Recording and Playback HOWTO

LCCS recording and playback is designed to store the "archive" of a recording on a customer-provided server. When recording LCCS, will store the recorded stream locally, then when the recording ends it will package them in a zip file and send them to the customer server. When playing back a recording, LCCS will retrieve the archive, uncompress it and playback the expanded streams.

This document describes how to setup a "repository" server and enable recording and playback for an LCCS account.

Setting up a repository server (WebDAV) :

In order to enable recording and playback the developer needs to provide and register a "repository server". LCCS will use a simple HTTP/WebDAV PUT to store a recording and a GET to fetch the recording. Any HTTP server that supports PUT and GET of resources can be used as the "repository server" but a server that supports WebDAV out of the box is probably the fastest and easiest way to start.

Two servers that can be used this way are Apache (with mod_dav) and Tomcat (with the embedded WebDAV servlet). Here is how to configure them:

Using Apache with mod_dav

- Install a recent version of Apache (i.e. httpd 2.x)
- Enable mod_dav in /etc/httpd/conf/httpd.conf

```
LoadModule dav_module modules/mod_dav.so
LoadModule dav_fs_module modules/mod_dav_fs.so

#
# WebDAV module configuration section.
#
<IfModule mod_dav_fs.c>
    # Location of the WebDAV lock database.
    DAVLockDB /var/lib/dav/lockdb
</IfModule>
```

- Configure mod_dav and repository folder (i.e. in /etc/httpd/conf.d/dav.conf) :

```
<Directory /var/www/html/dav>
DAV on
AllowOverride All
Options Indexes
Order deny,allow
Allow from all
</Directory>
```

- If needed enable stricter security (i.e. add basic authentication or IP filtering, disable indexes, etc.)
- Create the "repository folder" (i.e. /var/www/html/dav)
- Restart Apache, access http://localhost:8080/dav/ and verify that the access succeeds.

For more information refer to the Apache documentation: http://httpd.apache.org/docs/2.0/mod/mod_dav.html

Using Tomcat with the WebDAV servlet

- Install a recent version of Tomcat (i.e. Tomcat 6.x)
- Create the 'webdav' application:

```
mkdir -p .../tomcat/webapps/webdav/WEB-INF
```

- Configure the 'webdav' servlet and make sure writes are enabled (create and edit .../tomcat/webapps/webdav/WEB-INF/web.xml) :

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
version="2.5">
 <servlet>
  <servlet-name>webdav</servlet-name>
  <servlet-class>org.apache.catalina.servlets.WebdavServlet</servlet-class>
  <init-param>
   <param-name>debug</param-name>
   <param-value>0</param-value>
  </init-param>
  <init-param>
   <param-name>listings</param-name>
   <param-value>true</param-value>
  </init-param>
  <init-param>
   <param-name>readonly</param-name>
   <param-value>false</param-value>
  </init-param>
 </servlet>
 <servlet-mapping>
  <servlet-name>webdav</servlet-name>
  <url-pattern>/*</url-pattern>
 </servlet-mapping>
</web-app>
```

- If needed enable stricter security (i.e. with an authentication filter or IP filtering, etc.)
- Restart tomcat, access http://localhost:8080/webdav/ and verify that the access succeeds.

For more information refer to the Apache documentation:
http://tomcat.apache.org/tomcat-6.0-doc/api/org/apache/catalina/servlets/WebdavServlet.html

Register the endpoint

In order to register the repository endpoint the server integration scripts have been updated to provide a registerRepository method (similar to registerHook for event handling). You can use this API to register the repository server and enable recording and playback. Please refer to the documentation found at http://learn.adobe.com/wiki/display/lccs/LiveCycle+Collaboration+Service for detailed information.

Repository server authentication

In this preview release we don't provide any support for storing/using any form of authentication token when accessing the "repository server". But given that we use a standard command line too (curl) to issue the HTTP requests to the repository server authenticated access can be achieved by providing the appropriate credentials on the repository URL:

```
http://<username>:<password>@hostname/
```

Notes:

1. Basic authentication will send the credentials in clear so it's suggested that a secure URL is used (i.e. https://lccs:lccspassword@lccsrepo.com/dav/) or digest authentication is used.
2. Another way of dealing with authentication (i.e. making sure the requests for storing/retrieving recordings come from LCCS) is to instead implement IP filtering.
3. Currently the repository URL is stored in the LCCS repository unencrypted, so if the URL credentials are added to the URL they will also be unencrypted. Again, this will be fixed with the official release of this feature.

Archive files and requests to repository server

As mentioned before when recording the recorded streams are saved on the LCCS servers local disk and later packaged in a zip file and sent to the repository server.

The archive file name identifies the account, room and archiveID associated to the recording. For example:

```
na2-sdk-233e070b-f1e0-4df5-af1e-8527579cc8a2_x002F_camerarecording_x002F___defaultArchive__.zip
```

In this example 'na2-sdk-233e070b-f1e0-4df5-af1e-8527579cc8a2' is the account id (for example for the account 'testaccount'), 'camerarecording' is the room name and '*_defaultArchive' is the archiveID (if no id is specified when starting a recording the default name 'defaultArchive_*' is used.

The _x002F_ that separates the fields is the '/' character encoded as ISO9075.

LCCS will use the following commands to store and retrieve archives from the repository server:

STORE archive

```
curl -k --anyauth -T <archive.zip> <repository-url>
```

RETRIEVE archive

```
curl -k --anyauth -o <archive.zip> <repository-url>/<archive.zip>
```

Notes:

1. In this preview release the '-k' option is used to allow for self-signed SSL certificates. In the final release we will either disable this option or make it configurable for the account and/or repository endpoint.
2. The '--anyauth' is used to allow for both basic and digest authentication (or any other form of authentication that curl supports). Again, in this preview release when authentication is used the credentials need to be passed as part of the repository endpoint URL.
3. The '--T' option on the STORE command will generate, as explained before, an HTTP PUT request. LCCS expects that multiple PUT for the same resource will override the resource without returning an error. The developer can avoid overriding a recording at the client application level by providing a different value for archiveID every time a recording is started.

Notifications

For this preview release, LCCS doesn't not provide any mechanism to notify the developer and/or a server-side application associated to an LCCS account that recordings are being sent to the repository server. If required the developer can implement such a feature at the repository server level by, for example, adding a filter in front of the WebDAV servlet in a Java/J2EE application or using a similar approach for other repository implementations.