (they are very usefull for any parsing, char replacement,... and perl functions). So that's what it looks like :

LOGIN.XSP file :

```

<xsp:page language="java"
    xmlns:xsp="http://apache.org/xsp"
    xmlns:xsp-session="http://apache.org/xsp/session/2.0"
    xmlns:esql="http://apache.org/cocoon/SQL/v2"

```

```

xmlns:xsp-request="http://apache.org/xsp/request/2.0"
>

<xsp:structure>
  <xsp:include>org.apache.cocoon.environment.Session</xsp:include>
  <xsp:include>org.apache.avalon.framework.component.ComponentManager</xsp:include>
  <xsp:include>org.apache.oro.text.regex.*</xsp:include>
  <xsp:include>java.util.ArrayList</xsp:include>
  <xsp:include>java.util.Collection</xsp:include>
</xsp:structure>

<page>

<xsp:logic>
  Object[] myarray = null;
  java.util.Collection list = new ArrayList();
  PatternMatcher matcher = new Perl5Matcher();
  Pattern pattern = null;
  PatternCompiler compiler = new Perl5Compiler();
  String regularExpression, input, firstsplit, secondsplit;
  String myvara;
  Pattern searcher;
  PatternMatcherInput myvar;
  String result = "uprofile|test:Guest_0_guest";
  <!-- default value, so that if the user is not logged the name
        and role are set to guest -->

  myarray = <xsp-session:get-attribute-names as="array"/>.toArray();

  try {
    searcher = compiler.compile("uprofile");
  } catch(MalformedPatternException e) {
    System.out.println("Bad pattern.");
    System.out.println(e.getMessage());
    return;
  }

  <!-- Search the user information, the string is something like
        profile:something|somethingele:User_13_Michael -->
  for (int i= 0; i <= 5; i++) {
    myvara = myarray[i].toString();
    myvar = new PatternMatcherInput(myvara);
    while(matcher.contains(myvar, searcher)) {
      result = myarray[i].toString();
      i = 5;
    }
  }

  regularExpression = "\\|";

  try {
    pattern = compiler.compile(regularExpression);
  } catch(MalformedPatternException e){
    System.err.println("Bad pattern.");
    System.err.println(e.getMessage());
    System.exit(1);
  }

  Util.split(list,matcher,pattern,result);

  regularExpression = ":";
  try {
    pattern = compiler.compile(regularExpression);
  } catch(MalformedPatternException e){
    System.err.println("Bad pattern.");
    System.err.println(e.getMessage());
    System.exit(1);
  }

  myarray = list.toArray();

```

```

        firstsplit = myarray[1].toString();

        list.clear();
        Util.split(list,matcher,pattern,firstsplit);

        myarray = list.toArray();
        firstsplit = myarray[1].toString();

        regularExpression = "_";
        try {
            pattern = compiler.compile(regularExpression);
        } catch (MalformedPatternException e){
            System.err.println("Bad pattern.");
            System.err.println(e.getMessage());
            System.exit(1);
        }

        list.clear();
        Util.split(list,matcher,pattern,firstsplit);

        myarray = list.toArray();

        firstsplit = myarray[2].toString();
        secondsplit = myarray[3].toString();
    </xsp:logic>

    <a><xsp:attribute name="href">home</xsp:attribute><loadingimg/></a>
    <xsp:logic>
        String sector = null;
    </xsp:logic>
    <esql:connection>
        <esql:pool>wfmg</esql:pool>
        <esql:execute-query>
            <esql:query>
                SELECT ur_sector
                FROM Users
                WHERE ur_name = '<xsp:expr>secondsplit</xsp:expr>'
            </esql:query>
            <esql:results>
                <esql:row-results>
                    <xsp:logic>
                        sector = <esql:get-string column="ur_sector"/>;
                        session.setAttribute("sector",sector);
                    </xsp:logic>
                </esql:row-results>
            </esql:results>
        </esql:execute-query>
    </esql:connection>
    <xsp:logic>
        session.setAttribute("user",<xsp:expr>secondsplit</xsp:expr>);
        session.setAttribute("role",<xsp:expr>firstsplit</xsp:expr>);
    </xsp:logic>
</body>
</html>
</xsp:page>

```

– [MaximeGheysen](#)

Attachment: [ldap.xsl](#)

Attachment: [login.xsp](#)

Attachment: [ldap.xml](#)