Journal of Mobile, Embedded and Distributed Systems, vol. II, no. 2, 2010
ISSN 2067 – 4074

Implementing graphic passwords in Directory Services Systems

Emanuil REDNIC, Andrei TOMA

Economical Informatics Department, Academy of Economic Studies
Piata Romana, sector 1, Bucharest, ROMANIA
emanuil.rednic@oracle.com, andrei.toma@ie.ase.ro

Abstract: This paper presents the necessary steps for the implementation of multimedia password support within Directory Services. The trend of using multimedia mechanisms is quite new and furthermore relatively unused in authentication as well authorization. Implementing this type of password is justified by the increased level of security within the identity management of directory services.

Key-Words: database security, distributed activities, multimedia information, multimedia processing, watermarking.

1 Introduction

The study is based on our previously published paper, *Multimedia Flavor in Directory Services Environment* (*Informatica Economica Journa*l, nr. 13/2009).[1][2] There we presented the bases of how multimedia data is designed, stored, altered and manipulated in the directory services. The enhancement to the basic functionalities of LDAP was done by implementing this multimedia facet. As LDAP attributes and classes were first designed to cover only textual attributes, the step of introducing multimedia features contributed to the flexibility of these LDAP systems, allowing the management of any type of data, from numerical and alpha numerical to the binary ones.

2 Problem formulation

The problem at hand is implementing a graphical password which would be used for the authentication level as well as for the authorization level.

This is a post conference paper. Parts of this paper have been published in the Proceedings of the 2nd International Conference on Security for Information Technology and Communications, SECITC 2009 Conference (printed version).

2.1 Reasons for implementation

The need to increase the security level of multimedia systems led us to consider the implementation of a graphical password within the system itself.

Utilizing a binary password leads to a greater degree of flexibility for such a system. The added advantage would stem from LDAP integration which would allow leveraging the power of LDAP systems in what in what identity management is concerned in order to build more secure multimedia systems.

An added advantage stemming from LDAP integration is the increase of the security of distributed enterprise applications.

2.2 Prior research

Within the last years more and more LDAP systems have increased their level of integration of multimedia features.

This has in turn led to the easier integration of LDAP systems themselves, mainly through the use of common classes and attributes, which include the ones involved in multimedia operations.

Some steps in the direction of using images as authentication data have been made with some success acquiring the data through a mobile phone camera with the objective of providing secure location independent proof of identity[8]. However, such approaches did not attempt to

integrate the mechanism with an identity management system.
Within identity management systems, multimedia mechanisms were used only for user profile management (such as in HR applications). Even so, the images used were not involved in any kind of manipulation, such as altering or comparison. They were only used for personnel reports and were thus intended to be "processed" directly by the human user.

3 Problem Solution

The present section deals with the implementation of graphical passwords within a LDAP system.[3][4]
There are a number of prerequisites that must be in place for the implementation to be possible: a LDAP system which has support for multimedia data and multimedia operations as well as work with plug-ins (in our case, java plug-ins). Figure 1 shows a diagram of the plug-in process.

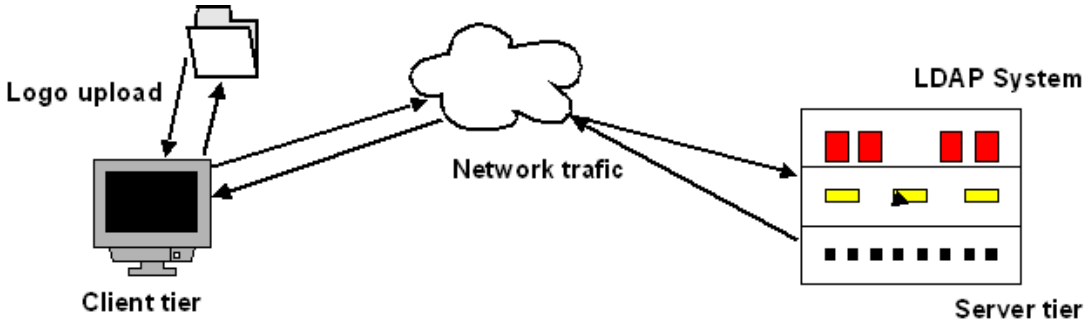

Figure 1. Network traffic for pre-login plugin

One of the most known LDAP's is OID (Oracle Internet Directory) which fulfills all the above conditions.

The first step is to design the plug-in which will be used at the authentication and/or authorization phase.

OID provides flexibility as to what approach the user takes when developing plug-ins.

One can elect to use either PL-SQL plug-ins (for the users which are used with Oracle's proprietary programming paradigm) or an oriented programming language such as Java since OID supports such plug-ins developed in a different language than PL-SQL.

An OID plug-in can be developed according to several profiles which specify when the plug-in fires. Such profiles include pre-[operation], post-[operation], when-[operation] and when-replaced-[operation] where [operation] can be any operation supported by LDAP.[5][6][7].

The particular plug-in we are discussing is a pre-login plug-in which will be fired before the login operation is performed.

In this phase, the user is required to enter a logo (we refer to the multimedia resource which was stored in the database as logo), at the user creation phase. Each user has their specific logo, which was provided manually by the LDAP administrator or automatically by the LDAP system. The logo will be assigned by a mechanism which guarantees the logo's uniqueness within the LDAP system.

Since every user has a unique logo, each user is required to upload it at the login phase.

ISSN 2067 – 4074

Storing the logo in a buffer variable will make the comparison operation between the data stored In the LDAP system and the volatile data stored in the buffer variable much easier. Once the compare is successful, the user can easily log into the system.

This above is the graceful scenario which is the sequence of events which allows the user access into the system. The ungraceful scenario is a failed login. This event will happen as a consequence of a failed comparison either due to pixel corruption within the local version of the logo or alteration of the logo by the user itself.

Alteration of the logo, be it on the side of the client or of the server, can be avoided by setting security policies. On the server side things are much simpler due to the already existent security rules for data which also apply to multimedia data. On the client side, the possibility of alteration depends on the end user's personal security policies.

The pre-login plug-in will perform a comparison operation by reading bit by bit the multimedia information stored as volatile information. The success of comparison operation is equivalent to the success of the authentication and/or authorization operation. [12][13]The process is shown in Figure 2 below.

The level of security can be increase by setting the LDAP system, in this case OID, to work in SSL mode, which will have the consequence that all LDAP operations will be on SSL mode. If somebody attempts to capture the LDAP traffic, for example by using a sniffer tool to capture network traffic, they will not be able to obtain the password itself either by performing the operation over SSL and/or text encoding the binary data.

Troubleshooting[9][10][11]the plug-in becomes much easier with the help of the LDAP system itself. OID gives the possibility to trace/debug plug-ins, no matter if they are written in PL-SQL or Java. The limits of the debugging capability depend on the level of debugging set for OID.

From OID version 10.1.4.x and later the direction in the plug-in development is to use the java programming language as the main programming language for the development phase. Of course, plug-ins can still be in the PL-SQL programming language in order to assure the backward compatibility with the previous systems.

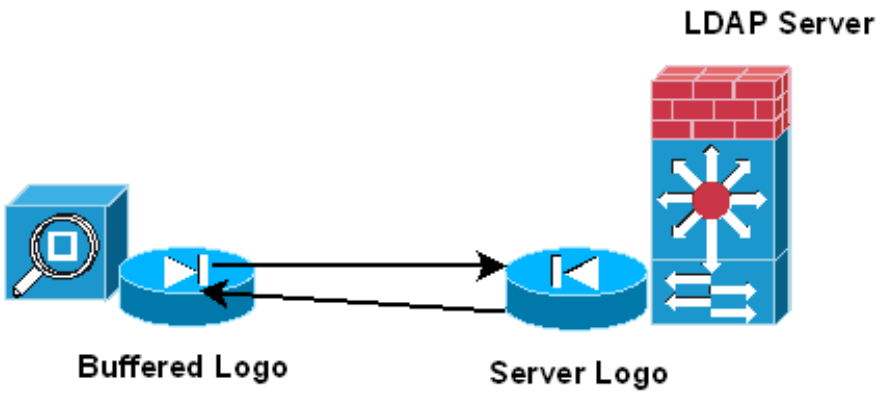

Figure 2. Logo comparison process

The main operations the java plug-in performs are:

- ldapbind, through which connection availability with the LDAP system is checked
- ldapcompare , the LDAP operation which is used for the login operation, the result of which can be either 5 for failure or 6 for a successful operation. Having the result 6 for this ldap

operation, the login will be able to be accomplished.

The actual structure[13][14][15] of a plug-in as it appears in plugin dump is presented below in figure 3.

```
dn:cn=checkmultimediapassword,cn=plugin,cn=subconfig
subentry
orclPluginFlexfield;minPwdLength: 8
objectclass: orclPluginConfig
objectclass: top
orclpluginname: CheckMultimediaPassword
orclplugintype: operational
orclplugintiming: pre
orclpluginldapoperation: ldapcompare
orclpluginenable: 1
orclpluginsubscriberdnlist:
cn=users,dc=ro,dc=phd,dc=com
orclpluginattributelist: jpegPhoto
orclPluginKind: Java
```

Figure 3. Plug-in dump

The attributes which are highly important for a plug-in are :

- orclpluginname is an attribute which stored the name of the plug-in
- orclplugintiming is an attribute which shows the time the plug-in fires. This attribute can be either pre, post, when or when_replace
- orclpluginsubscriberdnlist shows the LDAP tree on which the plug-in fires.
- orclpluginattributelist is the list of attributes on which the plugin will execute
- the last but not the least is orclPluginKind, which can be either Java or PLSQL

The most used java packages in developing plugins are:

```
java.io.*;
java.lang.*;
java.util.*;
javax.naming.*;
javax.naming.directory.*;
oracle.ldap.ospf.*;
```

As a final remark, once a plugin is compiled, is madatory for ospf.jar to be set in the environment variable CLASSPATH $CLASSPATH=$CLASSPATH:$ORACLE_HOME/ldap/jlib/ospf.jar.

4. Conclusions

4.1 Implementation possibilities

Implementation of such a system is possible via Oracle OID, as stated in the above sections.

Capabilities for partial image recognition are already present inside the system, making such functionalities relatively easy to include.

Moreover, more and more LDAP systems tend to include the possibility for multimedia information use, so a system implemented as in the present paper would leave the possibility of large scale inter-LDAP integration open.

4.2 Further research

An interesting direction to further our research is attempting the integration of inexact photos into the system, such as a photo of a person's face taken with a mobile phone camera [8].

ISSN 2067 – 4074

However, as described by other authors, this approach would require integrating some recognition mechanisms specific to artificial intelligence into LDAP.

4.3 Final remarks

Implementing such a system would present serious advantages over classical authentication and authorization mechanisms as it would allow for a larger amount of flexibility in managing the data.

Moreover, this approach would also allow for authentication based on valid, but partial authentication data, such as a part of the image used in the creation of the user account (should such an approach be needed).

References

[1] E. Rednic and A. Toma, Security Management in a Multimedia System, www.jaqm.ro, June 2009.

[2] E. REDNIC, A TOMA, Multimedia Flavor in Directory Services Environment, Informatica Economică vol. 13, no. 3/2009, pp 59-66

[3] S. Saha, Oracle Application Server 10g

Administration I, II, ORACLE, U.S.A., 2004.

[4] S. Saha, Oracle Application Server 10gr2

Administration I, II, ORACLE, U.S.A., 2004.

[5] L. Dhananjavan, Oracle Identity And

Management – all in one, ORACLE, U.S.A., 2007.

[6] S. Mavria, Oracle interMedia Feature

Overview, ORACLE, S.U.A., 2005.

[7] J. Mauro, Oracle interMedia Managing

Multimedia Content, ORACLE, U.S.A., 2008.

[8] I. Buhan, Cryptographic keys from noisy data: theory and applications, PhD thesis,University of Twente. CTIT Ph.D.-thesis series No. 08-129

[9] http://www.ietf.org/rfc/rfc2849.txt

[10] http://docs.sun.com/app/docs/doc/820-

2493/6ne3feem5?l=Ja&a=view

[11] http://download.oracle.com/docs/cd/

B28196_01/idmanage.1014/b15998/ldif

_appendix.htm#CHDIFIEG

[12] http://download-uk.oracle.com/docs/cd/

B28196_01/ idmanage.htm

[13] http://www.faqs.org/rfcs/rfc4511.html

[14] http://msdn.microsoft.com/en-us/library/

ms676813(VS.85).aspx

[15]http://download.oracle.com/docs/cd/B28196_01/idmanage.1014/b15991/plugin_intr.htm