1. GridGain In-Memory Data Fabric

GridGain's In-Memory Data Fabric is designed to deliver uncompromised performance for the widest array of in-memory computing use cases.

Following components are included in the fabric:

- `In-Memory High Performance Computing (HPC)` - includes distributed clustering, messaging, events, and computational features.
- `In-Memory Data Grid` - partitioned in-memory key-value store with support for ACID transactions, off-heap memory, SQL, and more.
- `In-Memory Streaming` - supports event workflow, rolling data windows and indexing, continuous querying, and more.

Enterprise Features:

- Portable Objects
- Dynamic data structure changes without restarting cluster
- Cross-language support, including Java (JVM), .NET (C#), C++
- Datacenter replication
- Rolling production updates
- Local recoverable store
- Network segmentation protection
- Secure authentication and Secure client sessions
- GUI Management & Monitoring

2. GridGain Installation

GridGain distribution comes in a ZIP file that simply needs to be unzipped, and `GRIDGAIN_HOME` environment variable can optionally be set to point to it.

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 or 8 (latest update is advisable).
3. Point `JAVA_HOME` environment variable to your JDK or JRE installation.
4. Optional: point `GRIDGAIN_HOME` environment variable to the GridGain installation folder.

2.1 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 example configuration.

```
bin/ggstart.{sh|bat} examples/config/example-compute.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, you should execute the following script:

```
/bin/ggvisorcmd.{sh|bat} (Command-line, available in open source)
/bin/ggvisorui.{sh|bat} (GUI mode, enterprise version only)
```

2.2 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. Maven

GridGain provides repository for its Maven artifacts.

- GridGain Open Source repository is hosted at Maven Central (no additional URL required)
- GridGain Enterprise repository is located at http://www.gridgainsystems.com/nexus/content/repositories/external

3.1 Maven Artifacts

GridGain Maven repository has `4` artifacts (add `'-ent'` for enterprise edition):

- gridgain-hpc - contains jars and dependencies for In-Memory High Performance Computing (HPC)
- gridgain-datagrid - contains jars and dependencies for In-Memory Data Grid
- gridgain-streaming - contains jars and dependencies for In-Memory Streaming
- gridgain-fabric - contains jars and dependencies for all GridGain editions

3.2 Maven Example

3.2.1 Open Source

```
<dependency>
    <groupId>org.gridgain</groupId>
    <artifactId>gridgain-fabric</artifactId>
    <version>${gridgain.version}</version>
    <type>pom</type>
</dependency>
```

3.2.2 Enterprise

```
<repository>
    <id>GridGain External Repository</id>
    <url>http://www.gridgainsystems.com/nexus/content/repositories/exter
nal</url>
</repository>
...
<dependency>
    <groupId>org.gridgain</groupId>
    <artifactId>gridgain-fabric-ent</artifactId>
    <version>${gridgain.version}</version>
    <type>pom</type>
</dependency>
```

4. Starting Grid Nodes

Grid nodes can be started by executing `bin/ggstart.{sh|bat}` script and passing a relative path to GridGain configuration file. If no file is passed, then grid nodes are started with default configuration using Multicast discovery protocol.

Here is an example of how to start GridGain node with non-default configuration:

```
`bin/ggstart.sh examples/config/example-cache.xml`
```

5. 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 (enterprise edition only), execute the following command:

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

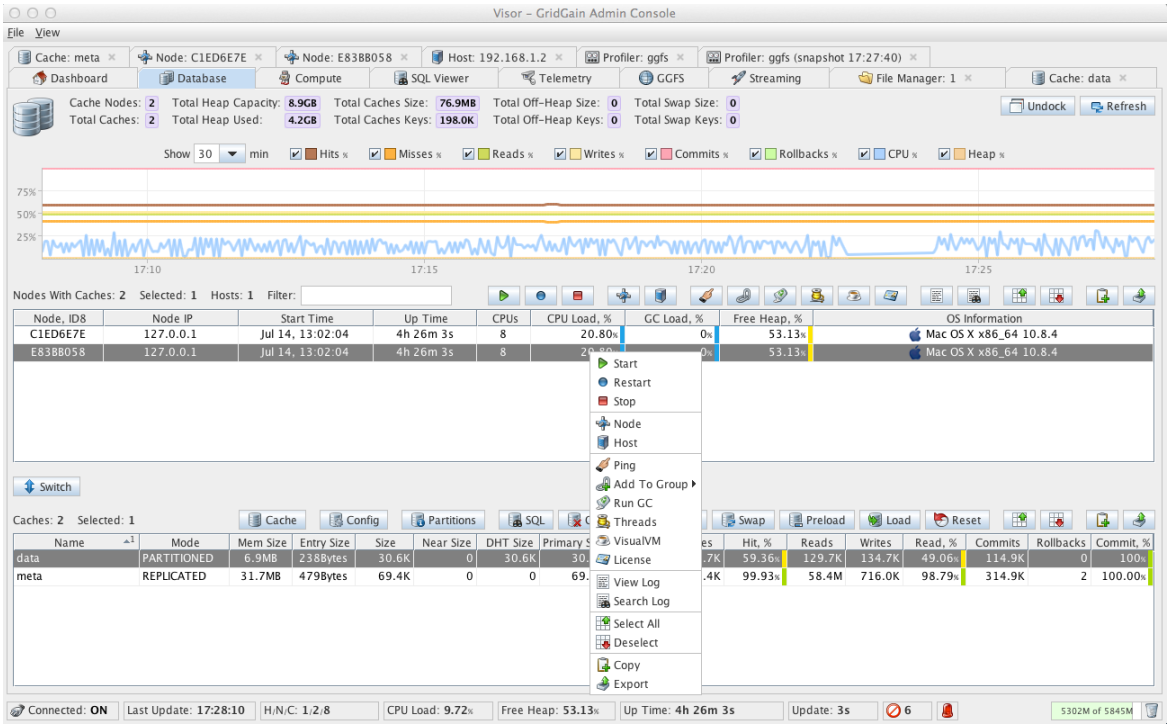
To start Visor in console mode you should execute the following command:

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

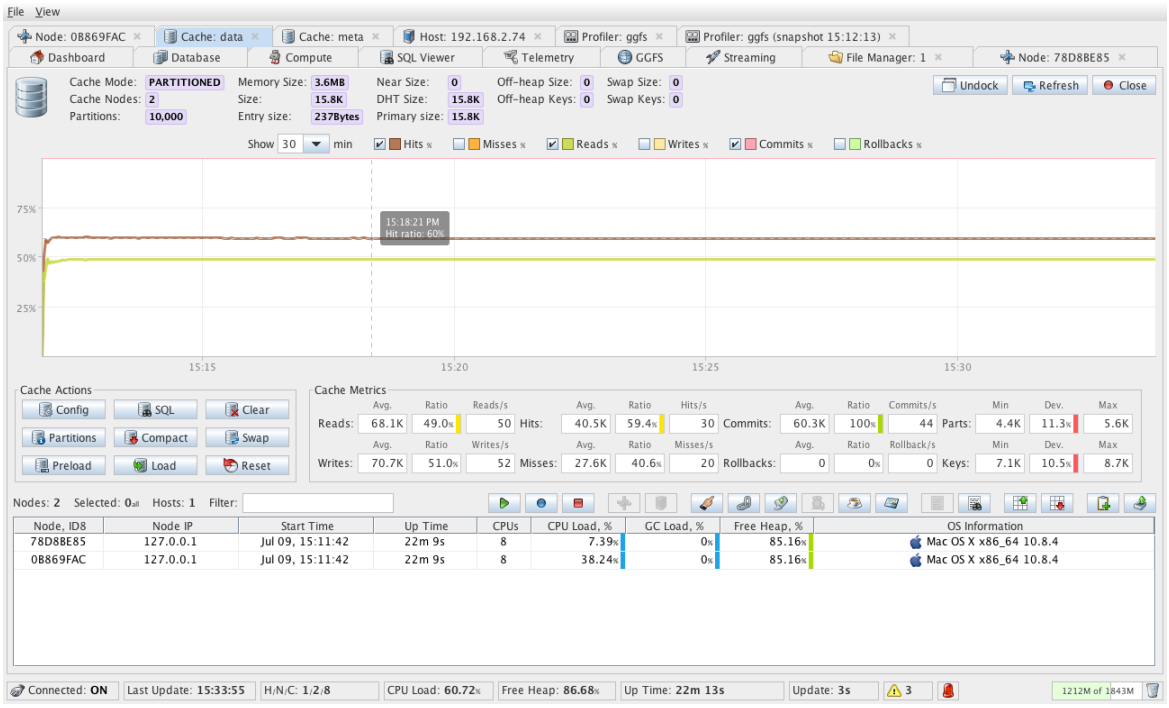
On Windows, run the same commands with `.bat` extension.

> **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 Data Grid Tab` which provides overall view on data grid.



Here is an example of `Visor Cache Tab` which provides view on individual cache.



6. Scala Integration

GridGain provides a very nice and easy to use DSL for Scala users called `Scalar`. If you like Scala, take a look at Scalar examples located under `examples/src/main/scala` folder.

7. Javadoc & Scaladoc

All documentation is shipped with it and you can find it under `docs/javadoc` and `docs/scaladoc` sub-folder respectively.