


# 1. GridGain In-Memory Data Grid

GridGain In-Memory Data Grid provides the following key features:

- Distributed java-based in-memory key-value store
- Distributed ANSI SQL and Full Text in-memory query capability over cached data
- JDBC driver for in-memory object data store
- PESSIMISTIC and OPTIMISTIC ACID distributed transactions
- JTA/JCA transaction integration
- Affinity routing with compute grid
- Local, fully replicable, and partitioned cache types
- Partitioned caches with active backups
- Pluggable expiration policies (LRU, random, time-based)
- Read-through, write-through, and write-behind persistence with pluggable cache store
- Pluggable data overflow storage
- Off-Heap memory support
- Tiered storage with on-heap, off-heap, swap space, SQL, and Hadoop
- Zero deployment for data

Enterprise Features

- Datacenter replication
- Rolling production updates
- Local restartable 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 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.

## 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 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, you should execute the following script:

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

### 2.3 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.

## 3. 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`
```

## 4. 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`
```

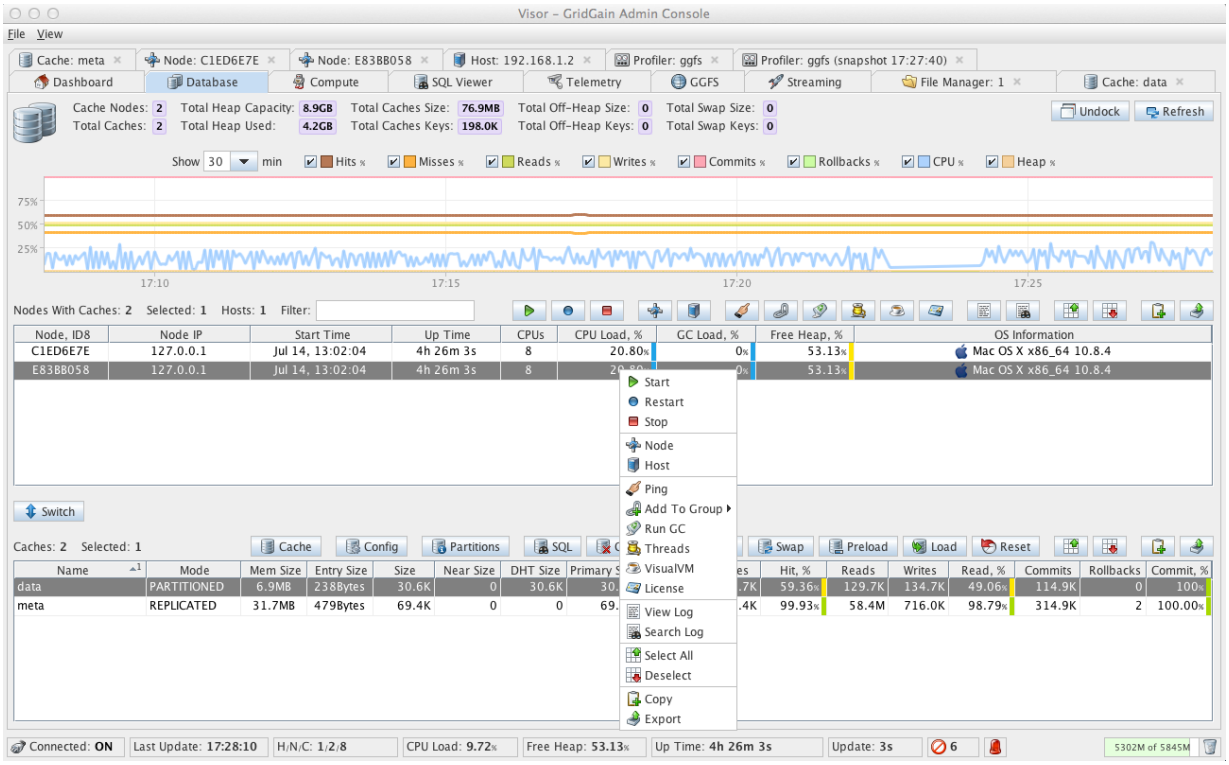
To start Visor in console mode you should execute the following command:

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

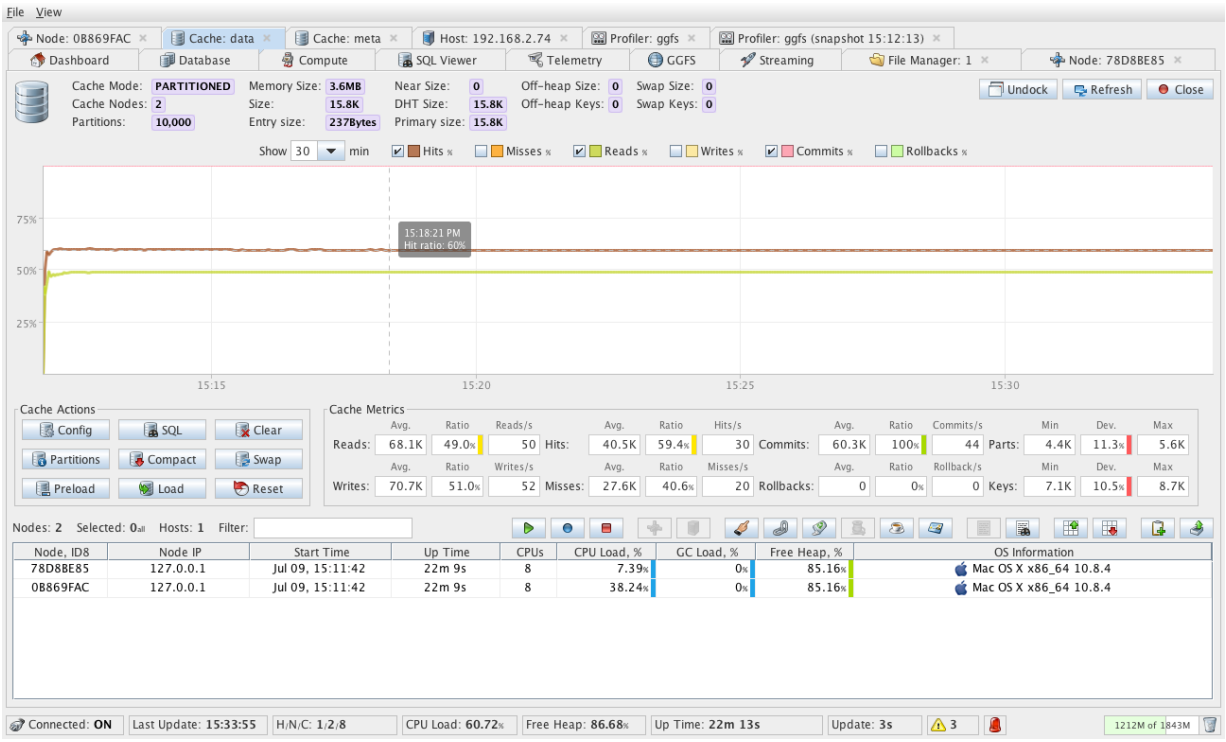
On Windows, run the same commands with `.bat` extension.

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




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




## 5. 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/scalar` folder.

## 6. Javadoc & Scaladoc

We spend significant amount of time on writing and maintaining Javadoc and Scaladoc for our product. All documentation is shipped with it and you can find it under `docs/javadoc` and `docs/scaladoc` sub-folder respectively.