CONNECTIONS BIDIRECTIONAL SYNCHRONIZATION

Michele Buccarello

12/10/13

Table of Contents

Authors

Michele Buccarello
Mail: buccarello.michele@gmail.com

Abstract

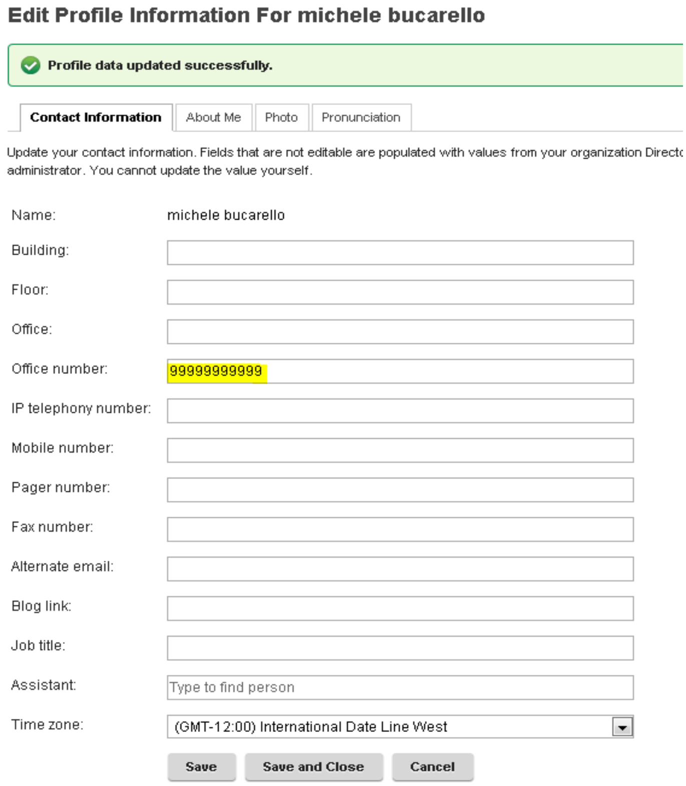
The user synchronization is normally made via sync_all_dns script that synchronize users information from LDAP to the EMPINST.EMPLOYEE table. If you synchronize a field such as "telephone number" when a user edits the value in IBM Connections at next run of sync_all_dns script the change is lost. To avoid this problem you must setup the synchronization from EMPLOYEE table to LDAP by using the script process_draft_updates.sh

Content

This document describes how to implement the synchronization process from EMPLOYEE table to the LDAP.

Sychronization Concept

When a user wants to update his telephoneNumber or other profile information he may use the “Edit Profiles” form, as you can see in the example below:



Edit Profile Information For michele bucarello

✓ Profile data updated successfully.

Update your contact information. Fields that are not editable are populated with values from your organization Directc administrator. You cannot update the value yourself.

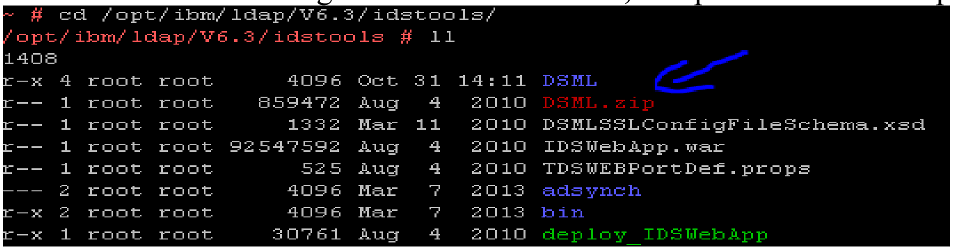
Name:	michele bucarello
Building:	
Floor:	
Office:	
Office number:	99999999999
IP telephony number:	
Mobile number:	
Pager number:	
Fax number:	
Alternate email:	
Blog link:	
Job title:	
Assistant:	Type to find person
Time zone:	(GMT-12:00) International Date Line West

this form updates the information in the EMPINST.EMPLOYEE table. So in this situation we have two different telephone numbers, one in Connections and another one in the LDAP. If we run the sync_all_dns.sh we lose the changes the user makes. To prevent this problem we can enable the synch from PROFILES to the LDAP user directory. The steps to follow are:

1) Install a DSLMv2 server in the ldap server. In this article we describe how to do this operation in Tivoli Directory Server 6.3
2) Configure IBM Connections for synchronization specific field. In this article only the Office Number becomes bidirectional.
3) Configure IBM Tivoli Directory Integrator with the DSLMv2 Server information and run the Assembly Line process_draft_updates.sh

Install DSLMv2 Server

Open the shell of the TDS server and go to the **idstools** folder, unzip the file DSML.zip.



```
~ # cd /opt/ibm/ldap/V6.3/idstools/
/opt/ibm/ldap/V6.3/idstools # ll
1408
r-x 4 root root     4096 Oct 31 14:11 DSML
r-- 1 root root   859472 Aug  4  2010 DSML.zip
r-- 1 root root     1332 Mar 11  2010 DSMLSSLConfigFileSchema.xsd
r-- 1 root root 92547592 Aug  4  2010 IDSWebApp.war
r-- 1 root root      525 Aug  4  2010 TDSWEBPortDef.props
--- 2 root root     4096 Mar  7  2013 adsynch
r-x 2 root root     4096 Mar  7  2013 bin
r-x 1 root root    30761 Aug  4  2010 deploy_IDSWebApp
```

Run the command **unzip DSML.zip -d DSML** to unzip the file in a folder named DSML. Make inside DSML a folder named **class,** inside in this folder we must put the jar used for the installation.

Download the following jar files:

- XMLParserAPIs.jar
- xercesImpl.jar
- activation.jar
- mail.jar
- soap.jar

You can download the XMLParserAPIs.jar and xercesImpl.jar from this link:

- http://archive.apache.org/dist/xml/xerces-j/Xerces-J-bin.2.8.0.tar.gz

Untar the file in the /tmp directory and rename the xml-apis.jar in XMLParserAPIs.jar, after the rename copy the two jars file in the **/opt/ibm/ldap/V6.3/idstools/DSML/class** folder.

The jar mail and activation you can get from this path **/opt/ibm/ldap/V6.3/appsrv/java/lib/** or from the TDI directory **/opt/ibm/TDI/V7.1/jars/3rdparty/others/**. Copy these files in the /opt/ibm/ldap/V6.3/idstools/DSML/class folder.

The soap.jar and soap.war can be dowloaded from this url:

- http://archive.apache.org/dist/ws/soap/version-2.3.1/soap-bin-2.3.1.tar.gz

Untar in the /tmp folder and copy the soap.jar in the /opt/ibm/ldap/V6.3/idstools/DSML/class folder and the soap.war in the application server profiles of the LDAP WAS Express, in my case the path is **/opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installableApps/soap.war**.

Now we need to install the soap.war via command line beause WAS express don't have the GUI. We need to go in the folder **/opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/bin/** and run this command:

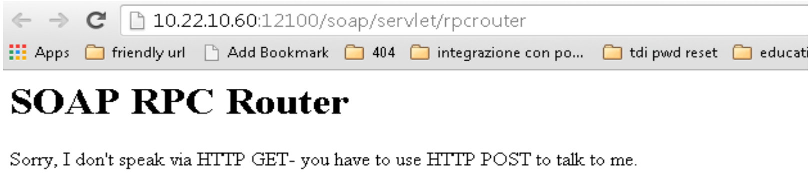
- wsadmin.sh -conntype NONE -c "\$AdminApp install
 {<WASinst>/installableApps/soap.war}
 {-configroot \"<WASinst>/config\"
 -node DefaultNode -usedefaultbindings -nodeployejb -appname soap.war
 -contextroot \"soap\"}"

in my case the command is

- wsadmin.sh -conntype NONE -c "\$AdminApp install {/opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installableApps/soap.war} {-configroot \"/opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/config\" -node DefaultNode -usedefaultbindings -nodeployejb -appname soap.war -contextroot \"soap\"}"

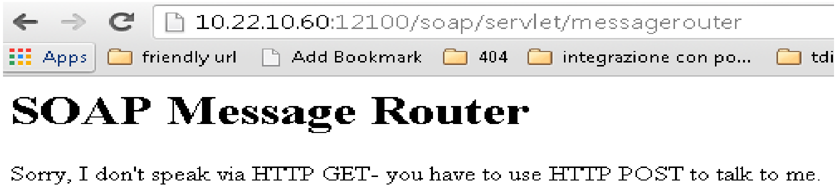
After the installation is completed you can check if the application has started via browser at the following urls:

- http://10.22.10.60:12100/soap/servlet/rpcrouter
- http://10.22.10.60:12100/soap/servlet/messagerouter



SOAP RPC Router

Sorry, I don't speak via HTTP GET- you have to use HTTP POST to talk to me.



SOAP Message Router

Sorry, I don't speak via HTTP GET- you have to use HTTP POST to talk to me.

Now you can start to install the DSLMv2 server. Before runing the install command we need to prepare the environment with these two export commands:

- export CLASSPATH= path of every jar in the class folder and jars folder inside the DSML folder

in my case the export command is

- export CLASSPATH=/opt/ibm/ldap/V6.3/idstools/DSML/class/soap.jar:/opt/ibm/ldap/V6.3/idstools/DSML/class/mail.jar:/opt/ibm/ldap/V6.3/idstools/DSML/class/activation.jar:/opt/ibm/ldap/V6.3/idstools/DSML/class/xerces.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/IBMLDAPJavaBer.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/auibase.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/dsml.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/regex4j.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/xercesImpl.jar:/opt/ibm/ldap/V6.3/idstools/DSML/jars/xmlParserAPIs.jar

The second export command is for the java program in PATH variabile:

- export PATH=$PATH:/opt/ibm/ldap/V6.3/appsrv/java/bin

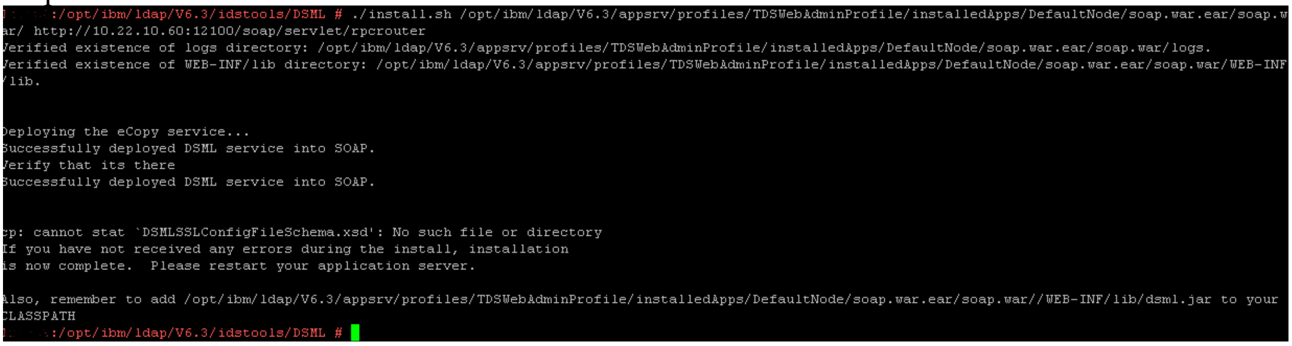
At this point you can run the installation command for DSLMv2 by writing in the shell this command:

- ./install.sh <SOAPHomeDir> <RPCRouterURL>

in my case

- ./install.sh /opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installedApps/DefaultNode/soap.war.ear/soap.war/ http://10.22.10.60:12100/soap/servlet/rpcrouter

At this point you have successfully deployed the application and you will see in the shell this output:



```
[redacted]:/opt/ibm/ldap/V6.3/idstools/DSML # ./install.sh /opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installedApps/DefaultNode/soap.war.ear/soap.w
ar/ http://10.22.10.60:12100/soap/servlet/rpcrouter
erified existence of logs directory: /opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installedApps/DefaultNode/soap.war.ear/soap.war/logs.
erified existence of WEB-INF/lib directory: /opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installedApps/DefaultNode/soap.war.ear/soap.war/WEB-INF
/lib.


eploying the eCopy service...
uccessfully deployed DSML service into SOAP.
erify that its there
uccessfully deployed DSML service into SOAP.


p: cannot stat `DSMLSSLConfigFileSchema.xsd': No such file or directory
f you have not received any errors during the install, installation
s now complete.  Please restart your application server.

lso, remember to add /opt/ibm/ldap/V6.3/appsrv/profiles/TDSWebAdminProfile/installedApps/DefaultNode/soap.war.ear/soap.war//WEB-INF/lib/dsml.jar to your
LASSPATH
[redacted]:/opt/ibm/ldap/V6.3/idstools/DSML #
```

Now we need to test if the DSMLv20 is up and running. Copy the files in the APPENDIX A, the DSMLv2.xsd and batchrequest.dsml in the /tmp directory and run the command:

- java com.ibm.ldap.dsmlClient.DsmlSoapClient "cn=root" "secret" -i "/tmp/batchrequest.dsml" -o "/tmp/result.xml" -l "/tmp/log.out" -x "urn:oasis:names:tc:DSML:2:0:core file:///tmp/DSMLv2.xsd" -d "3" -S "http://localhost:12100/soap/servlet/messagerouter" -s "ldap://localhost:389/"

In the result.xml you must see the users mail.

Configure the attribute stored in the EMP_DRAFT table

Run the wsadmin.sh script to implement the bidirectional synchronization to the telephonenumber field:

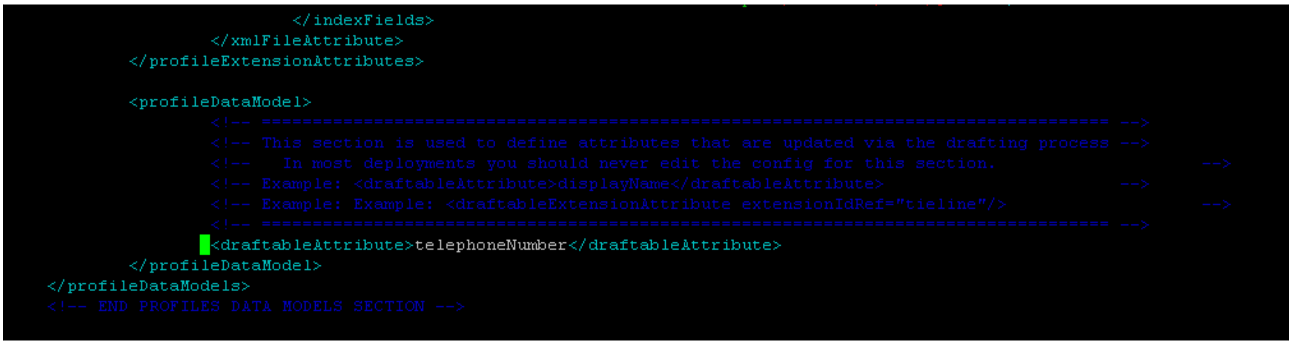
- cd /opt/ibm/WebSphere/AppServer/profiles/Dmgr01/bin
- ./wsadmin.sh -lang jython -user wasadmin -password wasadmin -port 8879

now in the wsadmin console run the commands:

- wsadmin>execfile("profilesAdmin.py")
- wsadmin>ProfilesConfigService.checkOutConfig("/tmp",AdminControl.getCell())

go in the tmp folder and edit the profiles-config.xml and the attribute to be synchronized.
In this case we choose the telephonenumber:

- <draftableAttribute>telephoneNumber</draftableAttribute>



```
                </indexFields>
            </xmlFileAttribute>
        </profileExtensionAttributes>

        <profileDataModel>
            <!-- ================================================================================ -->
            <!-- This section is used to define attributes that are updated via the drafting process -->
            <!--   In most deployments you should never edit the config for this section.             -->
            <!-- Example: <draftableAttribute>displayName</draftableAttribute>                    -->
            <!-- Example: Example: <draftableExtensionAttribute extensionIdRef="tieline"/>             -->
            <!-- ================================================================================ -->
            <draftableAttribute>telephoneNumber</draftableAttribute>
        </profileDataModel>
    </profileDataModels>
    <!-- END PROFILES DATA MODELS SECTION -->
```

save the file and check-in the configuration in IBM Connections with this command:

- wsadmin>ProfilesConfigService.checkInConfig()
- wsadmin>exit

At this point we must reboot to apply the change. After rebooting, everytime a user updates the Office Number field the EMPINST.EMPLOYEE is updated with the new telephonenumber and the EMPINST.EMP_DRAFT table is written with the new information, see the following example:

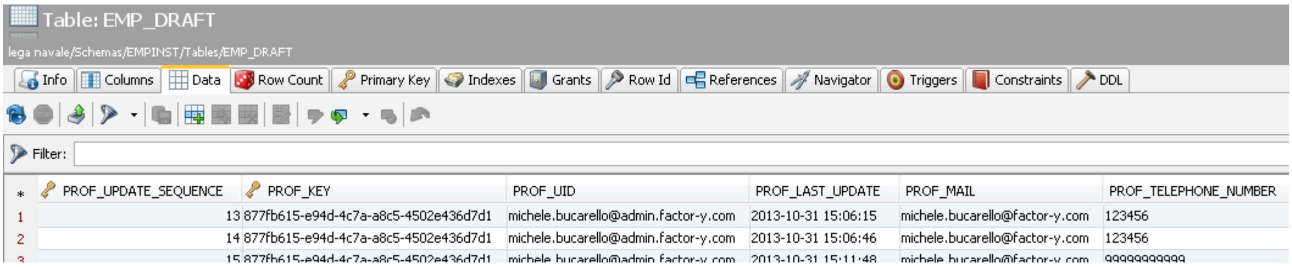


Table: EMP_DRAFT

lega navale/Schemas/EMPINST/Tables/EMP_DRAFT

*	PROF_UPDATE_SEQUENCE	PROF_KEY	PROF_UID	PROF_LAST_UPDATE	PROF_MAIL	PROF_TELEPHONE_NUMBER
1	13	877fb615-e94d-4c7a-a8c5-4502e436d7d1	michele.bucarello@admin.factor-y.com	2013-10-31 15:06:15	michele.bucarello@factor-y.com	123456
2	14	877fb615-e94d-4c7a-a8c5-4502e436d7d1	michele.bucarello@admin.factor-y.com	2013-10-31 15:06:46	michele.bucarello@factor-y.com	123456
3	15	877fb615-e94d-4c7a-a8c5-4502e436d7d1	michele.bucarello@admin.factor-y.com	2013-10-31 15:11:48	michele.bucarello@factor-y.com	99999999999

Tivoli Directory Integrator

To process the rows in the EMPINST.EMP_DRAFT table we need to setup the assembly line process_draft_updates.sh. Edit the file profiles_tdi.properties and set these properties:

- monitor_changes_dsml_server_url=http://10.22.10.60:12100/soap/servlet/messagerouter
- monitor_changes_dsml_server_password=root
- monitor_changes_dsml_server_username=secret

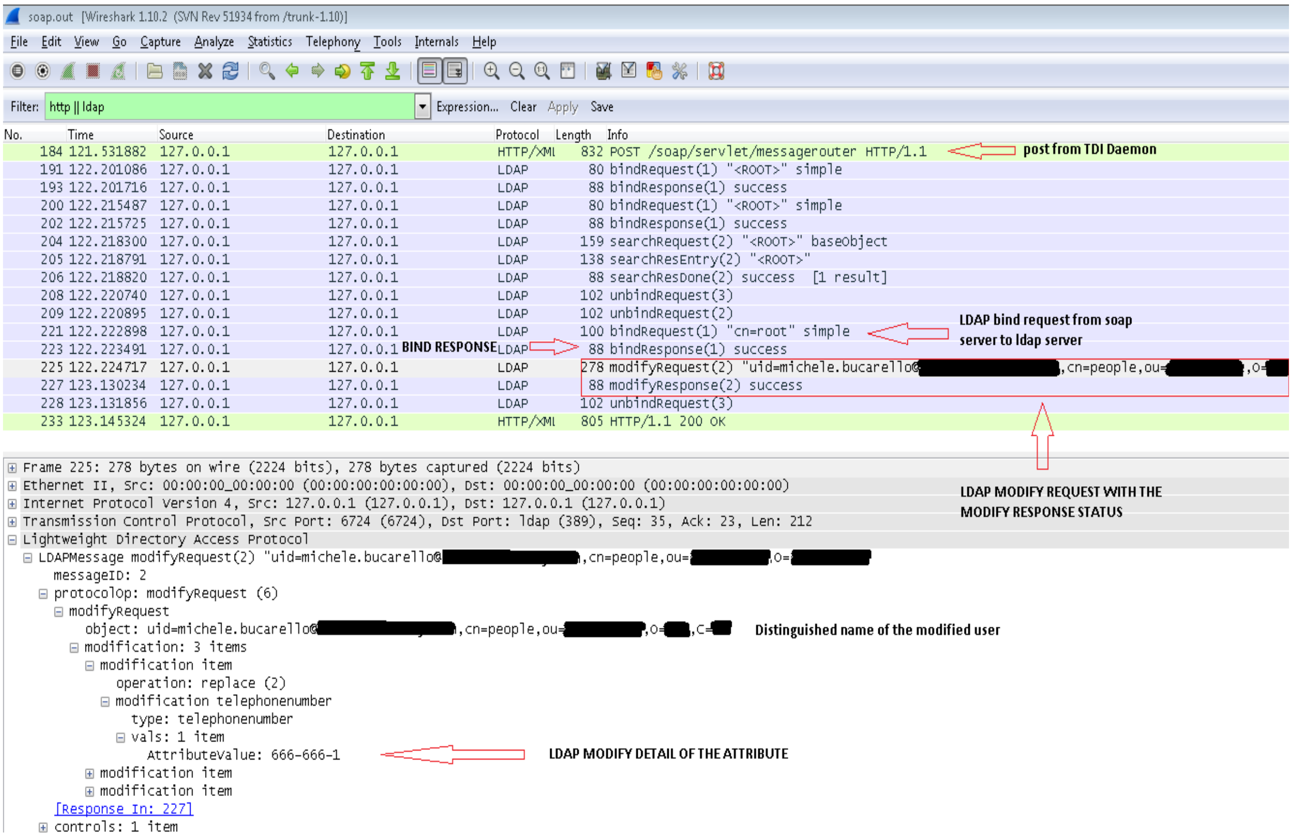
now we need to run the the script process_draft_updates.sh, this script starts a daemon that runs in the background. Every 30 seconds this daemon processes the EMP_DRAFT table and sends a DSML message for every row to the messagerouter.

Troubleshoot

To troubleshoot the comunication between the daemon and the messagerouter we can use tcpdump to analize the SOAP traffic by running this command on the TDS shell:

- tcpdump -s 0 -w /tmp/dsml_soap.pcap -i lo

we can open the file .pcap with Wireshark, the following image shows a successful scenario


soap.out [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: http || ldap

```
No.   Time        Source     Destination  Protocol  Length  Info
  184 121.531882  127.0.0.1  127.0.0.1    HTTP/XML   832 POST /soap/servlet/messagerouter HTTP/1.1
  191 122.201086  127.0.0.1  127.0.0.1    LDAP        80 bindRequest(1) "<ROOT>" simple
  193 122.201716  127.0.0.1  127.0.0.1    LDAP        88 bindResponse(1) success
  200 122.215487  127.0.0.1  127.0.0.1    LDAP        80 bindRequest(1) "<ROOT>" simple
  202 122.215725  127.0.0.1  127.0.0.1    LDAP        88 bindResponse(1) success
  204 122.218300  127.0.0.1  127.0.0.1    LDAP       159 searchRequest(2) "<ROOT>" baseObject
  205 122.218791  127.0.0.1  127.0.0.1    LDAP       138 searchResEntry(2) "<ROOT>"
  206 122.218820  127.0.0.1  127.0.0.1    LDAP        88 searchResDone(2) success  [1 result]
  208 122.220740  127.0.0.1  127.0.0.1    LDAP       102 unbindRequest(3)
  209 122.220895  127.0.0.1  127.0.0.1    LDAP       102 unbindRequest(2)
  221 122.222898  127.0.0.1  127.0.0.1    LDAP       100 bindRequest(1) "cn=root" simple
  223 122.223491  127.0.0.1  127.0.0.1    LDAP        88 bindResponse(1) success
  225 122.224717  127.0.0.1  127.0.0.1    LDAP       278 modifyRequest(2) "uid=michele.bucarello@[redacted],cn=people,ou=[redacted],o=[redacted]
  227 123.130234  127.0.0.1  127.0.0.1    LDAP        88 modifyResponse(2) success
  228 123.131856  127.0.0.1  127.0.0.1    LDAP       102 unbindRequest(3)
  233 123.145324  127.0.0.1  127.0.0.1    HTTP/XML   805 HTTP/1.1 200 OK
```

post from TDI Daemon

LDAP bind request from soap server to ldap server

BIND RESPONSE

LDAP MODIFY REQUEST WITH THE MODIFY RESPONSE STATUS

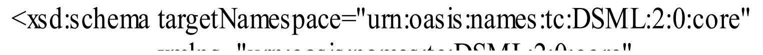
```
⊞ Frame 225: 278 bytes on wire (2224 bits), 278 bytes captured (2224 bits)
⊞ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
⊞ Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
⊞ Transmission Control Protocol, Src Port: 6724 (6724), Dst Port: ldap (389), Seq: 35, Ack: 23, Len: 212
⊟ Lightweight Directory Access Protocol
  ⊟ LDAPMessage modifyRequest(2) "uid=michele.bucarello@[redacted],cn=people,ou=[redacted],o=[redacted]
      messageID: 2
    ⊟ protocolOp: modifyRequest (6)
      ⊟ modifyRequest
          object: uid=michele.bucarello@[redacted],cn=people,ou=[redacted],o=[redacted],c=[redacted]
        ⊟ modification: 3 items
          ⊟ modification item
              operation: replace (2)
            ⊟ modification telephonenumber
                type: telephonenumber
              ⊟ vals: 1 item
                  AttributeValue: 666-666-1
          ⊞ modification item
          ⊞ modification item
      [Response In: 227]
    ⊞ controls: 1 item
```

Distinguished name of the modified user

LDAP MODIFY DETAIL OF THE ATTRIBUTE

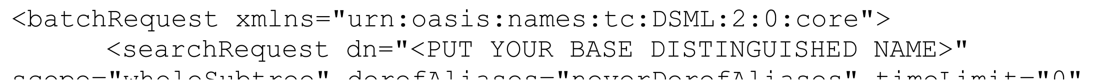
APPENDIX A

DSMLv2.xsd



```
<xsd:schema targetNamespace="urn:oasis:names:tc:DSML:2:0:core"
            xmlns="urn:oasis:names:tc:DSML:2:0:core"
```

batchrequest.dsml



```
<batchRequest xmlns="urn:oasis:names:tc:DSML:2:0:core">
      <searchRequest dn="<PUT YOUR BASE DISTINGUISHED NAME>"
scope="wholeSubtree" derefAliases="neverDerefAliases" timeLimit="0"
```