

# MatrixStore Plugins for Final Cut Server User Guide





Object Matrix Ltd.  
©2008, Object Matrix Ltd, All rights reserved.

Under the copyright laws, this manual may not be copied, in whole or in part, without the written consent of Object Matrix. Your rights to the software are governed by the accompanying software license agreement.

Every effort has been made to ensure that the information in this manual is accurate. While every precaution has been taken in the preparation of this document, Object Matrix assumes no responsibility for errors or omissions or for damages resulting from the use of the information contained herein. Object Matrix is not responsible for clerical or printing errors.

Object Matrix  
Orchard House  
Caerphilly Business Park  
Caerphilly  
CF83 3GP  
[www.object-matrix.com](http://www.object-matrix.com)

Apple, the Apple logo, Mac, Mac OS, Macintosh, and Xserve are trademarks of Apple Computer, Inc., registered in the U.S. and other countries.

Java is a trademark of Sun Microsystems, Inc. Other company and product names mentioned herein are trademarks of their respective companies. Mention of third-party products is for informational purposes only and constitutes neither an endorsement nor a recommendation. Object Matrix assumes no responsibility with regard to the performance or use of these products.

# Table of Contents

About Software for MatrixStore Clusters.....	4
Overview.....	5
Installing MatrixStore Plugins for Final Cut Server.....	5
DMG and Installer.....	5
Backing up Final Cut Server.....	6
Description.....	6
Preparation.....	6
Usage.....	6
Detailed information.....	7
Archiving Final Cut Server Productions .....	8
Preparation.....	8
Edit fcsvr plist file.....	8
Edit the sudoers file.....	8
Copy vault details file.....	9
Create new MatrixStore archive device in FCS.....	9
Adding Metadata in Final Cut Server.....	10
Metadata added to archived assets.....	14
Usage.....	17
Un-install.....	19
Appendix.....	20

## About Software for MatrixStore Clusters

You can manage a MatrixStore cluster from a host computer or a remote computer.

You use three applications to setup and administrate/monitor your MatrixStore cluster, MatrixStore Setup Assistant, MatrixStore Maintenance application and the MatrixStore Admin application. All MatrixStore applications can be downloaded from the Object Matrix website.

You use MatrixStore Setup Assistant to prepare an Xserve server for MatrixStore software installation.

You use MatrixStore Maintenance application to perform MatrixStore software maintenance, including:

- Install MatrixStore Software
- Upgrade MatrixStore Software

Use the MatrixStore Administration application to perform administrative tasks on a MatrixStore cluster. Administration tasks include:

- Administer MatrixStore Cluster
  - Add vaults
  - Monitor Usage
  - Enable SNMP monitoring
- Administer MatrixStore Vault(s)
  - Add users
  - Monitor Vault capacity
- Monitor MatrixStore Cluster Health at system and node level

For more technical support information please visit:

<http://www.getsatisfaction.com/objectmatrix>

## Overview

MatrixStore plugins for Final Cut Server are a set of scripts and utilities that provide extra features to FCS extending its usability and allowing a seamless integration of MatrixStore as an archive device into your workflows.

- Allow binding strategic metadata together with Final Cut Server assets and productions along the whole process from creation through edition to archiving by storing that metadata with their archived copies on MatrixStore device. Once archived, the user has the ability to find assets and productions easily on the archive device – MatrixStore using the search-driven Dropspot application.
- Provide easy backup of Proxies together with FCS database. By using Final Cut Server Backup function and MatrixStore plugins, the user can automate archiving FCS database and Proxies and keep them together in case of unexpected events and failures. Restoring the whole FCS without time consuming re-creation of Proxies is a huge advantage and allows to save time, especially for installations with vast array of assets.
- Give the possibility to archive whole Final Cut Server productions (project) directly from Final Cut Server client application to MatrixStore device, which is extremely useful for project-oriented users.

## Installing MatrixStore Plugins for Final Cut Server

You can install MatrixStore Plugins on any mac running Final Cut Server.

Please note that a MatrixStore cluster must be installed with vaults and vault users created before this process.

You must also install MatrixStore Connect as described in its users guide and create at least one account. Connect presents MatrixStore vaults as FTP accounts to Final Cut Server.

## DMG and Installer

Mount the MatrixStore Disk Image on your computer, go to “FinalCutServer integration” directory, run MatrixStore-FCS Plugins Installer.pkg and complete the installation process.

The installer installs all the necessary files in:

/Library/Application Support/MatrixStore/FCS Support

You can access the log files here:

/Library/Logs/MatrixStore/FCS Support/logs/

To finish installation you have to follow the instructions in Backing up Final Cut Server and Archiving Final Cut Server Productions. These steps are mandatory.

# Backing up Final Cut Server

## Description

Scripts located in

/Library/Application Support/MatrixStore/FCS Support/Backup

allow you to automate the action of backing up Final Cut Server Proxies and database.

Backup of FCS DB can be done automatically by FCS with a scheduled action which creates a zip file.

mxs\_fcs\_backup\_db.rb script looks for the latest zip file (latest modification date) created as a result of a backup action from FCS. The zip file is packed together with all proxies files into a single tar archive and then sent to the FTP server pointed as a parameter to the script.

## Preparation

The scripts to back up proxies requires at least one scheduled backup action in Final Cut Server Preferences. Make sure that you set up at least one Scheduled Backups in FCS Preferences (Backup tab).

## Usage

To launch the main script go to the script directory and run:

```
sudo ./mxs_fcs_backup_db.rb IP port user pass [tempdir]
```

Script has to be launched as sudo.

Parameters for the script are the following:

IP	- IP address of the FTP server where DB and Proxies will be kept
port	- port on which FTP listens for connections
user	- user account name on the FTP e.g. 1028b248-00a1-11dd-8e6f-a1b297666b5b
pass	- user password in single or double quotes e.g. 'tell least head arrange variety'
[tempdir]	- optional parameter which specifies where the tar file will be written before sending it to the FTP server.

Before sending it, the tar file is created on tempdir location if user specifies this parameter or under /tmp if he doesn't.

The tar archive is deleted from temporary directory upon errors or successful upload to the FTP server.

Script `mxs_fcs_backup_db.rb` sends some progress information to standard output and writes the same information to the log file (under log directory).

The log file isn't overwritten every time script is launched, instead, new information is appended to the log file.

You can also use Automator to launch the scripts few minutes after FCS database zip archive is created by scheduled backups.

## Detailed information

The archive file (with DB content and Proxies) has a name which is created on the basis of date and looks like this:

```
fcs_db_backup-2008-04-11_at_0208PM.tar
```

That file is written on a certain location on the FTP server, which is:

```
/fcs_backups
```

where `/` points to the root directory of the user account passed as a parameter to the script.

After successful upload the modification date from DB backup zip file becomes a new reference point.

When the script is launched again, archiving of the Proxies and FCS database zip file will take place only if the new zip file modification date is later than the zip file which was created last time when the script was launched. If there's no newer zip file than that reference point, the script ends its execution.

The reference point is kept in the file:

```
/Library/Application Support/MatrixStore/FCS Support/Backup/config/lastDB_backup.txt
```

You can delete that file if you want to force archiving the latest zip file when the script is launched next time

The paths where zip files are being searched are taken from FCS Preferences Backup tab. `mxs_fcs_backup_db.rb` can handle several entries in Schedule Backup window. In that case, it compares the dates of all zip files from all locations and picks up the newest.

Disabling the Schedule does not have influence on the script, `mxs_fcs_backup_db.rb` still check the zip files kept under Schedule's location.

It is important to keep only backup `.zip` files in directories specified in FCS Preferences. `mxs_fcs_backup_db.rb` doesn't check if the zip file is actually a DB backup.

You may prefer to do FCS database backups to a Proxies device as they are often backed up to tape or another device. `mxs_fcs_backup_db.rb` automatically checks if the path to Proxies and the path to zip files are the same. In that case it doesn't add zip file to a tar archive as zip file will be tar'ed anyway.

# Archiving Final Cut Server Productions

## Preparation

Before starting using the provided plugin to automate archiving productions you need to do additional steps in order to prepare Final Cut Server to launch external scripts.

All these steps are necessary and must be done by Final Cut Server administrator with admin privileges on FCS machine. Wrong order or missing points in preparation procedure may effect that plugin and it will not work properly so we strongly advise backing up FCS database before going on.

## Edit fcsvr plist file

To allow Final Cut Server launch the provided scripts you need to edit FCS plist file.

Run from command line:

```
sudo vi /Library/LaunchDaemons/com.apple.FinalCutServer.fcsvr_stored.plist
```

In <dict> section for a key <EnvironmentVariables> you need to add PATH key and associated value:

```
<key>PATH</key>  
<string>/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/Library/  
Application Support/MatrixStore/FCS Support/ArchiveProduction/scripts</string>
```

If specified key already exists just add missing entries.

Do not use escape character in path.

Write your changes and exit vi (:wq in vi command line)

## Edit the sudoers file

In order to allow scripts changing metadata on assets and productions you need to edit the sudoers file. This will enable launching FCS command line tools without entering the password.

To edit /etc/sudoers file; run command

```
sudo visudo
```

in the section

```
# User privilege specification
```

add a single line:

```
username hostname=NOPASSWD: /Library/Application\ Support/Final\ Cut\  
Server/Final\ Cut\ Server.bundle/Contents/MacOS/fcsvr_client
```

As username write the name of the user who is owner of the process fcsvr\_stored.  
You can check that running from command line command:

```
ps aux | grep fcs
```

As a hostname put the name of the machine where FCS is installed. You can use domain-type name only if the FCS machine has valid name. You can check it from shell running:

```
hostname
```

Write your changes and exit vi (:wq in vi command line)

## Copy vault details file

The scripts launch a Java application in order to add some metadata for archived assets. The application contacts with MatrixStore cluster. That is why it is necessary to provide valid vault details to specify where archived assets are located.

Copy file production-archive.vault with your vault details to

```
/Library/Application Support/MatrixStore/FCS Support/ArchiveProduction/config
```

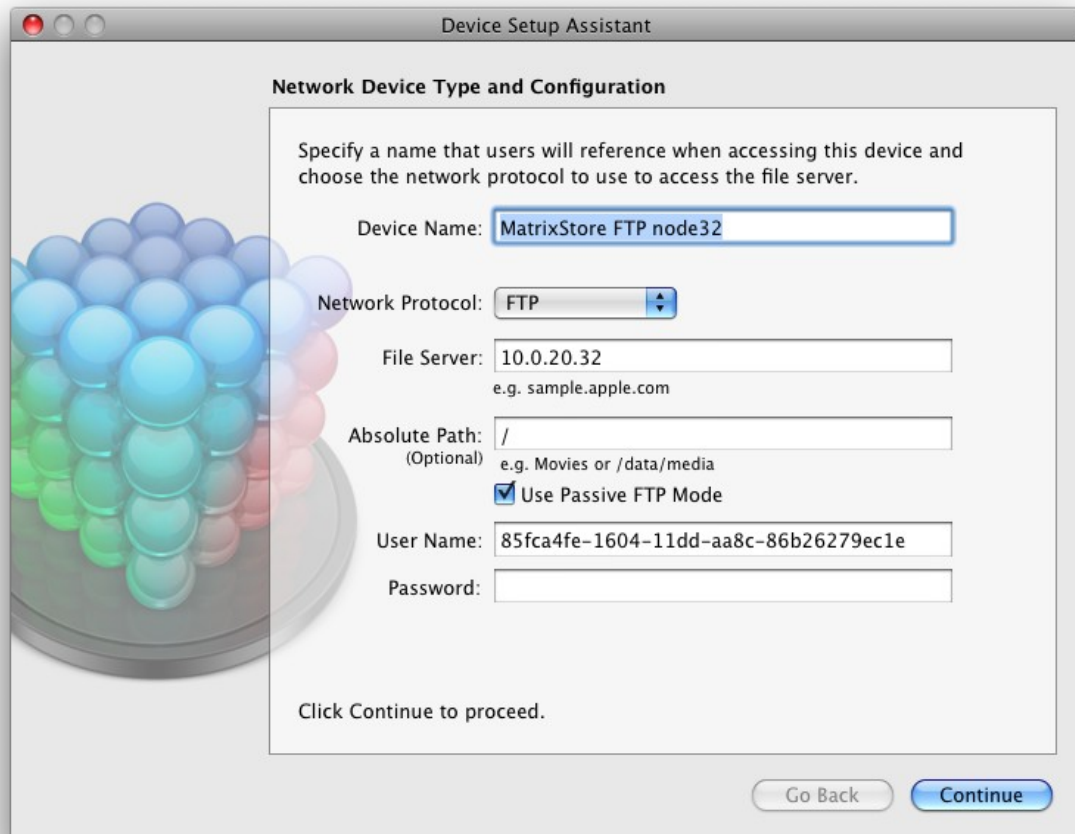
An example of production-archive.vault can be found in the appendix.

## Create new MatrixStore archive device in FCS

Create new FTP device using System Preferences -> Final Cut Server.

Do not use Administration Panel from FCS client as password specified for a device created in such a way won't be added properly to the keychain in the host machine.

Device name has to start with 'MatrixStore' (without quotes) e.g. MatrixStore Vault 1



## Adding Metadata in Final Cut Server

The next step is to add the necessary metadata elements (metadata fields, subscriptions, responses etc.) in Final Cut Server. You can find all necessary details in Final Cut Server Administration Guide document.

To work properly, MatrixStore plugins require the creation of a set of Final Cut Server metadata in following groups:

- Lookup
- Metadata Field
- Metadata Group
- Responses
- Subscriptions

## Lookup

- 1) "Mxs Production Archive Status"  
Data Type: Unicode String  
[Name] / [Value]
  - Archiving / Archiving
  - Archived / Archived
  - Restoring / Restoring
  - Restored / Restored
  
- 2) "Mxs Production Archive Command"  
Data Type: Unicode String  
[Name] / [Value]
  - Archive / Archive
  - Restore / Restore
  - Neutral / Neutral
  
- 3) "Mxs Asset Archive Command"  
Data Type: Unicode String  
[Name] / [Value]
  - Archive / Archive
  - Restore / Restore
  - Neutral / Neutral

## Metadata Field

- 1) Mxs Production Archive Status  
Data Type: Unicode String  
Category: None  
Lookup Values: Mxs Production Archive Status

Check if added Metadata field has FieldID = [PA\\_MD\\_CUST\\_MXS\\_PRODUCTION\\_ARCHIVE\\_STATUS](#) and if not change that string in mxs\_archive\_prod.rb & mxs\_restore\_prod.rb

- 2) Mxs Production Archive Command  
Data Type: Unicode String  
Category: None  
Lookup Values: Mxs Production Archive Command
  
- 3) Mxs Asset Archive Command  
Data Type: Unicode String  
Category: None  
Lookup Values: Mxs Asset Archive Command

Check if added Metadata field has FieldID = [PA\\_MD\\_CUST\\_MXS\\_ASSET\\_ARCHIVE\\_COMMAND](#) and if not change that string in mxs\_archive\_prod.rb & mxs\_restore\_prod.rb

## Metadata Group

- 1) Create "Mxs Production"  
Fields:
  - add 'Mxs Production Archive Command'  
Editable: true  
Field Lookup: Mxs Production Archive Command
  - add 'Mxs Production Archive Status'  
Editable: false  
Field Lookup: Mxs Production Archive Status  
Actions: (at least) Edit details, View details.  
Remove 'Create' from 'Actions' list Metadata Sets: e.g. Commercial (Production)
  
- 2) Create "Mxs Asset Metadata"  
Fields:
  - add Mxs Asset Archive Command  
Editable: true  
Field Lookup: Mxs Asset Archive Command Label only: true  
Actions: (at least) Edit details, View details..  
Remove 'Create' from 'Actions' list Metadata Sets: Document, Graphics, Media, All Media Assets(if available)
  
- 3) modify "Production Filter"  
Fields:
  - add 'Mxs Production Archive Command'  
Editable: true  
Field Lookup: Mxs Production Archive Command
  - add 'Mxs Production Archive Status'  
Editable: true  
Field Lookup: Mxs Production Archive Status  
Actions: (at least) Create, Edit details, View details, Search
  
- 4) modify "Asset Filter"  
Fields:
  - add 'Mxs Asset Archive Command'  
Editable: true  
Field Lookup: Mxs Asset Archive Command  
Actions: (at least) Create, Edit details, View details, Search

## Responses

Create following responses:

- 1) Mxs Archive Asset to MatrixStore  
Response Action: Move to Archive  
Archive: select MatrixStore FTP device created in one of the previous steps
  
- 2) Mxs Restore Asset from MatrixStore  
Response Action: Restore from Archive

- 3) Mxs Archive Production to MatrixStore  
 Response Action: Run external script or command Command  
 Path: mxs\_fcs\_archive\_prod.rb  
 Command Parameters: [Production ID]
- 4) Mxs Restore Production from MatrixStore  
 Response Action: Run external script or command Command  
 Path: mxs\_fcs\_restore\_prod.rb  
 Command Parameters: [Production ID]
- 5) Mxs Set Production Command to Neutral  
 Response Action: Set Production Metadata  
 Production Metadata: on Mxs Production:  
 Mxs Production Archive Command = Neutral;  
 Mxs Production Archive Status = None
- 6) Mxs Set Asset Command to Neutral  
 Response Action: Set Asset Metadata  
 Production Metadata: on Mxs Production:  
 Mxs Production Archive Command = Neutral;  
 Mxs Production Archive Status = None
- 7) Mxs Add Metadata to Archived Asset  
 Response Action: Run external script or command Command  
 Path: mxs\_fcs\_archive\_asset.rb  
 Command Parameters: [Asset ID]

## Subscriptions

Create following subscriptions:

- 1) Sub, Asset, Mxs Cmd Archive – Archive to MatrixStore  
 Subscribe to: Asset  
 Enabled = true  
 Event Type Filter = Modified  
 Response List:
  - Mxs Archive Asset to MatrixStore
  - Mxs Set Asset Command to Neutral
 Asset Filter:  
 Mxs Asset Archive Command: Equals, Archive, Trigger if changed = true
- 2) Sub, Asset, Mxs Cmd Restore – Restore from MatrixStore  
 Subscribe to: Asset  
 Enabled = true  
 Event Type Filter = Modified  
 Response List:
  - Mxs Restore Asset from MatrixStore
  - Mxs Set Asset Command to Neutral
 Asset Filter:  
 Mxs Asset Archive Command: Equals, Restore, Trigger if changed = true

- 3) Sub, Asset, Mxs Wait for Archived – Tag on MatrixStore  
Subscribe to: Asset  
Enabled = true  
Event Type Filter = Modified  
Response List:
  - Mxs Add Metadata to Archived Asset
  - Mxs Set Asset Command to Neutral
Asset Filter:  
Archive Status: Equals, Offline, Trigger if changed = true
  
- 4) Sub, Prod, Mxs Cmd Archive – Archive to MatrixStore  
Subscribe to: Production  
Enabled = true  
Event Type Filter = Modified  
Response List:
  - Mxs Archive Production to MatrixStore
  - Mxs Set Production Command to Neutral
Production Filter:  
Mxs Production Archive Command: Equals, Archive, Trigger if changed = true
  
- 5) Sub, Prod, Mxs Cmd Restore – Restore from MatrixStore  
Subscribe to: Production  
Enabled = true  
Event Type Filter = Modified  
Response List:
  - Mxs Restore Production from MatrixStore
  - Mxs Set Production Command to Neutral
Production Filter:  
Mxs Production Archive Command: Equals, Restore, Trigger if changed = true

Now, **REBOOT FINAL CUT SERVER MACHINE** to complete the installation.

## Metadata added to archived assets

Searching for archived assets and productions on MatrixStore device can be done by using Dropspot application. This can be done by specifying either an exact key and value or by specifying list of keywords in Dropspot Search tab.

As an example of specifying an exact key/value metadata:

ASSET\_NUMBER=99 or  
CUST\_TITLE="MatrixStore advertisement"

in Dropspot Search tab.

As an example of specifying only keywords:

production commercial juice home

in Dropspot Search tab.

Both of them are useful to find assets or productions using and depends on the user which one is more convenient for him/her.

Final Cut Server Administrator can specify which metadata should be copied to

MatrixStore to provide that search functionality in DropSpot. The list of metadata fields copied from Final Cut Server to MatrixStore device is defined in config files (plist type).

MatrixStore Plugins use two config files: one for assets and one productions.

Config files also allow Final Cut Server administrator to specify that all metadata should be transferred to MatrixStore (KeepAllKeys field = true). Setting this option to true is not recommended though, as copying a vast amount of metadata to MatrixStore may later affect the performance of MatrixStore on Search operation. Instead, Final Cut Server administrator should choose which metadata fields have strategic meaning for his company's workflow and explicitly list them in the appropriate config file.

An exemple of asset config file looks as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>KeepAllKeys</key>
  <false/>
  <key>KeepXMLAsAttribute</key>
  <true/>
  <key>KeepKeys</key>
  <dict>
    <key>ASSET_NUMBER</key>
    <true/>
    <key>DB_ENTITY_ID</key>
    <true/>
    <key>CUST_TITLE</key>
    <true/>
    <key>PA_MD_CUST_FILENAME</key>
    <true/>
  </dict>
  <key>KeywordKeys</key>
  <dict>
    <key>CUST_KEYWORDS</key>
    <true/>
    <key>CUST_TITLE</key>
    <true/>
  </dict>
  <key>Version</key>
  <integer>1</integer>
</dict>
</plist>
```

This file may be edited or written with any text editor. User may want to create that file using OSX Property List Editor. In that case the one line referring to DTD validation document:

```
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://
www.apple.com/DTDs/PropertyList-1.0.dtd">
```

must be manually removed from created file.

The meaning of the keys in the config file is as follows:

### **KeepAllKey**

if true, plugin transfers all metadata associated with asset to MatrixStore. Those metadata are stored as key/value where key, value are Strings (UTF-8).  
Recommended value = false

### **KeepXMLAsAttribute**

if true plugin will keep all metadata associated with an asset as additional attribute kept with object on MatrixStore.  
Recommended value = true

## KeepKeys

List of Final Cut Server Metadata Fields (Field ID) which should be copied from Final Cut Server to MatrixStore on archive operation.

Those fields will be used to search using key=value syntax in Dropspot

Recommended value:

At least one field: ASSET\_NUMBER with flag true should be used, to find required asset on MatrixStore

## KeywordKeys

List of Final Cut Server Metadata Fields (Field ID) which should be transferred copied from Final Cut Server to MatrixStore on archive operation.

Those fields will be used to search using keywords syntax in Dropspot

Recommended value:

CUST\_KEYWORDS, CUST\_TITLE equal true

MatrixStore FCS plugin creates list of keywords from specified fields using as separator following characters:

*\_-./()& tab newline space*

Minimum length of a keyword is 3 characters. For example, FCS metadata field: CUST\_TITLE First advertisement by Object-Matrix about Matrix Store

will create the following keywords on MatrixStore archive device:  
first advertisement object matrix about store

Configuration file for archiving and tagging productions has exactly the same syntax. The meaning of fields is the same as for asset configuration file but is related to metadata associated with project (production) created in Final Cut Server.

Assets and productions archived on MatrixStore are tagged with additional metadata in order to find them easily using Dropspot. This metadata is:

**MXS\_FCS\_PRODUCTION\_ID** - production identifier for example /project/2  
**MXS\_FCS\_PRODUCTION\_CUST\_TITLE** - production custom title e.g. Project one  
**MXS\_FCS\_ASSET\_ID** - asset identifier in FCS e.g. /asset/28  
**MXS\_FCS\_ORIGINAL\_MEDIA** - path to original file (primary representation) e.g.  
/dev/4/Picture\_23.png

You can use those keys to filter your assets in Dropspot for example:

MXS\_FCS\_PRODUCTION\_ID=/project/2

or

MXS\_FCS\_PRODUCTION\_CUST\_TITLE="project one"

## Usage

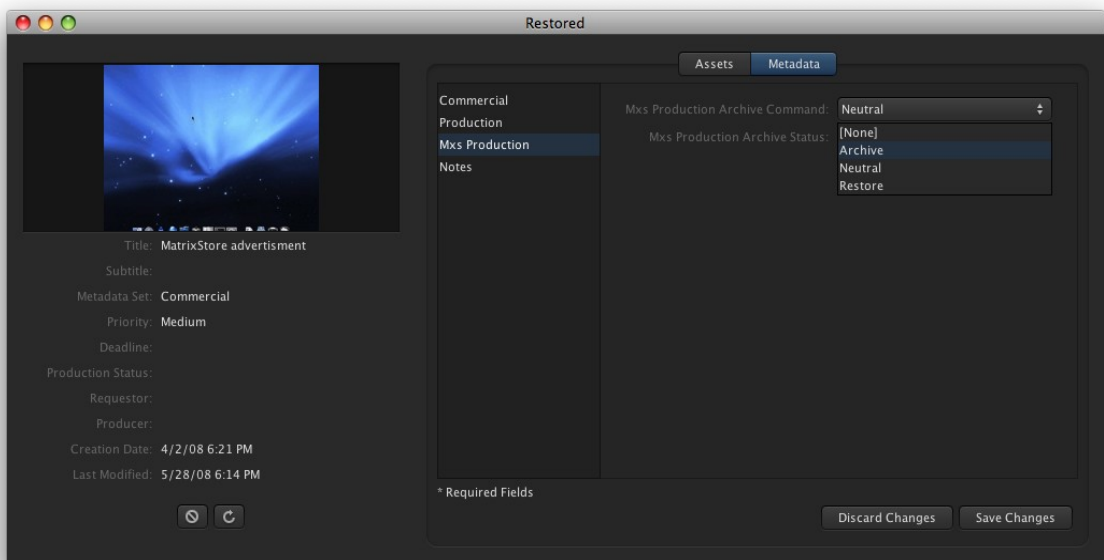
Final Cut Server Administrators can archive a single asset to MatrixStore archive device by using any of standard methods in Final Cut Server.

An archived asset is tagged with specified metadata fields and keywords automatically. There is no additional step user have to perform.

Archiving a FCS production can be done by listing information about production. From the new Metadata Group 'Mxs Production' user can change the Command Metadata field to Archive as shown in the picture below.

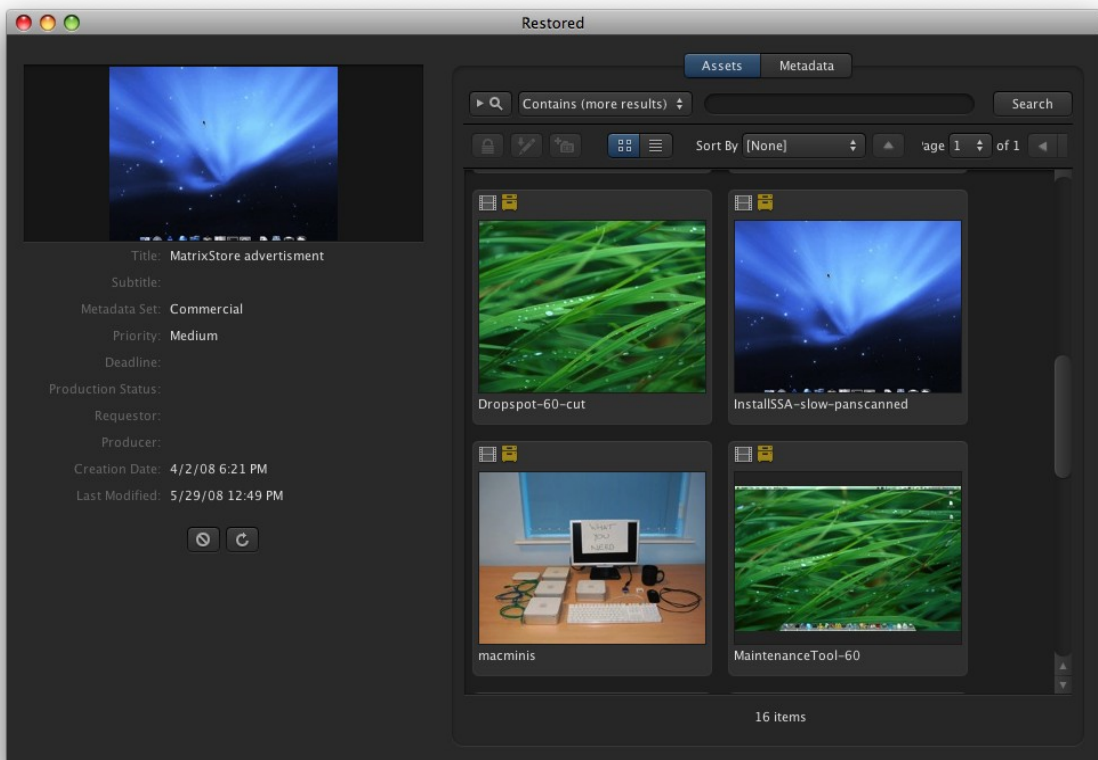
Saving updated metadata triggers a response, which archives all assets belonging to that production and tag them with metadata fields specified in the config file.

A production object on MatrixStore is also tagged with relevant metadata.

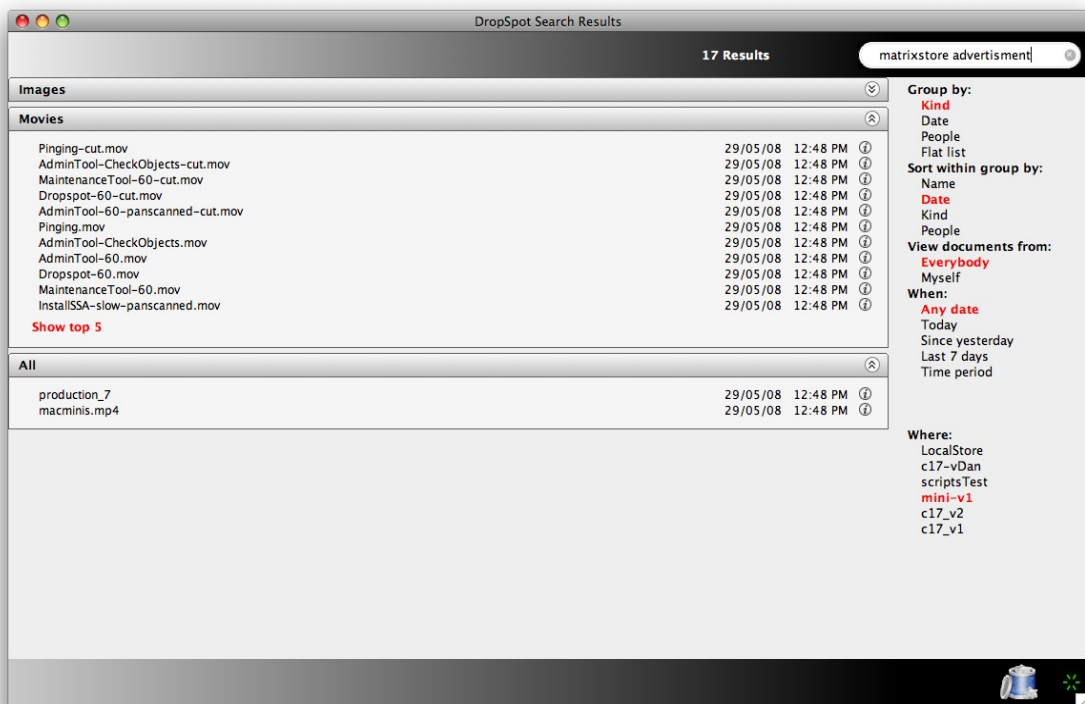


Finished archiving action is signalled by changing 'Mxs Production Archive Status' to

'Archived'. At this point all assets are stored on MatrixStore device and are marked as Archived in Final Cut Server as shown in the picture below.



Administrators can search for archived elements on MatrixStore using Dropspot as shown below.



Restoring a production can be done by changing 'Mxs Production Archive Command' to 'Restore'

## **Un-install**

To un-install MatrixStore Plugins for Final Cut Server, just remove the directory:

/Library/Application Support/MatrixStore/FCS Support

# Appendix

Example archive-production.vault file:

```
#Tue Apr 29 16:54:17 BST 2008
cluster=00000000-0000-0000-0000-000000000000
user=test-user
password=pass0
vault=test-vault
addresses=localsnode
vaultName=LocalStore
```

Example asset configuration file: archive\_asset\_config.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>KeepAllKeys</key>
  <false/>
  <key>KeepXMLAsAttribute</key>
  <true/>
  <key>KeepKeys</key>
  <dict>
    <key>ASSET_NUMBER</key>
    <true/>
    <key>DB_ENTITY_ID</key>
    <true/>
    <key>CUST_TITLE</key>
    <true/>
    <key>PA_MD_CUST_FILENAME</key>
    <true/>
  </dict>
  <key>KeywordKeys</key>
  <dict>
    <key>CUST_KEYWORDS</key>
    <true/>
    <key>CUST_TITLE</key>
    <true/>
  </dict>
  <key>Version</key>
  <integer>1</integer>
</dict>
</plist>
```

## Example production configuration file: archive\_production\_config.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>KeepAllKeys</key>
  <false/>
  <key>KeepXMLAsAttribute</key>
  <true/>
  <key>KeepKeys</key>
  <dict>
    <key>DB_ENTITY_ID</key>
    <true/>
    <key>CUST_TITLE</key>
    <true/>
    <key>PROJECT_TYPE</key>
    <true/>
    <key>MD_LAST_MODIFIED</key>
    <true/>
  </dict>
  <key>KeywordKeys</key>
  <dict>
    <key>CUST_TITLE</key>
    <true/>
    <key>PROJECT_TYPE</key>
    <true/>
  </dict>
  <key>Version</key>
  <integer>1</integer>
</dict>
</plist>
```