1. GridGain In-Memory Accelerator For Hadoop®

GridGain's `In-Memory Accelerator For Hadoop` edition is based on the industry's first high-performance dual-mode in-memory file system that is 100% compatible with HDFS. GridGain File System (GGFS) is a plug-and-play alternative to the disk-based Hadoop HDFS enabling up to 10x faster performance for IO and network intensive Hadoop MapReduce jobs running on any Hadoop distribution.

2. Hadoop Installation

> This section describes **absolute minimal** steps on how to install and start **Hadoop only**. GridGain installation and integration instructions will come in the following sections.

If you already have Hadoop installed, skip this section. The instructions below assume:

- Linux or MacOS environment
- Hadoop 1.x or Hadoop 2.x distro
- Java 7 (latest update is recommended)

Before installing and starting Hadoop, check that you can login to localhost via SSH without passphrase:

```
ssh localhost
```

If *passphrase* is requested, generate new SSH key (don't forget to backup the existing one):

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
```

And add this key to authorized keys list:

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

Now you should be able to login without *passphrase*.

2.1 Hadoop 1.x Installation

Perform the following steps:

1. Download Hadoop 1.0.4 (or the latest version) from http://www.us.apache.org/dist/hadoop/common
2. Unzip archive or install package.
3. Open `conf/hadoop-env.sh` file and add following line to set correct `JAVA_HOME` path:

```
export JAVA_HOME="your/java/home/path"
```

4. Open `conf/core-site.xml` file and add following lines inside `configuration` tag:

```
<property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
</property>
```

5. If you need MapReduce functionality, open `conf/mapred-site.xml` file and add following lines inside `configuration` tag:

```
<property>
    <name>mapred.job.tracker</name>
    <value>localhost:9001</value>
</property>
```

6. Go to Hadoop `bin` folder and run `./hadoop namenode -format` command to make initial formatting of name node folder.

7. Run `start-all.sh` script which will run name node, data node, job tracker and task tracker locally. If you don't need MapReduce functionality, use `start-dfs.sh` script.

For more information and troubleshooting refer to Hadoop documentation: http://hadoop.apache.org/docs/r1.0.4/.

2.2 Hadoop 2.x Installation

Perform the following steps:

1. Download Hadoop 2.0.3 (or the latest version) from http://www.us.apache.org/dist/hadoop/common
2. Unzip archive or install package.
3. Open `etc/hadoop/hadoop-env.sh` file and add following line to set correct `JAVA_HOME` path:

```
export JAVA_HOME="your/java/home/path"
```

4. Open `etc/hadoop/core-site.xml` file and add following lines inside `configuration` tag:

```
<property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
</property>
```

5. If you need MapReduce functionality open `etc/hadoop/mapred-site.xml` file. If this file does not exist, then create it as a copy of `mapred-site.xml.template` file in the same folder. Add following lines inside `configuration` tag:

```
<property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
</property>
```

   Open `etc/hadoop/yarn-site.xml` file and add following lines inside `configuration` tag:

```
<property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>localhost:8031</value>
</property>
<property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>localhost:8030</value>
</property>
<property>
    <name>yarn.resourcemanager.address</name>
    <value>localhost:8032</value>
</property>
<property>
    <name>yarn.nodemanager.address</name>
    <value>0.0.0.0:8042</value>
</property>
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce.shuffle</value>
</property>
```

6. Go to Hadoop `bin` folder and run `./hadoop namenode -format` command to make initial formatting of name node folder.
7. Go to Hadoop `sbin` folder and run `./start-dfs.sh` command to start HDFS and `./start-yarn.sh` to start MapReduce.

For more information and troubleshooting refer to Hadoop documentation: http://hadoop.apache.org/docs/r2.0.3-alpha/

2.3 Troubleshooting

2.3.1 File <..> could only be replicated to 0 nodes, instead of 1

When running benchmarks, you can get the following error:

```
DataStreamer Exception: org.apache.hadoop.ipc.RemoteException: java.io.IOException: File <..> could only be
replicated to 0 nodes, instead of 1
```

In this case you should:

1. Stop whole HFDS cluster using `stop-dfs.sh` script.
2. Delete and recreate name node directory (by default it's `/tmp/<hadoop-username>/dfs/name`, but it can be customized with `dfs.name.dir` in `hdfs-site.xml` configuration file).
3. Format name node using `./hadoop namenode -format` command.
4. Restart the cluster.

2.3.2 Unable to load realm info from SCDynamicStore

When starting HFDS on Mac OS, you can get the following warning:

```
Unable to load realm info from SCDynamicStore
```

In this case you should add the following line to script `conf/hadoop-env.sh` in hadoop 1.x or `etc/hadoop/hadoop-env.sh` in hadoop 2.x:

```
export HADOOP_OPTS="$HADOOP_OPTS -Djava.security.krb5.realm= -Djava.security.krb5.kdc="
```

If running on JDK7, then need to specify JDK6 explicitly by adding the following:

```
export JAVA_HOME=`/usr/libexec/java_home -v 1.6`
```

2.3.3 java.net.UnknownHostException: unknown host: shmem

When starting HFDS, you can get the following error:

```
Exception in thread "main" java.net.UnknownHostException: unknown host: shmem
```

It this case your `HADOOP_CONF_DIR` environment variable is pointing to GGFS configuration directory. You need to set `HADOOP_CONF_DIR` to empty string and restart HFDS.

2.3.3 org.apache.hadoop.hdfs.server.namenode.NotReplicatedYetException: Not replicated yet

When when working with GGFS in `DUAL_SYNC` or `DUAL_ASYNC` modes, you can get the following exception:

```
org.apache.hadoop.ipc.RemoteException: org.apache.hadoop.hdfs.server.namenode.NotReplicatedYetException:
Not replicated yet
```

This exception indicates that reading is performed from a secondary file system, but corresponding file block was not replicated to this node (replication process is asynchronous). In this case you need either have some delay before reading file, or set replication factor to 1.

2.3.4 java.io.IOException: Failed to replace a bad datanode on the existing pipeline…

This is a known issue usually appearing on Hadoop 2.x installations: https://issues.apache.org/jira/browse/HDFS-4600 . If you get this issue, you should add the following to your `core-site.xml` file:

```
<property>
    <name>dfs.client.block.write.replace-datanode-on-failure.policy</name>
    <value>NEVER</value>
</property>
```

3. GridGain Installation

If you are reading this document you most likely already installed GridGain.

GridGain should be installed on all machines on which Hadoop is installed. GridGain distribution comes in a ZIP file that simply needs to be unzipped and `GRIDGAIN_HOME` environment variable should be set to point to `distro` folder under GridGain installation.

You can use any tools to perform this installation on multiple computers. There are no additional steps required for GridGain installation in such multi machine setup.

Installation requirements:

1. Windows, Linux, or MacOS environment.
2. Java 7 (latest update is advisable).
3. Point `JAVA_HOME` environment variable to your JDK or JRE installation.
4. Point `GRIDGAIN_HOME` environment variable to the `distro` folder under GridGain installation.

3.1 Check GridGain Installation

To verify GridGain installation, you can execute the GridGain startup script.

The following command will startup GridGain with default configuration using Multicast node discovery.

```
bin/ggstart.{sh|bat}
```

The following command will startup GridGain with default configuration using TCP node discovery for all nodes running on local host.

```
bin/ggstart.{sh|bat} config/tcp/spring-tcp-vm.xml
```

If GridGain was installed successfully, the output from above commands should produce no exceptions or errors. Note that you may see some warnings during startup, but this is OK as they are meant to inform that certain functionality is turned on or off by default.

You can execute the above commands multiple times on the same machine and make sure that nodes discover each other. Here is an example of log printout when 2 nodes join topology:

```
... Topology snapshot [nodes=2, CPUs=8, hash=0xD551B245]
```

You can also start GridGain Management Console, called Visor, and observe started nodes show up on Visor Dashboard. To startup Visor in GUI mode, you should execute the following script:

```
bin/ggvisorui.{sh|bat}
```

3.2 Running GridGain Examples

GridGain comes with many well documented examples. All examples have documentation about how they should be started and what the expected outcome should be.

> Use provided pom.xml to import examples into IDE of your choice.

3.3 Configure GridGain Node Discovery

When using TCP-based discovery make sure to update configuration for IP Finder with actual IP addresses like so:

```
<property name="discoverySpi">
    <bean class="org.gridgain.grid.spi.discovery.tcp.GridTcpDiscoverySpi">
        <property name="ipFinder">
            <bean class="org.gridgain.grid.spi.discovery.tcp.ipfinder.vm.GridTcpDiscoveryVmIpFinder">
                <property name="addresses">
                    <list>
                        <value>10.1.2.3:47500</value>
                        <value>10.1.2.4:47501</value>
                    </list>
                </property>
            </bean>
```

```
        </property>
    </bean>
</property>
```

On startup, GridGain node will try to connect to the specified IP addresses one-by-one until it succeeds.

> **NOTE:** you are only required to specify at least 1 IP address of the grid node that will be started first - other IP addresses are optional.

4. Integration with Hadoop

In addition to direct file system API, GGFS can be integrated into Hadoop so that you can use Hadoop HDFS command line scripts and MapReduce engine to work with data stored in GGFS on GridGain nodes.

GGFS can be integrated with Hadoop using 4 different modes:

- `PRIMARY` mode - In this mode GGFS serves as a *primary standalone* distributed in-memory file system
- `PROXY` mode - In this mode GGFS serves as a proxy which will *always delegate* to HDFS without caching anything in memory.
- `DUAL_SYNC` - In this mode GGFS will synchronously `read-through` from HDFS whenever data is requested and is not cached in memory, and *synchronously* `write-through` to HDFS whenever data is updated/created in GGFS. Essentially, in this case GGFS serves as an intelligent caching layer on top of HDFS.
- `DUAL_ASYNC` - In this mode GGFS will synchronously `read-through` from HDFS whenever data is requested and is not cached in memory (just like in `DUAL_SYNC` mode), and *asynchronously* `write-through` to HDFS whenever data is updated/created in GGFS. Since data is modified in HDFS *asynchronously*, there is a lag between GGFS updates and HDFS updates, however the performance of updates is significantly faster than using HDFS directly. Essentially, in this case GGFS again serves as an intelligent caching layer on top of HDFS.

You can configure GGFS mode by setting `GridGgfsConfiguration.getPathModes()` configuration property either in XML configuration or directly from code. By default GGFS runs in `PRIMARY` mode.

> **NOTE:** that in all modes other than `PRIMARY` mode HDFS data nodes must be started in tandem with GGFS data nodes. Refer to step (4.6) below for detailed instructions on Hadoop startup.

4.1 Integration with Hadoop 1.x

To integrate GGFS with Hadoop 1.x, do the following:

4.1.1 Create a separate config directory for GGFS

You will need separate configurations to start HFDS and Hadoop map-reduce with GGFS. Copy Hadoop `conf` configuration directory under Hadoop installation root to a new folder, called `conf-ggfs`. This step will separate HFDS configuration from GGFS configuration and will allow to start Hadoop with GGFS only, with HFDS only, or both with GGFS and HFDS.

4.1.2 Update `conf-ggfs/core-site.xml`

Configure GGFS implementation of Hadoop file system in `conf-ggfs/core-site.xml` file:

```
<property>
    <name>fs.ggfs.impl</name>
    <value>org.gridgain.grid.ggfs.hadoop.v1.GridGgfsHadoopFileSystem</value>
</property>
```

After GGFS implementation is registered you can use GGFS explicitly providing URI to GGFS path: `ggfs://ipc/path/to/my/file`. In this case all intermediate Hadoop files (including task jar files) will still be stored in HDFS, and only paths directly referenced as GGFS will be stored in GGFS.

If you want to set GGFS as a default file system implementation, add following property to `conf-ggfs/core-site.xml`:

```
<property>
    <name>fs.default.name</name>
    <value>ggfs://ipc</value>
```

```
</property>
```

4.1.3 Update Hadoop Classpath

Add GridGain JAR and all libraries it depends on to Hadoop classpath. To do this, add following lines to `conf-ggfs/hadoop-env.sh` script in Hadoop distribution (replace `GRIDGAIN_HOME` with correct path):

```
export GRIDGAIN_HOME=/path/to/GridGain/distribution
export HADOOP_CLASSPATH=$GRIDGAIN_HOME/gridgain*.jar

for f in $GRIDGAIN_HOME/libs/*.jar; do
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f;
done
```

4.2 Integration with Hadoop 2.x

To integrate with Hadoop 2.x, do the following:

4.2.1 Create a separate config directory for GGFS

You will need separate configurations to start HFDS and Hadoop map-reduce with GGFS. Copy Hadoop `etc/hadoop` configuration directory under Hadoop installation root to a new folder, called `etc/hadoop-ggfs`. This step will separate HFDS configuration from GGFS configuration and will allow to start Hadoop with GGFS only, with HFDS only, or both with GGFS and HFDS.

4.2.2 Update `etc/hadoop-ggfs/core-site.xml`

Configure GGFS implementation of Hadoop file system in `etc/hadoop-ggfs/core-site.xml` file:

```
<property>
    <name>fs.ggfs.impl</name>
    <value>org.gridgain.grid.ggfs.hadoop.v1.GridGgfsHadoopFileSystem</value>
</property>
<property>
    <name>fs.AbstractFileSystem.ggfs.impl</name>
    <value>org.gridgain.grid.ggfs.hadoop.v2.GridGgfsHadoopFileSystem</value>
</property>
```

Note that both Hadoop 1.x and Hadoop 2.x entries should be added to configuration because Hadoop command line interface uses Hadoop 1.x File System API.

After GGFS implementation is registered you can use GGFS explicitly providing URI to GGFS path: `ggfs://ipc/path/to/my/file`. In this case all intermediate Hadoop files (including task jar files) will still be stored in HDFS, and only paths directly referenced as GGFS will be stored in GGFS.

If you want to set GGFS as a default file system implementation, add following property to `conf-ggfs/core-site.xml`:

```
<property>
    <name>fs.default.name</name>
    <value>ggfs://ipc</value>
</property>
```

4.2.3 Update Hadoop Classpath

Add GridGain JAR and all libraries it depends on to Hadoop classpath. To do this, add following lines to `etc/hadoop-ggfs/hadoop-env.sh` script in Hadoop distribution (replace `GRIDGAIN_HOME` with correct path):

```
export GRIDGAIN_HOME=/path/to/GridGain/distribution
export HADOOP_CLASSPATH=$GRIDGAIN_HOME/gridgain*.jar

for f in $GRIDGAIN_HOME/libs/*.jar; do
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f;
```

```
done
```

4.3 Configure GGFS

> **NOTE:** make sure to have set `GRIDGAIN_HOME` environment variable to point to `distro` folder under GridGain installation.

GridGain comes with default configuration for GGFS file system. Hadoop needs to use file system remotely from client nodes as well as directly on data nodes. Client nodes are responsible for basic file system operations as well as accessing data nodes remotely. Usually, client nodes are started together with `job-submitter` or `job-tracker` processes, while data nodes are usually started together with Hadoop `task-tracker` processes.

Configuration files for GGFS client and data nodes are located in GridGain `config` folder. You can get started with default configuration and then change individual properties as you progress. To start client and data nodes respectively, you will use the following commands:

```
`bin/ggstart.sh config/ggfs/default-ggfs-client.xml`
`bin/ggstart.sh config/ggfs/default-ggfs-data.xml`
```

Refer to respective configuration files and `GridGgfsConfiguration` class JavaDoc for more information.

> **NOTE:** to make sure that GridGain nodes find each other over network, make sure to update `default-ggfs-base.xml` file with actual node IP address for `GridTcpDiscoverySpi` configuration. You can specify only 1 or 2 IP addresses of the nodes that you will start first.

4.4. Start GGFS Client and Data Nodes

After having configured Hadoop with GGFS as described above, you can start GGFS nodes. By default GridCain will use Hadoop 2.x libraries to connect to HDFS whenever this is needed. You should pass `-h1` command line argument to GridGain node to switch to Hadoop 1.x libraries.

1. On Hadoop `job-submitter` or `job-client` machine start GGFS client node as follows:
   ```
   bin/ggstart.sh config/ggfs/default-ggfs-client.xml
   ```
2. On Hadoop `job-tracker` machine start GGFS client node using the same command as in (1).
3. On machines that have Hadoop `task-tracker` and `data-node` start GGFS data node as follows:
   ```
   bin/ggstart.sh config/ggfs/default-ggfs-data.xml
   ```

> **NOTE:** you do not need to start multiple GGFS nodes on the same machine (although it is allowed). For example, if you submitting jobs from the same machine on which `job-tracker` or `task-tracker` is running, then you do not need to start additional GGFS nodes.

4.5 Start HDFS Data Nodes

> **NOTE:** skip this step if starting GGFS in `PRIMARY` mode.

If you wish to start GGFS in tandem with HDFS (either `PROXY`, `DUAL_SYNC`, or `DUAL_ASYNC` modes described above), you will need to start HDFS data nodes. To do this, do the following:

1. Set `HADOOP_CONF_DIR` environment variable either to *empty string* or to point to `/path/to/hadoop/conf` directory.
2. Execute `bin/start-dfs.sh` script under Hadoop 1.x installation and `sbin/start-dfs.sh` script under Hadoop 2.x installation as you would normally do to start HDFS.

4.6 Start Hadoop MapReduce

To start Hadoop MapReduce framework, do the following:

1. Set `HADOOP_CONF_DIR` environment variable to point to `/path/to/hadoop/conf-ggfs` directory.
2. Start Hadoop Map-Reduce with `bin/start-mapred.sh` under Hadoop 1.x installation directory and `sbin/start-yarn.sh` under Hadoop 2.x installation directory as you would normally do.

> **NOTE:** you should not start HDFS (specifically running `bin/start-dfs.sh` script) with `HADOOP_CONF_DIR` pointing to `conf-ggfs` folder as it will lead to *"java.net.UnknownHostException: unknown host: shmem"* error.

4.7 Use Hadoop with GGFS

You can now use Hadoop with GGFS utilizing all standard APIs and tools, just like you would normally do with any standard Hadoop installation.

> **NOTE:** By default GGFS does not back up data to other nodes for performance and capacity reasons. This means that whenever running in `PRIMARY` mode, any node shutdown would result in data loss and require restart of the cluster. If you need to add redundancy, then configure number of backups to value other than `0` in your data cache (the value is configured in `GridCachePartitionedAffinity.setBackups(...)` configuration property). In any mode other than `PRIMARY`, number of backups should always be `0`, as data can always be reloaded from disk in case of any crash and it does not make sense to back it up.

5. Benchmarks

GridGain ships with various benchmarks to compare GGFS performance to HDFS. You can find GGFS benchmarks under `benchmark/filesystem` folder in GridGain distribution. The following benchmark results were acquired on 7-node cluster of Dell R610 with Dual 8-Core CPUs (1 node to submit jobs and 6 data nodes):

Benchmark	GGFS (milliseconds)	HDFS (milliseconds)
Directory Create (1000 directories)	9,912	12,478
Directory Listing (local name-node, 100 dirs x 10 files)	**30**	**77**
Directory Listing (remote name-node, 100 dirs x 10 files)	**30**	**88**
Directory Delete (3 files per dir)	728	1,243
Directory Delete (empty dir)	240	1,245
Directory Random Operation (100 ops x 100 dirs)	943	3,651
File Create (10 x 1MB files)	**96**	**961**
File Create (5 x 256MB files)	**4,300**	**23,000**
File Delete (100 x 1MB files)	185	1,234
File Read/Scan (100 x 1MB files)	**273**	**720**
File Random Block Access (10 x 1MB files)	5	21
File Random Operation (100 ops x 10 x 1MB files)	431	2,931

No changes to default configuration were done when running these benchmarks.

5.1 Teragen and Terasort

We have also ran Hadoop `Teragen` and `Terasort` examples on 20GB data set. Note that the reason we are not seeing >10x performance with GGFS here is because both examples spend significant amount of time computing and sorting data vs. interacting with

file system. Yet, with GGFS we still see significant performance improvements over HDFS.

Benchmark	GGFS (seconds)	HDFS (seconds)
Teragen	33	70
Terasort	120	180

The following additional configuration parameters were passed to Hadoop when running `Teragen` and `Terasort`:

1. Teragen
   ```
   -Dio.file.buffer.size=$(( 256 * 1024)) // this is 256kb
   -Dmapred.map.tasks=24
   ```
2. Terasort
   ```
   -Dio.file.buffer.size=$(( 256 * 1024)) // this is 256kb
   -Dmapred.reduce.tasks=6 // number of reduce tasks equal to number of data nodes.
   ```

6. Management & Monitoring with Visor

GridGain comes with GUI and CLI (command) based DevOps Managements Consoles delivering advance set of management and monitoring capabilities. Visor GUI is based on a standalone Java application and CLI version is built on top of Scala REPL providing fully scriptable and customizable DevOps environment.

To start Visor GUI console, execute the following command:

```
`bin/ggvisorui.sh`
```

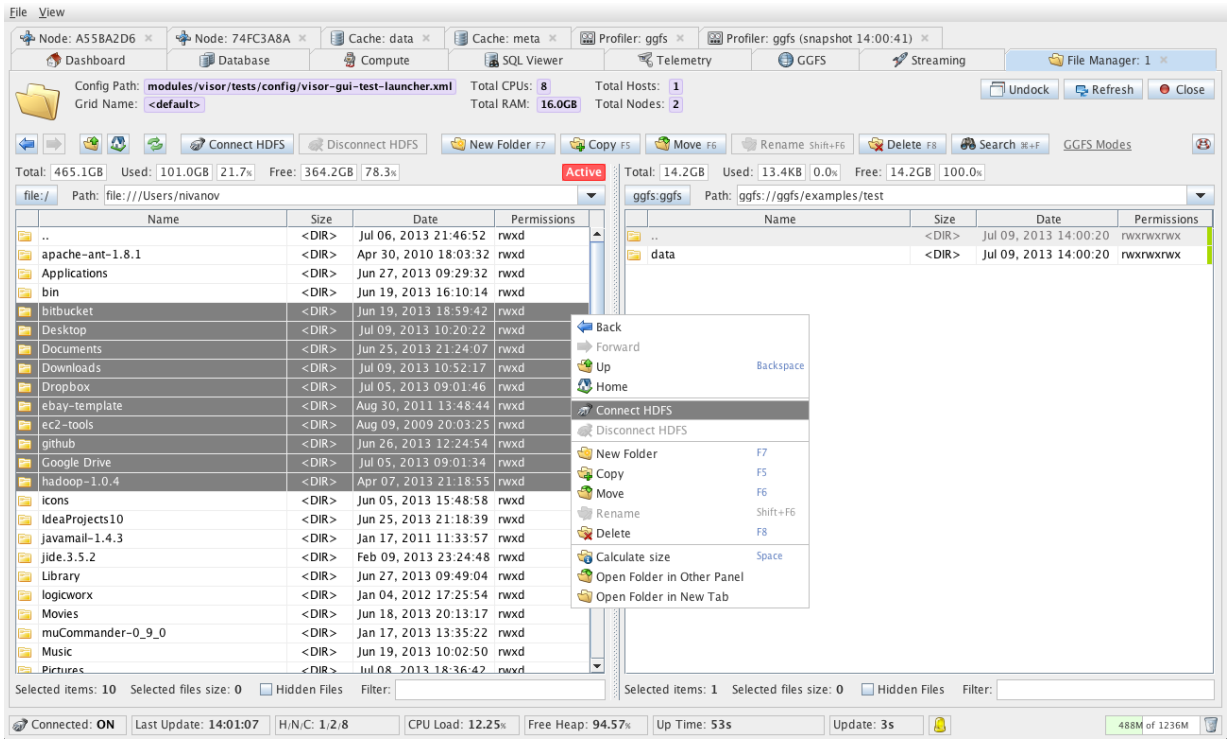
To start Visor in console mode you should execute the following command:

```
`bin/ggvisorcmd.sh`
```

On Windows, run the same commands with `.bat` extension.

> **NOTE:** Visor GUI console has a much richer set of functionality over Visor command-based console. You should always prefer Visor GUI console whenever possible.

Here is an example of `Visor File Manager Tab` which allows basic file system operations within same file system or across different file systems.


File View
Node: A55BA2D6
Node: 74FC3A8A
Cache: data
Cache: meta
Profiler: ggfs
Profiler: ggfs (snapshot 14:00:41)
Dashboard
Database
Compute
SQL Viewer
Telemetry
GGFS
Streaming
File Manager: 1
Config Path: modules/visor/tests/config/visor-gui-test-launcher.xml
Grid Name: <default>
Total CPUs: 8
Total RAM: 16.0GB
Total Hosts: 1
Total Nodes: 2
Undock
Refresh
Close
Connect HDFS
Disconnect HDFS
New Folder F7
Copy F5
Move F6
Rename Shift+F6
Delete F8
Search ⌘+F
GGFS Modes
Total: 465.1GB Used: 101.0GB 21.7% Free: 364.2GB 78.3%
Active
file:/
Path: file:///Users/nivanov
Total: 14.2GB Used: 13.4KB 0.0% Free: 14.2GB 100.0%
ggfs:ggfs
Path: ggfs://ggfs/examples/test
Name Size Date Permissions
.. <DIR> Jul 06, 2013 21:46:52 rwxd
apache-ant-1.8.1 <DIR> Apr 30, 2010 18:03:32 rwxd
Applications <DIR> Jun 27, 2013 09:29:32 rwxd
bin <DIR> Jun 19, 2013 16:10:14 rwxd
bitbucket <DIR> Jun 19, 2013 18:59:42 rwxd
Desktop <DIR> Jul 09, 2013 10:20:22 rwxd
Documents <DIR> Jun 25, 2013 21:24:07 rwxd
Downloads <DIR> Jul 09, 2013 10:52:17 rwxd
Dropbox <DIR> Jul 05, 2013 09:01:46 rwxd
ebay-template <DIR> Aug 30, 2011 13:48:44 rwxd
ec2-tools <DIR> Aug 09, 2009 20:03:25 rwxd
github <DIR> Jun 26, 2013 12:24:54 rwxd
Google Drive <DIR> Jul 05, 2013 09:01:34 rwxd
hadoop-1.0.4 <DIR> Apr 07, 2013 21:18:55 rwxd
icons <DIR> Jun 05, 2013 15:48:58 rwxd
IdeaProjects10 <DIR> Jun 25, 2013 21:18:39 rwxd
javamail-1.4.3 <DIR> Jan 17, 2011 11:33:57 rwxd
jide.3.5.2 <DIR> Feb 09, 2013 23:24:48 rwxd
Library <DIR> Jun 27, 2013 09:49:04 rwxd
logicworx <DIR> Jan 04, 2012 17:25:54 rwxd
Movies <DIR> Jun 18, 2013 20:13:17 rwxd
muCommander-0_9_0 <DIR> Jan 17, 2013 13:35:22 rwxd
Music <DIR> Jun 19, 2013 10:02:50 rwxd
Pictures <DIR> Jul 08, 2013 18:36:42 rwxd
.. <DIR> Jul 09, 2013 14:00:20 rwxrwxrwx
data <DIR> Jul 09, 2013 14:00:20 rwxrwxrwx
Back
Forward
Up Backspace
Home
Connect HDFS
Disconnect HDFS
New Folder F7
Copy F5
Move F6
Rename Shift+F6
Delete F8
Calculate size Space
Open Folder in Other Panel
Open Folder in New Tab
Selected items: 10 Selected files size: 0 Hidden Files Filter:
Selected items: 1 Selected files size: 0 Hidden Files Filter:
Connected: ON
Last Update: 14:01:07
H/N/C: 1/2/8
CPU Load: 12.25%
Free Heap: 94.57%
Up Time: 53s
Update: 3s
488M of 1236M