FastScan for Lustre Documentation

Miria 4.0

Publication Number: MIRIA-LUS-PDF-EN-1122-REV1

Publication Date: November 2022

Atempo SAS
23 Avenue Carnot
91300 Massy - France

Contents

CHAPTER 1 - About FastScan for Lustre

Miria allows you to perform incremental backups. These backups contain only the files modified since the last backup. In order to know these files, Miria must scan the entire file system and find the files whose modification date has changed.

In order to avoid having Miria do this job, you can install FastScan for Lustre.

Lustre offers a file logging feature, listing all creation, modification or deletion events taking place on the system.

By adding a FastScan for Lustre server to Miria, this log is read continuously and stores the list of modified files in a database. The list of all modified files is created without Miria needing to browse the entire system. When the time comes, the FastScan for Lustre server can return this list to Miria.

In this document, FastScan for Lustre will be referred to as FSL.

CHAPTER 2 - Architecture

The FSL server is organized around a database that permanently stores the list of modified files. In addition to this database, it has three main modules:

- an fsld service, running continuously, in the background. It is in charge of reading the changelogs and storing the list of modified files in the database.
- an fslweb client interface which allows, via an HTTP API, to interact with the service, to restore the list to the application and to monitor the status of the service.
- an fslc client that offers the same services as the web interface, but on command line.

The image (Figure 1) shows you the architecture of the FSL server and the three modules:

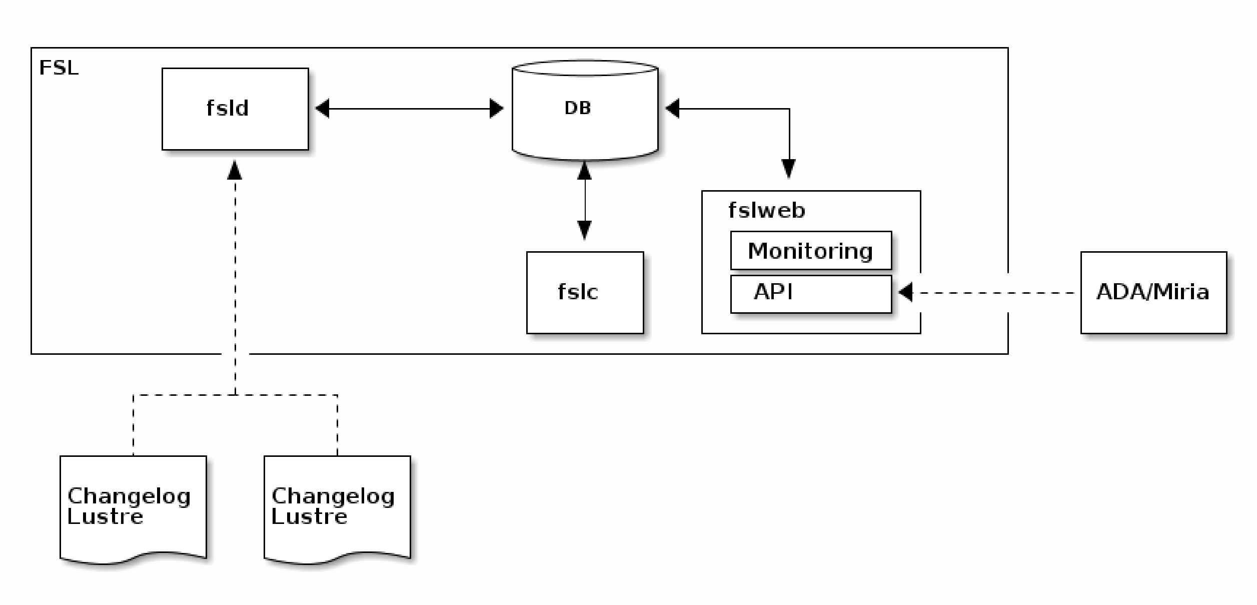


Figure 1: FSL server architecture

The centrality of the database in this architecture ensures:

- the isolation of concurrent reads and writes. This allows the database to continue to fill while a list of changes is generated.
- Failover because change log events are deleted only if they exist in the database.

The fsld service is controlled by the creation of audit points in the database. Each audit point is attached to a directory in the filesystem. If at least one audit point exists on a directory, then all changes affecting the files contained in this directory are recorded in the database. This avoids storing all the system activity, and in particular the directories that are not intended to be saved (scratch directories for example). The comparison between two of these audit points allows the generation of the changelist.

CHAPTER 3 - Install FSL server

The FSL server is intended to be installed on a Lustre client, i.e. a machine with access to the Lustre file system. The communication with Miria datamovers is done via the network. Please note the following:

- An FSL server can cover only one Lustre file system.
- It is not recommended to install the engine on the same machine as the Miria datamover, although it is possible.
- An FSL server supports Linux RedHat and Centos 7 in x86 64bits.

Prerequisites

Before installing FSL server, check the prerequisites in the sections below.

Server specifications

If you install an FSL server, it is recommended to have:

- 1 CPU with 12 cores.
- 64 GB of memory, mainly for the database.
- 1 GB for the distribution of the product.
- 200 GB of disk space (sufficient to store 1 billion objects in the database).

Operating system

GNU/Linux distributions using RPM (Red Hat, CentOS, Fedora, etc.)

FastScan for Lustre versions

Version 2.12.3, 2.12.4, 2.12.5, 2.12.6, 2.12.7, 2.12.8, 2.14.0 of open source Lustre are supported.

Installation prerequisite

Set up a NTP

Date and time between the FSL server and MDT/MGT must be kept accurate. The Lustre file system uses client clocks for timestamps. When not synchronized correctly, files will have different timestamps. To avoid any sync problems, it is recommended to set up a Network Time Protocol (NTP) server before installing Lustre. For more information, see also https://www.ntp.org/.

Special package

The use of RPM allows you to take advantage of all the packages available for these platforms. However, some dependencies require the `epel-release` repository to be enabled. On Centos7, activation can be done by installing a special package:

```
$ yum install epel-release
```

Port to open

Open port 5000 on the FSL server to be able to access the fslweb client.

Install Neo4j

The Neo4j database requires at least Java version 11. To install and activate Java, run:

```
$ yum install java-11-openjdk
$ alternatives --config java
  ## and select java-11-openjdk
```

The Neo4j database must be installed via an external repository. To configure it you need to:

1. Add the official Neo4j repository. See the Neo4j documentation for more information.

```
$ rpm --import https://debian.neo4j.com/neotechnology.gpg.key
$ cat <<EOF> /etc/yum.repos.d/neo4j.repo
  [neo4j]
  name=Neo4j RPM Repository
  baseurl=https://yum.neo4j.com/stable
  enabled=1
  gpgcheck=1
  EOF
```

2. Install the neo4j package:

```
$ yum install neo4j
```

3. Uncomment the following line in the configuration file `/etc/neo4j/neo4j.conf` to remove the authentication:

```
dbms.security.auth_enabled=false
```

4. Restart the Neo4j database:

```
$ systemctl restart neo4j
```

Install FSL RPM

The FSL RPM package is part of the Miria server binary. It contains:

- The Python fsl modules.
- The fslc client executable.
- The necessary configuration files for the `systemd fsld` and `fslweb` services.
- The Atempo registry file `/etc/Atempo/FSL/env.ib`.
- An `advanced_env_report.sh` collector.

To install it, run:

```
curl -O -k https://<miriaserver>/fsl_current_x86_64.rpm
```

Where `<miriaserver>` is the hostname or IP address of the Miria server.

We recommend that you update RPM FSL as soon as you change your Miria server version.

You can upgrade or remove the FSL RPM:

To upgrade:

1. Stop the fsld and fslweb service:
   - $ systemctl stop fsld
   - $ systemctl stop fslweb
2. Run: `yum localinstall fsl_current_x86_64.rpm`

To remove:

1. As you remove the FSL package, you have to stop the service before uninstalling it:
   - $ systemctl stop fsld
   - $ systemctl stop fslweb
2. Run: `rpm -e fsl.x86_64`

CHAPTER 4 - Configure FSL Server

The configuration of your FSL server involves activating changelogs and configuring the FSL. This chapter explains how to carry out these steps.

Activate Changelogs

For more details about changelogs, please refer to the Lustre documentation (Chapter 12.1), and to the Setting Changelogs Mask Appendix.
Lustre is a network file system. It stores metadata on devices (named MDT) attached to one or more servers (named MDS). The diagram (Figure 2) shows this architecture:

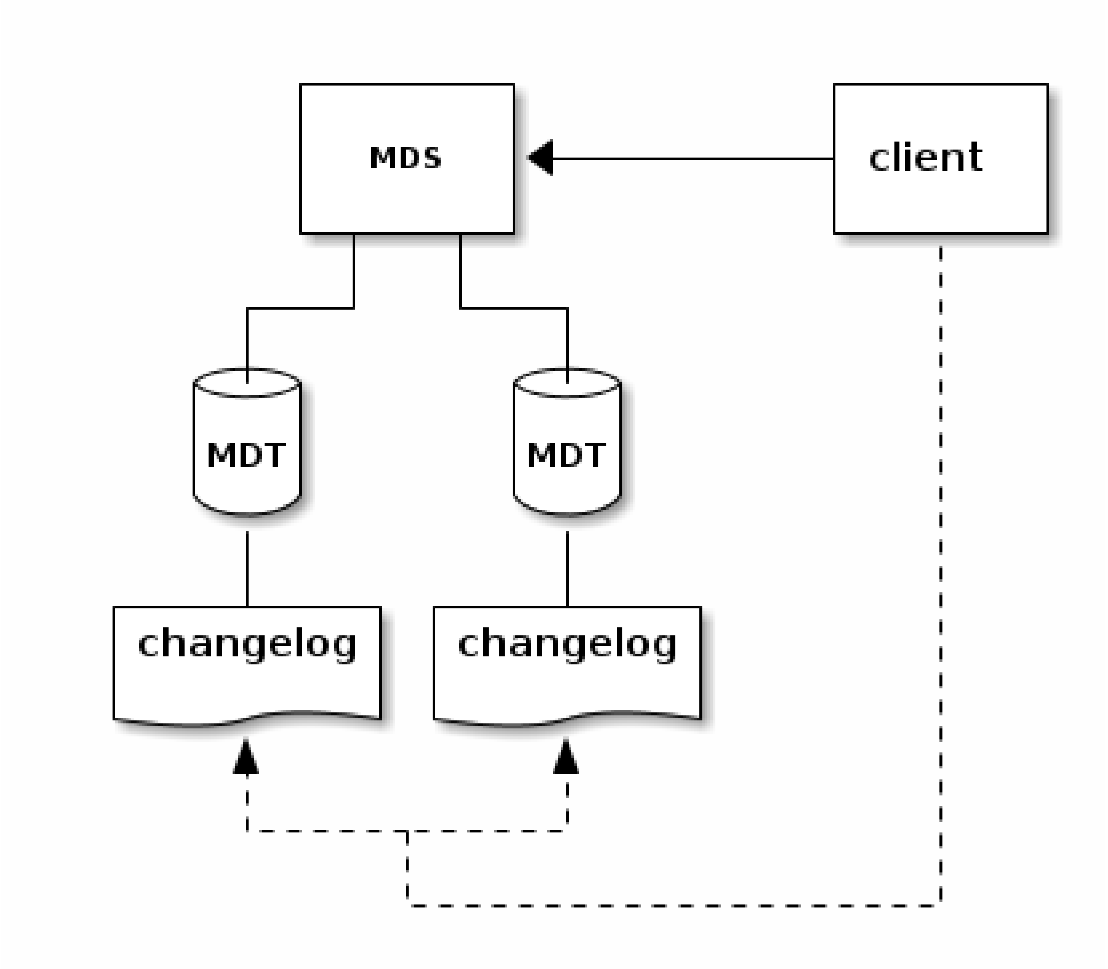


Figure 2: Lustre file system architecture

Each MDT is identified by a name, composed as follows: `<Lustre name>- MDT<number>`.

The procedure below will take as an example a Lustre system named `ai200`. The first MDT will be `ai200- MDT0000`, the second `ai200-MDT0001`, etc.

To activate the changelogs you must:

1. Run the following command on a Lustre client, in order to retrieve the address of the MDS to which an MDT is attached.

```
$ lctl get_param mdc.*.mds_conn_uuid
mdc.ai200-MDT0000-mdc-ffff944af46e5000.mds_conn_uuid=192.168.3.1@o2ib
mdc.ai200-MDT0001-mdc-ffff944af46e5000.mds_conn_uuid=192.168.3.1@o2ib
mdc.ai200-MDT0002-mdc-ffff944af46e5000.mds_conn_uuid=192.168.3.3@o2ib
mdc.ai200-MDT0003-mdc-ffff944af46e5000.mds_conn_uuid=192.168.3.3@o2ib
```

2. Enable Lustre changelogs on each MDS by registering one user per changelog.

> **Example**: To activate `ai200-MDT0000`, the following command must be executed on the concerned MDS (here identified by the IP address 192.168.3.1):

```
$ lctl --device ai200-MDT0000 changelog_register
ai200-MDT0000: Registered changelog userid 'cl4'
```

The return of the command indicates that a new user of changelog cl4 has been created.

3. Run the following command on the MDS to obtain the list of users:

```
$ lctl get_param mdd.*.changelog_users
mdd.ai200-MDT0000.changelog_users=
current index: 50653643
ID index (idle seconds)
cl4 50652755 (83203)
cl5 50653643 (151)
mdd.ai200-MDT0001.changelog_users=
current index: 1146655
ID index (idle seconds)
cl2 1146655 (420958)
```

4. Create users specifically for the FSL server. This enables the system to identify records that have not yet been read by all clients and those that can be deleted.

Enabling changelogs causes special devices `/dev/changelog-<mdtname>` to appear directly on the client. For example, here are the special devices for the `ai200 system`, with four changelogs enabled:

```
$ ls -l /dev/changelog*
crw-------. 1 root root 10, 52 Aug 11 13:49 /dev/changelog-ai200-MDT0000
crw-------. 1 root root 10, 54 Aug 11 13:49 /dev/changelog-ai200-MDT0001
crw-------. 1 root root 10, 53 Aug 11 13:49 /dev/changelog-ai200-MDT0002
crw-------. 1 root root 10, 51 Aug 11 13:49 /dev/changelog-ai200-MDT0003
```

Configure FSL

1. Go to the file `/etc/Atempo/FSL/fsl.conf` in INI format (i.e. key=value, grouped in sections).
2. Configure the following variables in the `[Lustre]` section:
   - `mountpoint` Indicates the mount point of the Lustre file system on the client, recoverable for example with the `df` command;

- the list of MDTs associated with the changelog readers, obtained when activating the `changelogs` on the MDS. The MDT list items must always be in this form `mdt<number> = name, changelog reader`.

> **Example**: For a Lustre filesystem mounted on `/lustre/ai200/client`, with two MDTs named `ai200-MDT0000` and `ai200-MDT0001`, activated for cl3 and cl1 readers respectively, the section is as follows:

```
[lustre]
mountpoint = /lustre/ai200/client
# map {MDT name, changelog reader}
mdt0 = ai200-MDT0000,cl3
mdt1 = ai200-MDT0001,cl1
```

3. Restart the fsld and fslweb service to make the new configuration work:

```
$ systemctl restart fsld
$ systemctl restart fslweb
```

To display the current configuration explicitly (including default values), run the `fslc config_show` client command. (see the Use the fslc Client for more information). For exemple:

```
$ fslc config_show
[lustre]
mountpoint = /lustre/ai200/client
mdt0 = ai200-MDT0000,cl3
mdt1 = ai200-MDT0001,cl1

[fsl]
cl_clear = 1000
debug_level = 1
debug_level_fsld = 2
clean_enable = yes
clean_batch_size = 10000
clean_log_min_duration = 250
```

The parameters in section `[fsl]` are as follows :

- `cl_clear`: number of records at the end of which fsl will try to clean lustre changelog. Default is 1000.
- `debug_level`: controls the level of logging of client commands, i.e. the amount of information reported by these commands. Can vary between 0 and 3. 1 by default and 3 the highest level of logging.
- `debug_level_fsld`: logging level specific to the fsld service. Default is 2.
- `clean_enable`: controls the automatic database cleaning mechanism. Yes by default, No to disable this mechanism.
- `clean_batch_size`: controls the size of the cleanup request. 10000 by default. If the value is high, each cleanup request takes time, but fewer requests are needed to completely clean the database. A positive number is expected.
- `clean_log_min_duration`: if the cleaning request exceeds this value in ms, a warning is added to the log. 250 by default. A value of 0 will result in a warning for each cleaning request.

Control and Monitoring of the fsld Service

As fsld is a systemd service. It can be controlled by the usual `systemctl` commands (refer to your system documentation for more details). The following commands allows you to start, stop and restart it:

```
$ systemctl start fsld
systemctl stop fsld
systemctl restart fsld
```

The `status` sub-command is used to query the status :

```
systemctl status fsld
```

The logs associated with the service can be viewed with the command `journalctl` :

```
$ journalctl -eu fsld
```

The fsld service depends on the Neo4j database. This dependency is managed by `systemd` on behalf of the user. This means that:

- starting the fsld service results in automatically start Neo4j.
- stopping the database automatically causes the stop of fsld.

Furthermore, it is recommended to configure fsld to start automatically at system startup, which can be done with the command systemctl enable.

Be careful however, the activation of the Neo4j database is not automatic in this case, and must be done explicitly:

```
$ systemctl enable neo4j
$ systemctl enable fsld
```

CHAPTER 5 - Use the fslc Client

The use of the command line client is documented by the `--help` option or with one of the sub-commands:

```
usage: fslc [-h] [--version]
    {create,delete,list,changelist,config_show,password_set,status,aer,fid2path
    ...

fsl client: interact with the Lustre fastscan system

positional arguments:
    {create,delete,list,changelist,config_show,password_set,status,aer,fid2path,dir_fix
        create Create an audit point
        delete Delete an audit point
        list List audit points
        changelist List files modified between two audit points
        config_show List fsl configuration values
        password_set Update fsl web interface password
        status Miscellaneous fsl parameters and database information
        aer List fsl variables of interest to the AER collectors
        fid2path Translate a Lustre FID to a path using the fsl db
        dir_fix Try to fix the FSL private directory permissions

optional arguments:
    -h, --help show this help message and exit
    --version show program's version number and exit
```

Create an Audit Point

The `create` sub-command allows you to create an audit point. Each audit is associated to a unique identifier (uid) used to manipulate it, and to a directory. If specified during creation, the path of the monitored directory must be relative to the Lustre mount point and start with a `/`.

```
$ fslc create --path /example
uid 921c1087-61d9-4f82-a3b6-c20816e1bdd9
path /example
datecrea 2020-09-28T14:03:51.384457+00:00
```

List of Audit Points

The sub-command `lists` allows you to generate a list of system audit points.

Here is an example list of the audit points:

```
$ fslc list
uid             c63a948e-8d40-4336-97e0-3b7cbd66488f
path             /test/n2
datecrea         2020-09-25T17:33:52.106668+00:00
file_count         0
```

```
uid             d94d8e84-f213-48e1-8f9a-8a451807eca9
path             /test2
datecrea         2020-09-28T09:28:35.730526+00:00
file_count         131104

uid             921c1087-61d9-4f82-a3b6-c20816e1bdd9
path             /example
datecrea         2020-09-28T14:03:51.384457+00:00
file_count         65552

uid             7eca93c3-582b-4813-a0a8-1d7476fdc96e
path             /example
datecrea         2020-09-28T14:11:47.173012+00:00
file_count         0
```

In this example, the audit point previously created on the example directory is present in the penultimate position. It is associated with 65,552 files, which means that a number of events have taken place in the meantime. A second audit point has also been created, on the same directory.

You can also filter the list by directory:

```
$ fslc list --path /example
uid                 921c1087-61d9-4f82-a3b6-c20816e1bdd9
path                 /example
datecrea             2020-09-28T14:03:51.384457+00:00
file_count             65552

uid                 7eca93c3-582b-4813-a0a8-1d7476fdc96e
path                 /example
datecrea             2020-09-28T14:11:47.173012+00:00
file_count             0
```

List of Modified Files Between Two Audit Points

The `changelist` sub-command enables to generate a text file containing the list of modified files between two audit points identified by their uid:

```
$ fslc changelist 921c1087-61d9-4f82-a3b6-c20816e1bdd9 7eca93c3-582b-4813-a0a8-1d7476fdc96e
path                           /lustre/ai200/client/.fsl/5e9adc88-0195-11eb-830e-
801844ec1c3d
```

This command may take some time, roughly proportional to the number of modified files. The generated file is written in a hidden directory (by default .fsl) at the root of the Lustre mount point (here `/lustre/ai200/client`). This directory can be configured by setting private_dir in section [fsl] in /etc/Atempo/FSL/fsl.conf. This file can be read directly by the backup application.

Delete an Audit Point

The `delete` sub-command deletes an audit point identified by its uid.

```
$ fslc delete 7eca93c3-582b-4813-a0a8-1d7476fdc96e
uid           7eca93c3-582b-4813-a0a8-1d7476fdc96e
```

Other Features

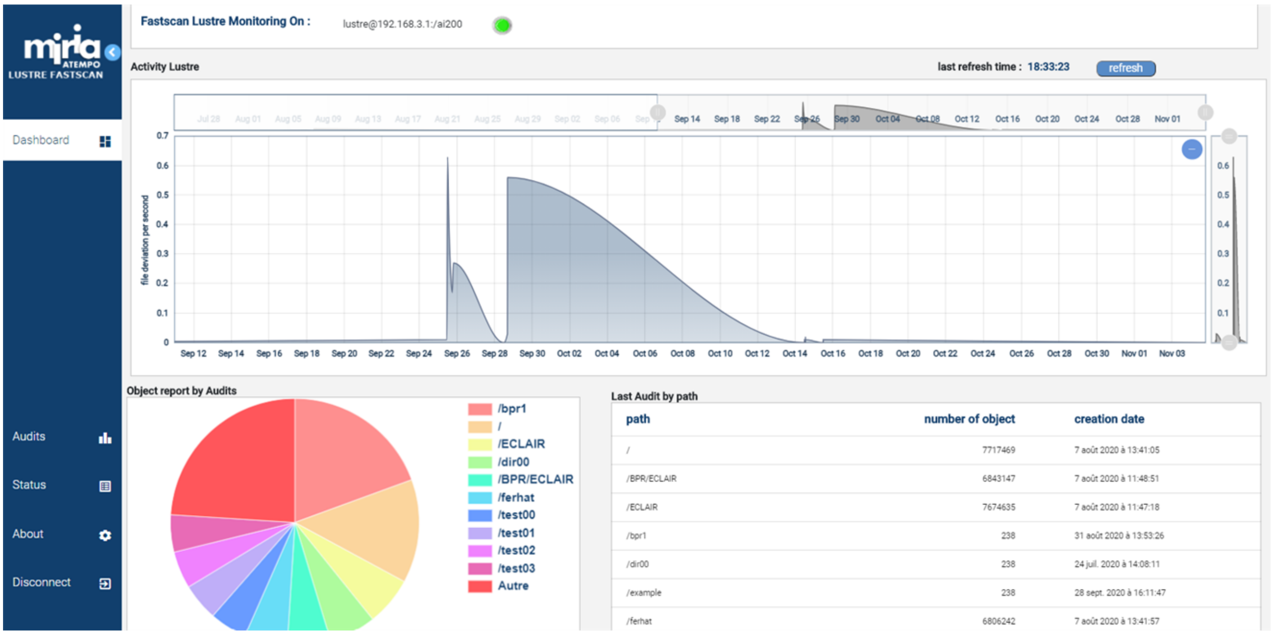
- The `aer` subcommand generates a list of parameters intended to be used by AER external collectors.
- The `config_show` subcommand displays the list of configurable parameters and their values in INI format and can therefore be directly copied into the configuration file.
- The `password_set` sub-command allows users of the web interface to change the password.
- The `dir_fix` sub-command allows you to check and possibly correct the permissions of the FSL private directory.

CHAPTER 6 - Use fslweb Client

The fslweb web client is used to provide:

- The HTTP API for Miria.
- A monitoring interface:
  - available on port 5000
  - at: `http://<fslserver>:5000/front` (where `<fslserver>` is the hostname or IP address of the server hosting the fsl engine)
  - default login: api
  - default password: apifsl

The interface (Figure 3) allows to see information about the activity and to know if your Fastscan server is working well.


miria ATEMPO
LUSTRE FASTSCAN
Dashboard
Audits
Status
About
Disconnect
Fastscan Lustre Monitoring On : lustre@192.168.3.1:/ai200
Activity Lustre
last refresh time : 18:33:23
refresh
file deviation per second
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Sep 12 Sep 14 Sep 16 Sep 18 Sep 20 Sep 22 Sep 24 Sep 26 Sep 28 Sep 30 Oct 02 Oct 04 Oct 06 Oct 08 Oct 10 Oct 12 Oct 14 Oct 16 Oct 18 Oct 20 Oct 22 Oct 24 Oct 26 Oct 28 Oct 30 Nov 01 Nov 03
Object report by Audits
/bpr1
/
/ECLAIR
/dir00
/BPR/ECLAIR
/ferhat
/test00
/test01
/test02
/test03
Autre
Last Audit by path

path	number of object	creation date
/	7717469	7 août 2020 à 13:41:05
/BPR/ECLAIR	6843147	7 août 2020 à 11:48:51
/ECLAIR	7674635	7 août 2020 à 11:47:18
/bpr1	238	31 août 2020 à 13:53:26
/dir00	238	24 juil. 2020 à 14:08:11
/example	238	28 sept. 2020 à 16:11:47
/ferhat	6806242	7 août 2020 à 13:41:57

Figure 3: The fslweb client dashboard

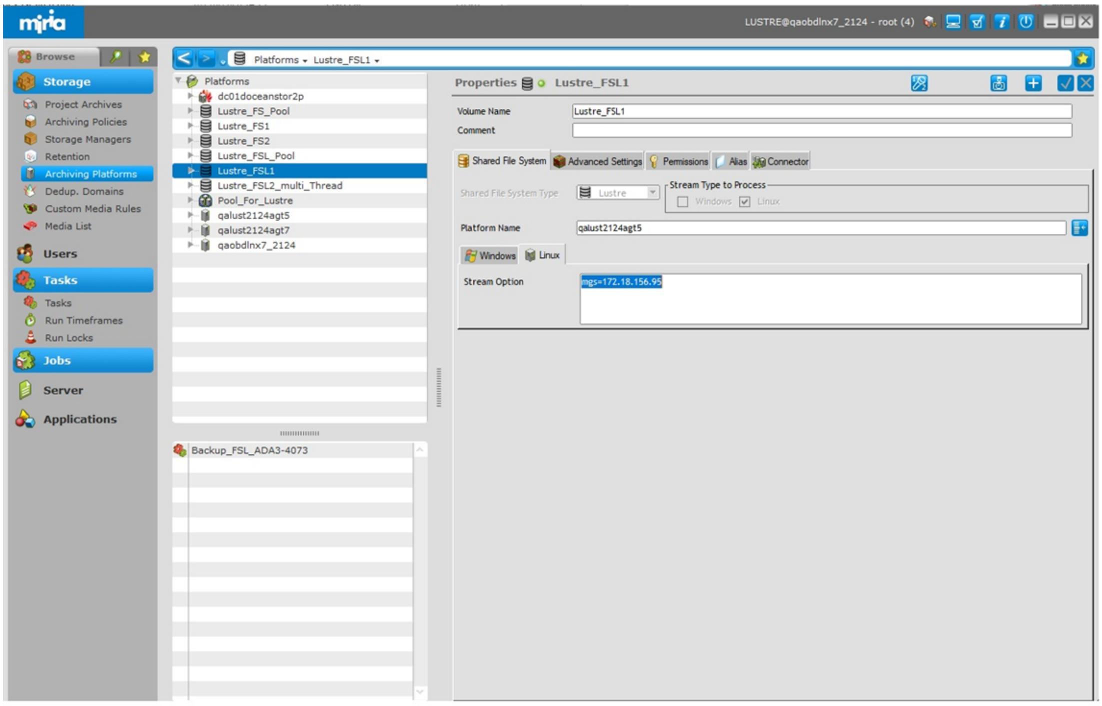
It is composed of:

- A dashboard including:
  - A visualization on the MDT activity on the product.
  - Object reported by audit.
  - Information on the last audits created.
- An Audits page to see the list of all audits and to make a search.
- An About page to have status information: Fastscan engine, Lustre informations, MDT list with record number, Fastscan database information.

CHAPTER 7 - Configure FSL in Miria

Once you have installed and configured the FSL server, you need to configure it in Miria's admin console:

1. From the left pane, select **Browse > Storage > Archiving Platform**.
2. Click the + at the top right and select **New Platform**.
3. Select the **Shared file system** tab.
4. In the **Volume Name** field, enter a name for the new shared file system. The name must not contain any spaces.
5. In the **Shared file system type** field, select Lustre.
6. Click the check mark to validate the shared file system creation.
7. Select the newly-created volume.
8. Complete the **Shared File System** tab:
    a. In the **Platform Name** field, select the agent that hosts the Lustre volume.
    b. Add a Linux **Stream Option**.
9. Complete the **Connector** tab:
    a. Enable the connector.
    b. Enter the Network Adress.
    c. Enter username and password. By default api and apifsl.
    d. Enable FastScan.
10. Click the check mark to validate the Lustre creation.



Volume Name: Lustre_FSL1

Comment:

Shared File System Type: Lustre

Stream Type to Process: □ Windows ☑ Linux

Platform Name: qalust2124agt5

Stream Option: mgs=172.18.156.95

APPENDIX Setting Changelogs Mask

This table describes the Changelogs Mask that are required in order to configure your FSL server and their use (Table 1):

Table 1: Minimum Changelogs Mask

Value	Description
CLOSE	Close
CREAT	Regular file creation
CTIME	CTIME change
FLRW	File Level Replication: file initially written
HLINK	Hard link
HSM	HSM specific event
LYOUT	Layout change
MARK	Internal recordkeeping
MIGRT	Migration event
MKDIR	Directory creation
MKNOD	Other file creation
MTIME	MTIME change
RENME	Rename, original
RESYNC	File Level Replication: file re-synced
RMDIR	Directory removal
RNMTO	Rename, final
SATTR	Attribute change
SLINK	Soft link
TRUNC	Regular file truncated
UNLNK	Regular file removal
XATTR	Extended attribute change (setxattr)