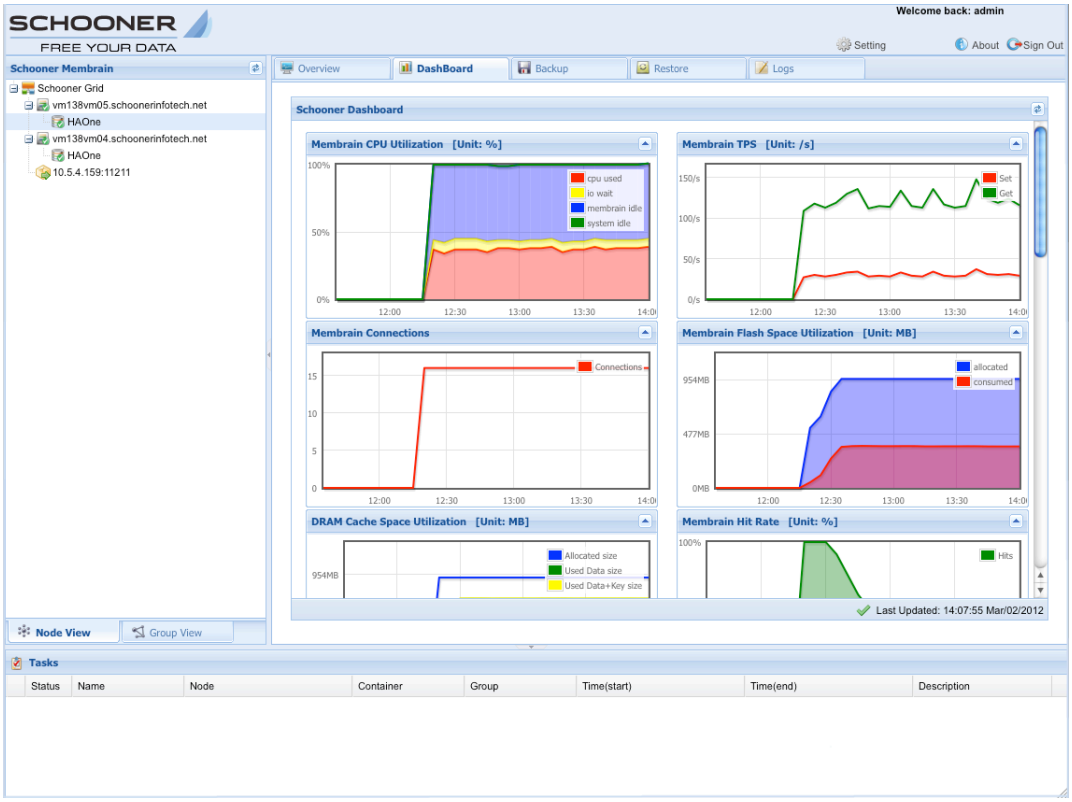
QuickStart Guide

Schooner Membrain™

Version 4.0

Software that transforms standard x86 servers and flash memory into NoSQL and caching super-servers



SCHOONER
FREE YOUR DATA

Technical Support

Technical support for Schooner Information Technology products in North America is available from the following sources:

- Phone: (877) 888-5064; (408) 888-1619
- Fax: (408) 736-4212
- Email: support@schoonerinfotech.com
- Website: www.schoonerinfotech.com/support

Documentation ID: S-M-v4.0-QG-01

Schooner Membrain™ QuickStart Guide

Issued March 2012

Schooner Information Technology
501 Macara Ave., Suite 101
Sunnyvale, CA 94085, USA
Tel: (408) 773-7500 (Main)
(877) 888-5064 (Sales and Support)
Fax: (408) 736-4212

Contents

Chapter 1: Introduction

Welcome

Thank you for purchasing Schooner Membrain™ and welcome to Schooner's new world of smart, fast, scalable, and cost-effective data access.

This document explains:

- Schooner Membrain™ concepts and features.
- Basic maintenance commands.
- Membrain statistics.

To find more detailed information on configuration, see the *Schooner Membrain™ Application & Administration Guide*.

Schooner Membrain™ Concepts

Schooner Membrain™ is based on the open source Memcached protocol and is completely compatible with the 1.4.4 Memcached version. In addition to supporting state-of-the-art Memcached performance and per node capacity, Schooner Membrain™ provides the following advanced features:

- Containers.
    - Containers define a set of access policies for a set of key-value pairs.
    - A container can be thought of as a Memcached instance, although for performance reasons, a single Schooner Membrain™ instance can support multiple containers, up to 128 in the current release.
    - Container policies control:
        - Size.
        - Addressing (TCP/UDP port).
        - Access (open or SASL).
        - Eviction (on/off).
        - Persistence (on/off).
        - Cache type (writeback/writethru).
    - Using these policies, containers can be tailored to provide controlled, high-speed access to persistent data.
- Persistence.
    - Containers may be configured for data persistence.
    - The cache type determines the durability (time between data is written to a container until it is persistently stored) of key-value pairs.

        - In writethru mode, data is written immediately to persistent storage.
        - In writeback mode, data is written to persistent storage if the cache manager decides that a permanent copy is required or if the application issues a “sync” or “sync_all” call. These calls are extensions to the Memcached protocol.
    - In the event of shutdown or failure, the container data store will be recovered to the last synchronization point if persistence is configured for the container. The data will be consistent to the last successful write in writethru mode and the last successful sync or sync_all call in writeback mode. If the container persistence policy is set off, the data will not be recovered and the container will be brought up empty.
- Synchronous replication groups.
    - “Mirror groups” support synchronous replication of containers.
    - A mirror group is a pair of Schooner Membrain™ servers that replicate a common set of containers.
    - Replication is synchronously performed at the key-value level by high-speed mechanisms.
        - In writeback mode, response to write operations is returned as soon as the replicated data has reached the mirrored node’s cache.
        - In writethru mode, response to write operations is returned as soon as the replicated data has reached the mirrored node’s persistent storage.
    - Each server supports read/write access to all containers.
        - Users may chose the access pattern to a mirrored group:
            - Single writer, single reader.
            - Single writer, multi-reader.
            - Multi-reader, multi-writer.
- High-speed cache.
    - A DRAM cache provides high-speed access to a much larger secondary store dataset.
    - The secondary data store is used for both persistent and non-persistent containers. This provides a data address range that is much greater than can be reasonably maintained in DRAM.
    - The DRAM cache is common to all containers. This allows the Schooner Membrain™ cache manager to effectively utilize DRAM resources to sustain the highest levels of performance.
- Solid-state storage.
    - Schooner Membrain™ can execute on any type of secondary storage, but it has been designed around high-speed solid-state secondary storage.

Schooner Membrain™ Administration Architecture

Administration Hierarchy

Schooner Membrain™ administration is based on a hierarchy of servers:

- Grid
  - A grid is a collection of Schooner Membrain™ servers that are managed under a common administration domain.
  - Schooner Membrain™ servers within a common grid can be administered from the same administration tools and be configured for replication.
- Cluster
  - A cluster is a collection of Schooner Membrain™ servers in a grid that can be administered by a configured set of administrators.
  - Schooner Membrain™ is currently limited to a single cluster within a grid.
- Group
  - A group is a set of nodes within a cluster that provide replication for a common dataset.

Chapter 2: Configuration and Management

Basic Configuring Using CLI

CLI Login

You can configure and manage Schooner Membrain™ using the CLI. However, you must complete the initial setup of your Schooner Membrain™ server using the Schooner First Time Wizard (see Installation & Setup Guide).

To configure and manage Schooner Membrain™ using the Centralized CLI:

1. Log in as "mbradmin", the user configured for Schooner Membrain™ administration.
2. At the > prompt, execute `'enable'`.
3. At the `#` prompt, execute `'configure terminal'`
4. At the `(config) #` prompt, execute `'membrain'`.

Once the Centralized CLI returns the (membrain) # prompt, you are now ready to use the various commands to configure and manage Schooner Membrain™ . You can use the `' ?'`, `'show ?'`, and/or `'list'` commands to learn about all the commands you need to configure and manage Schooner Membrain™ . The following paragraphs discuss only some of the commands.

Start, Stop and Restart Schooner Membrain™

- To start Membrain:

  ```
  (membrain) # start
  ```

- To stop Membrain:

  ```
  (membrain) # stop
  ```

- To enable or disable Membrain autorestart (restart on server boot):

  ```
  (membrain) # autorestart enable/disable
  ```

Container Configuration and Control

Containers are storage entities supported by Schooner Membrain™ to give you fine-grained control over memory and flash resources. Membrain clients access a container via its TCP/UDP port or IP addresses. You may additionally assign a name to each container for ease of maintenance.

Once you have created a container, it will remain inactive until you start it. Similarly, you must stop the container before you delete it.

Add a Container in cache eviction mode

```
(membrain)# container add TCPPORT UDPPORT CAPACITY yes
(writeback|writethru) (yes|no) CNAME (anyip|IPLIST)

TCPPORT:       Container TCP port number e.g., 11211
UDPPORT:       Container UDP port number e.g., 11211
CAPACITY:      Container capacity in gigabytes
yes:           Enable eviction
writeback:     Enable eviction with write-thru mode
writethru:     Enable eviction with write-back mode
no:            Disable persistence
CNAME:         Container name
anyip:         Any IP address
IPLIST:        A list of IP addresses separated by comma
```

Add a Container in persistent store mode

```
(membrain)# container add TCPPORT UDPPORT CAPACITY no (yes|no)
CNAME (anyip|IPLIST)

TCPPORT:       Container TCP port number e.g., 11211
UDPPORT:       Container UDP port number e.g., 11211
CAPACITY:      Container capacity in gigabytes
no:            Disable eviction
yes:           Enable persistence
CNAME:         Container name
anyip:         Any IP address
IPLIST:        A list of IP addresses separated by comma
```

Delete a Container

```
(membrain)# container delete CNAME

CNAME:   Container Name
```

Start a Container

```
(membrain)# container start CNAME

CNAME:   Container Name
```

Stop a Container

```
(membrain)# container stop CNAME

CNAME:   Container Name
```

Show All Containers

```
show containers

containers:    List of containers

Sample output:

Container 1
status      : on
name        : ABC
```

```
id          : 01
tcp_port    : 11211
udp_port    : 11211
eviction    : Yes
size        : 1024(MB)
Container 2
status      : on
name        : DEF
id          : 02
tcp_port    : 11212
udp_port    : 11212
eviction    : yes
size        : 1024(MB)
```

Create and Delete Mirrored Groups

Mirrored groups are the Schooner entities used to control container replication. Each Schooner Membrain™ node can belong to either the Independent Group (non-replicating) or the Mirrored Group (replicating).

When you add/delete/start/stop a container belonging to a node in a Mirrored Group, the Admin interface will perform the operation automatically on both of the mirrored nodes.

Create a Mirrored Group

```
(membrain)# mirrorgroup add GRPNAME NODE1 NODE2

GRPNAME: Group Name
NODE1:   Name of the first node
NODE2:   Name of the second node
```

Delete a Mirrored Group

```
(membrain)# mirrorgroup delete GRPNAME

GRPNAME:       Group Name
```

Container Backup/Restore

Schooner Membrain™ supports the backup and restore of persistent container data (cache containers cannot be backed up).

Backup

A full backup is a logical copy of all objects in a container. An incremental backup is a logical copy of objects in a container that are new or have changed since the previous backup, as well as a logical representation of deleted objects. A full backup is taken to start a new backup "series", which contains the full backup, plus zero or more incremental backups. There is no limit on the number of incremental backups that may be taken in a backup series (though there are practical considerations).

Restore

Restoring a backup is the process of replaying backup streams to the server. A backup can be restored to any container. The target container must already exist; the restore

process does not create a container. The user must ensure there is enough space in the container to hold all the restored objects.

Backups within a backup series are restored as a set, beginning with the full backup. Any backup series can be restored, and a backup series can be restored to an arbitrary backup within the series. All backups within the series, starting with the full backup, up to and including the desired backup will be restored.

Note: Refer to the Schooner Membrain™ Application and Administration Guide for more information and instructions on backup and restore of server configuration and Membrain data.

Backup a Container

```
(membrain)# container backup CNAME (incr|full) (local|nfs)

CNAME:   Container Name
incr:          Backup mode type incremental
full:          Backup mode type full
local:   Back up to the local machine
nfs:           Back up to an NFS machine
```

Restore a Container

```
(membrain)# container restore CNAME (local|nfs) BACKUPFILE

CNAME:   Container Name
local:   Back up to the local machine
nfs:           Back up to an NFS machine
BACKUPFILE:    Name of the backup file
```

Basic Configuration Using GUI

GUI Login

The Schooner Administrator GUI provides simple, graphical access to the same set of configuration interfaces shown above in the CLI description. To login to the GUI:

1. Enter the URL “http://<server ip>/admin.
2. The Schooner Administrator login page appears.
3. Enter the login credentials “admin/admin”.
4. Click “Login”.
5. The Schooner Administrator Overview page is displayed:

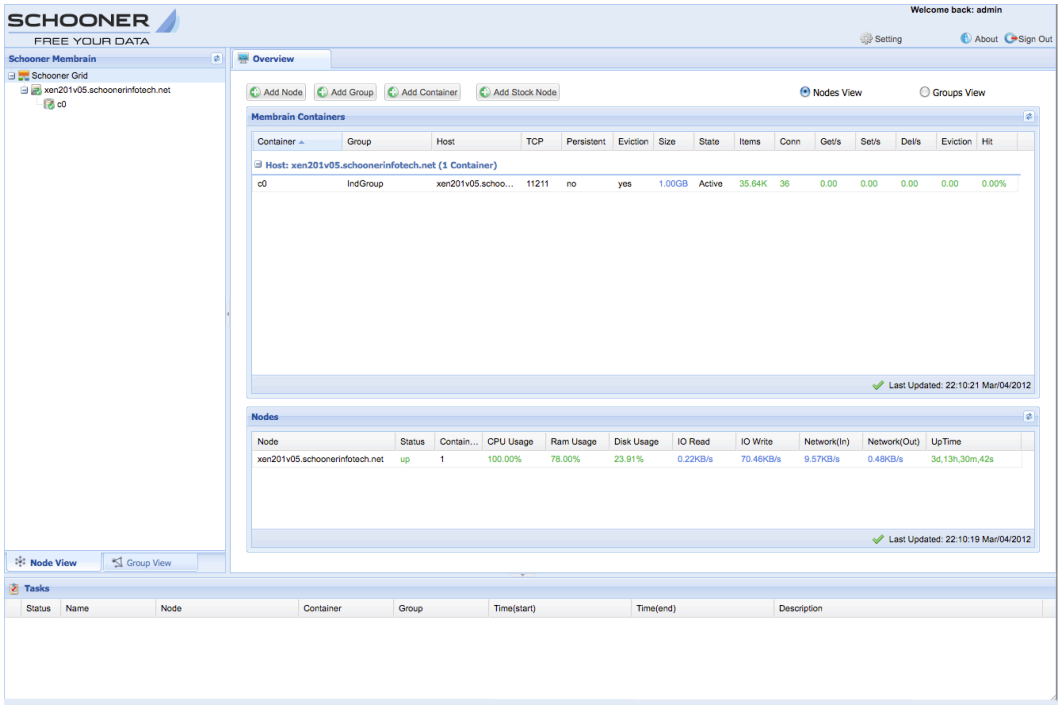


Figure 2-1: Overview Page

Start, Stop and Restart Schooner Membrain™

To start a Schooner Membrain™ instance:

1. Select the instance you wish to start in the navigation panel.
2. Click on the “Start” button:

Figure 2-2: Start Instance Button

3. The Schooner Membrain™ instance executes.

To stop a Schooner Membrain™ instance:

1. Select the instance you wish to stop in the navigation panel.
2. Click on the “Stop” button:

Figure 2-3: Stop Instance Button

3. The Schooner Membrain™ instance is stopped.

To restart a Schooner Membrain™ instance:

1. Select the instance you wish to restart in the navigation panel.
2. Click on the “Restart” button:

Figure 2-4: Start Instance Button

3. The Schooner Membrain™ instance is restarted.

Container Configuration and Control

Containers are storage entities supported by Schooner Membrain™ to give you fine-grained control over memory and flash resources. Membrain clients access a container via its TCP/UDP port or IP addresses. You may additionally assign a name to each container for ease of maintenance.

Once you have created a container, it will remain inactive until you start it. Similarly, you must stop the container before you delete it.

Add a Container in cache eviction mode

To add a container:

1. Select the node to host the container.
2. Click “Add Container”:

Figure 2-5: Add Container Button

3. The “Add Container” dialog displays:

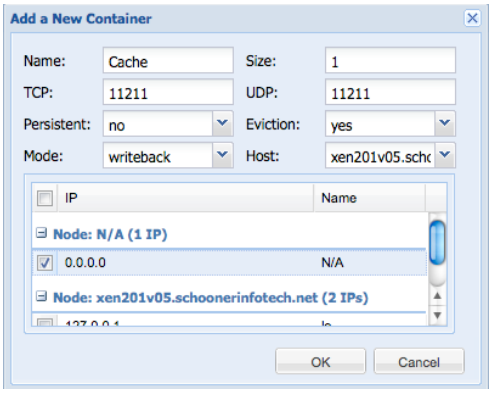


Figure 2-6: Add Container Dialog

- Enter the container name.
- Enter the TCP/UDP port numbers.
- Set persistence to “no” and eviction to “yes”.
- Select cache writeback or writethru mode.
- Select the IP addresses for the container.
- Click “Add”.

4. The message panel will display the status of the container add operation.
5. After the container has been created, you must click the “Start” button in order to activate the container.

Add a Container in store mode

To add a container:

1. Select the node to host the container.
2. Click “Add Container”:

Figure 2-7: Add Container Button

3. The “Add Container” dialog displays:

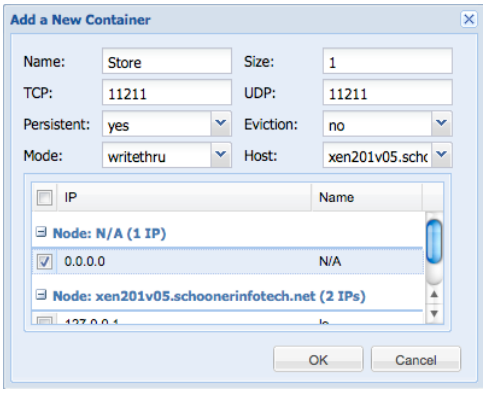


Figure 2-8: Add Container Dialog

   - Enter the container name.
   - Enter the TCP/UDP port numbers.
   - Set persistence to “yes” and eviction to “no”.
   - Select cache writeback or writethru mode.
   - Select the IP addresses for the container.
   - Click “Add”.
4. The message panel will display the status of the container add operation.
5. After the container has been created, you must click the “Start” button in order to activate the container.

Delete a Container

You can remove any container that is no longer needed. Keep in mind that you must stop a container before you attempt to delete it.

To delete a container:

1. Select the container.
2. Click the “Stop” button:

Figure 2-9: Stop Container Button

3. Click the “Remove” button:

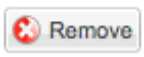


Figure 2-10: Remove Container Button

4. The message panel will display the status of the container removal.

Start a Container

To start a container:

1. Select the container.
2. Click the “Start” button:

Figure 2-11: Container Start Button

3. The message panel will display the status of the container start.

Stop a Container

To stop a container:

1. Select the container.
2. Click the “Stop” button:

Figure 2-12: Container Stop Button

3. The message panel will display the status of the container stop.

Create and Delete Mirrored Groups

Mirrored groups are the Schooner entities used to control container replication. Each Schooner Membrain™ node can belong to either the Independent Group (non-replicating) or the Mirrored Group (replicating). A Mirrored Group is limited to 2 nodes.

When you add/delete/start/stop a container belonging to a node in a Mirrored Group, the Admin interface will perform the operation automatically on both of the mirrored nodes.

Create a Mirrored Group

To create a mirrored group:

1. Click on the “Schooner Grid” icon.
2. Click on the “Add Group” button:

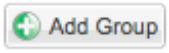


Figure 2-13: Add Group Button

3. The “Add Group” dialog displays:

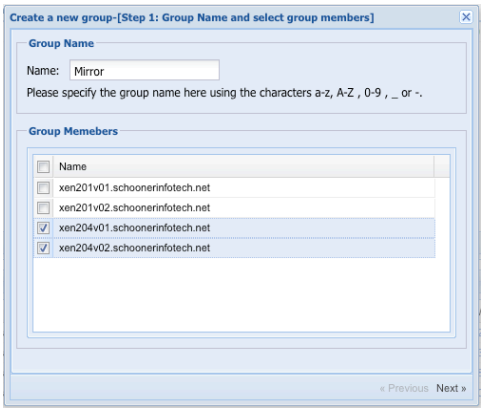


Figure 2-14: Add Group Dialog

- Enter the name of the mirror group.
- Select two nodes to form the mirror group.
- Click “Next”.

1. The “VIP” dialog displays:

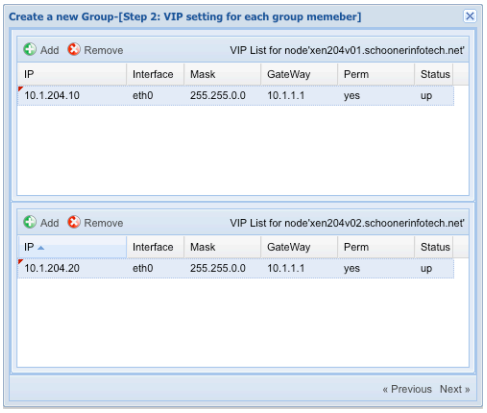


Figure 2-15: VIP Dialog

- Click “Add” to enter VIP configuration for each node.
- Click “Next”.

2. The “Admin Interface” dialog displays:

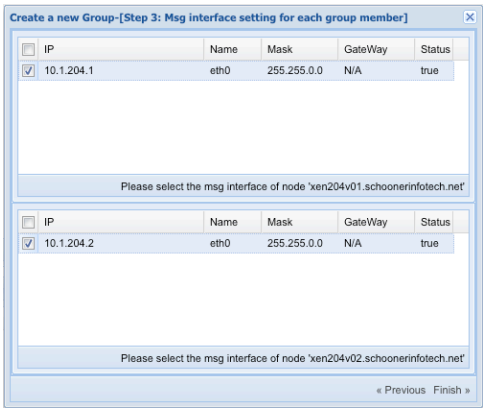


Figure 2-16: Admin Interface Dialog

- Select the admin interface for each node.
- Click “Finish”.

3. The message panel will display the status of the mirror group create.

Delete a Mirrored Group

To delete a mirrored group:

1. Click on the icon of the mirrored group.
2. Click on the “Remove Group” button:

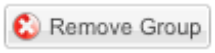


Figure 2-17: Remove Group Button

3. The message panel will display the status of the mirrored group delete.
4. On successful completion, the members of the mirrored group will be assigned to the independent pool. Their containers will be left intact but no longer in replication mode.

Container Backup/Restore

Schooner Membrain™ supports the backup and restore of persistent container data (cache containers cannot be backed up).

Backup

A full backup is a logical copy of all objects in a container. An incremental backup is a logical copy of objects in a container that are new or have changed since the previous backup, as well as a logical representation of deleted objects. A full backup is taken to start a new backup "series", which contains the full backup, plus zero or more incremental backups. There is no limit on the number of incremental backups that may be taken in a backup series (though there are practical considerations).

Restore

Restoring a backup is the process of replaying backup streams to the server. A backup can be restored to any container. The target container must already exist; the restore process does not create a container. The user must ensure there is enough space in the container to hold all the restored objects.

Backups within a backup series are restored as a set, beginning with the full backup. Any backup series can be restored, and a backup series can be restored to an arbitrary backup within the series. All backups within the series, starting with the full backup, up to and including the desired backup will be restored.

Note: Refer to the Schooner Membrain™ Application and Administration Guide for more information and instructions on backup and restore of server configuration and Membrain data.

Backup a Container

To back up a container:

1. Click on the container in the “Navigation Panel”.
2. Click on the “Backup” tab.
3. Click the “Create” button”:

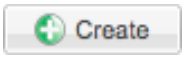


Figure 2-18: Create Button

4. The “Backup Job” dialog displays:

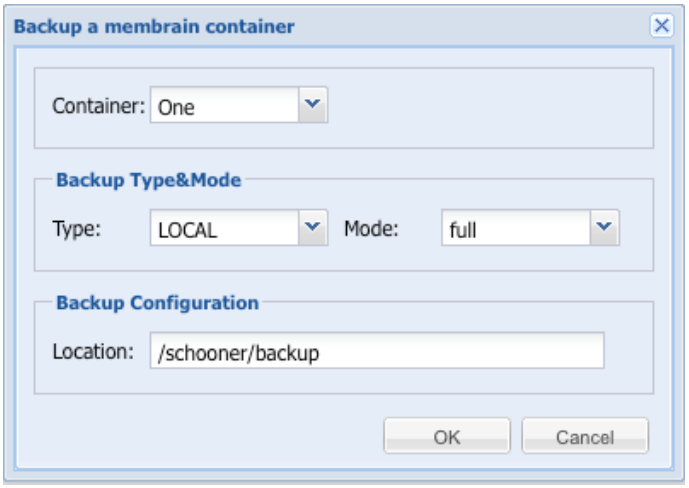


Figure 2-19: Backup Job Dialog

- Select the container to backup.
- Select the type of backup, local or remote.
  - If remote, select the node.
- Select the mode, full or incremental.
  - If incremental, select the full backup file associated with this backup.
- Enter the directory path for the backup.
- Click “OK”.

5. The message panel displays that status of the backup job.

Restore a Container

To restore a container from backup:

1. Click on the container in the “Navigation Panel”.
2. Click on the “Restore” tab.
3. Click the “Create” button”:

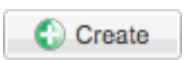


Figure 2-20: Create Button

4. The “Restore Job” dialog displays:

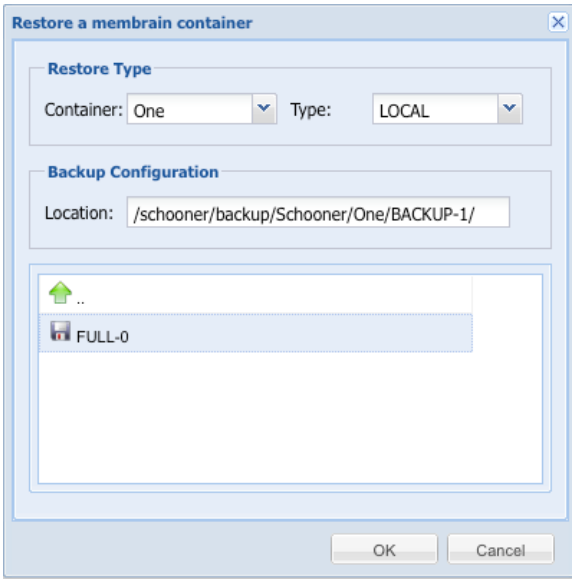


Figure 2-21: Restore Job Dialog

   - Select the container to restore.
   - Select the type of backup, local or remote.
      - If remote, select the node.
   - Select the backup file.
   - Click “OK”.
5. The message panel displays that status of the restore job.

Chapter 3: Verification

Schooner Membrain™ provides a suite of hardware and software verification tests:

- A storage system performance check of the data storage file system or devices.
- An application test (memslap) of Schooner Membrain™ .

Storage Verification

After installation of Schooner Membrain™ , you should run the storage system verification tests that are designed to gauge the throughput of your database log and data file systems and associated devices.

Data File System Test

To execute the data file system test follow the example below.

```
#cd /opt/schooner/bin
#sh membrain-fs-test.sh

These tests verify the following:

   Storage file system type (xfs) and performance (3500 iops)

Your performance may be sub-optimal if your server fails to meet any of these
requirements

Please enter Membrain storage type:
  1 File
  2 Device
  3 Exit

Enter choice: 1

  1 Storage performance test
    ************************
    ***     WARNING!!    ***
    ************************
    THIS TEST WILL CREATE A
    LARGE FILE AND/OR OVER-
    WRITE ANY DATA ON YOUR
    FILE SYSTEM OR DEVICE.
    DO NOT CONTINUE WITH
    THIS TEST WITHOUT
    BACKING UP YOUR DATA!!
    ************************
  2 Exit

Enter choice: 1
/dev/sda3 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sdb1 on /opt/schooner/backup type ext3 (rw)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/md0 on /schooner/data type xfs
(rw,noatime,nodiratime,sunit=512,swidth=7168,attr2,largeio,prjquota)

Please enter the data file system path: /schooner/data
 file type system is xfs
Enter the size of the test file in MB (>=100GB recommended): 100000
Fri Feb 4 02:44:26 PST 2011
Creating test file /schooner/data/ssddata0, size 100000 MB
100001+0 records in
100001+0 records out
104858648576 bytes (105 GB) copied, 128.333 seconds, 817 MB/s
Running io data test...
```

```
AIO_PERF> Opened ssd file '/schooner/data/ssddata0'.
AIO_PERF> Starting time: Fri Feb  4 02:46:40 2011
 0% (0.000430097 kIOPs)  1% (1.88375 kIOPs)  2% (1.88441 kIOPs)  3% (1.85681 kIOPs)  4%
(1.84912 kIOPs)  5% (1.7451 kIOPs)  6% (1.68267 kIOPs)  7% (1.6751 kIOPs)  8% (1.64175
kIOPs)  9% (2.66603 kIOPs)  10% (7.51354 kIOPs)
…
93% (4.93229 kIOPs)  94% (4.87085 kIOPs)  95% (4.66317 kIOPs)  96% (4.98953 kIOPs)  97%
(4.36201 kIOPs)  98% (3.98959 kIOPs)  99% (5.21159 kIOPs)  1e+02% (3.46978 kIOPs)
AIO_PERF>   Ending time: Fri Feb  4 03:30:31 2011

AIO_PERF> ================================================================
AIO_PERF> Bandwidth: 0.0579962 GB/sec (152.588 Gbytes in 2631 sec)
AIO_PERF> ================================================================
AIO_PERF> Total IOPs:    3.801 kIO/sec (     1e+07 IO's in     2631 sec)
AIO_PERF> Read  IOPs:        0 kIO/sec (         0 IO's in     2631 sec)
AIO_PERF> Write IOPs:    3.801 kIO/sec (     1e+07 IO's in     2631 sec)
AIO_PERF> ================================================================
Cleaning up...
Done
Fri Feb 4 03:30:31 PST 2011
```

Device Test

To execute the data file system test follow the example below. Note that this test can take up to an hour to complete:

```
#cd /opt/schooner/bin
#sh membrain-fs-test.sh
These tests verify the following:

   Storage file system type (xfs) and performance (3500 iops)

Your performance may be sub-optimal if your server fails to meet any of these
requirements

Please enter Membrain storage type:
  1 File
  2 Device
  3 Exit

Enter choice: 2

  1 Storage performance test
    ************************
    ***     WARNING!!    ***
    ************************
    THIS TEST WILL CREATE A
    LARGE FILE AND/OR OVER-
    WRITE ANY DATA ON YOUR
    FILE SYSTEM OR DEVICE.
    DO NOT CONTINUE WITH
    THIS TEST WITHOUT
    BACKING UP YOUR DATA!!
    ************************
  2 Exit

Enter choice: 1
Please enter the full path to the data device: /dev/md0
Enter the size of the test file in MB: (>=100GB recommended) 10000
AIO_PERF> Opened ssd file '/dev/md0'.
AIO_PERF> Starting time: Fri Feb  4 02:10:33 2011
 0% (3.18674 kIOPs)  1% (8.56294 kIOPs)  2% (7.44593 kIOPs)  3% (7.82226 kIOPs)  4%
(8.22797 kIOPs)  5% (8.21473 kIOPs)  6% (7.96949 kIOPs)  7% (8.44484 kIOPs)
…
94% (8.97505 kIOPs)  95% (8.90026 kIOPs)  96% (8.88163 kIOPs)  97% (8.82374 kIOPs)  98%
(9.07537 kIOPs)  99% (8.80172 kIOPs)  1e+02% (9.02719 kIOPs)
AIO_PERF>   Ending time: Fri Feb  4 02:29:24 2011

AIO_PERF> ================================================================
AIO_PERF> Bandwidth: 0.134914 GB/sec (152.588 Gbytes in 1131 sec)
AIO_PERF> ================================================================
AIO_PERF> Total IOPs:    8.842 kIO/sec (     1e+07 IO's in     1131 sec)
AIO_PERF> Read  IOPs:        0 kIO/sec (         0 IO's in     1131 sec)
AIO_PERF> Write IOPs:    8.842 kIO/sec (     1e+07 IO's in     1131 sec)
AIO_PERF> ================================================================

Mon Jan 17 12:08:37 PST 2011
```

Application Verification

In order to verify that Schooner Membrain™ is functioning properly, execute the following test:

```
# cd /opt/schooner/membrain_4.0/bin/membrain-verify
# ./mc-verify.pl -H 127.0.0.1 -p 11211 -d 5
windows size: 10k
cache size: 0.0G
ssd size: 0.0G
set miss rate: set_prop=0.05 set_md=1.00 set_mf=1.00
get miss rate: get_prop=0.95 get_md=0.05 get_mf=0.00

STAT pid 32219
STAT uptime 43
STAT time 1297143935.117291
STAT version 1.4.4-schooner 16105 membrain 3.1.0
STAT pointer_size 64
STAT rusage_user 39.142446
STAT rusage_system 60.487780
STAT curr_items 0
STAT total_items 6
STAT bytes 0
STAT curr_connections 4
STAT total_connections 13
STAT connection_structures 5
STAT cmd_get 76
STAT cmd_set 6
STAT get_hits 76
STAT get_misses 0
STAT evictions 0
STAT bytes_read 23508
STAT bytes_written 316118
STAT limit_maxbytes 1073741824
STAT threads 1
END


STAT pid 32219
STAT uptime 347
STAT time 1297144239.64436
STAT version 1.4.4-schooner 16105 membrain 3.1.0
STAT pointer_size 64
STAT rusage_user 235.002686
STAT rusage_system 346.753670
STAT curr_items 86017
STAT total_items 186595
STAT bytes 352325632
STAT curr_connections 0
STAT total_connections 76
STAT connection_structures 5
STAT cmd_get 3545237
STAT cmd_set 186595
STAT get_hits 3545237
STAT get_misses 0
STAT evictions 0
STAT bytes_read 1058231570
STAT bytes_written 13132746379
STAT limit_maxbytes 1073741824
STAT threads 1
END

cmd_get: 3545241
cmd_set: 186595
get_misses: 0
written_bytes: 1058230439
read_bytes: 13132338659
object_bytes: 687065625

run time: 302.8s opt: 3731836 TPS: 12323 net_rate: 44.7M/s
```

Chapter 4: Performance Monitoring

Schooner Membrain™ provides a set of performance charts. The charts are based on system and Membrain metrics collected by the “emt” monitoring system. This monitor samples metrics for cpu, memory, IO and Membrain at 5 minute intervals. Data is displayed for the past 2 hours.

To access the charts, select either a node or container and click on the Dashboard tab.

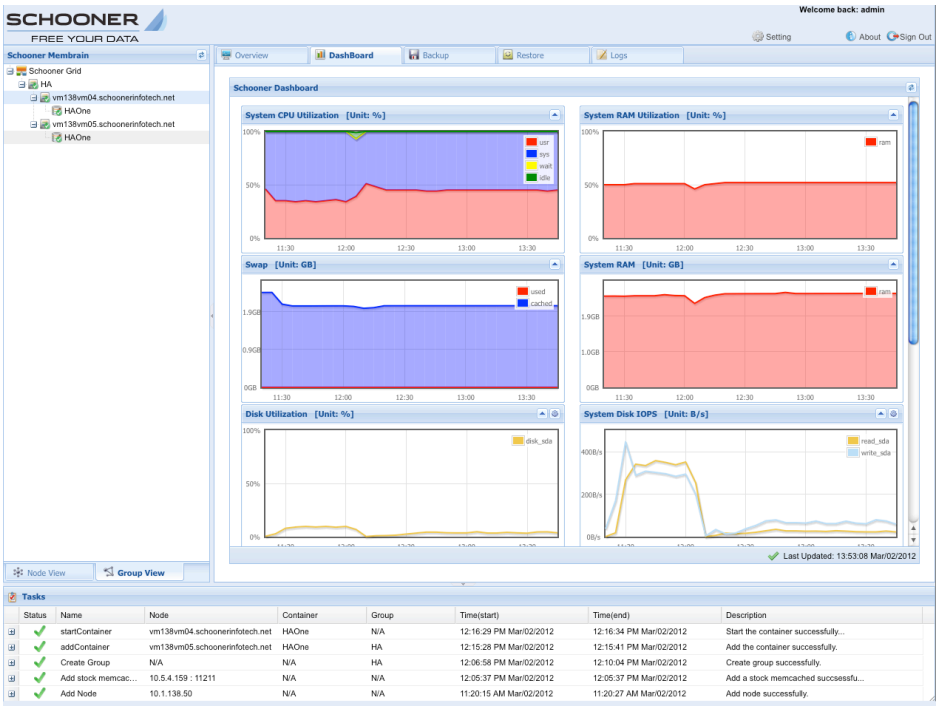


Figure 4-1: System Dashboard

System Dashboard

The system dashboard displays the following metrics:

- CPU utilization
- Memory utilization
- Disk bandwidth
- Network bandwidth

Container Dashboard

The container dashboard displays the following metrics:

- Membrain get/set TPS
- Container hit ratio
- Container connections
- Network bandwidth

- Average key length
- Average data length
- Eviction rate
- Membrain add/delete/replace TPS
- Membrain append/prepend TPS
- Membrain increment/decrement/CAS TPS
- Membrain sync/sync all/flush all TPS

Appendix: Schooner Membrain™ Statistics

Schooner Membrain™ supports the standard Memcached statistics as well as its own set of statistics.

Standard Memcached Statistics

The following are the standard Memcached statistics:

```
stats
STAT pid 30621
STAT uptime 17345
STAT time 1296879516.80556
STAT version 1.4.4-schooner 16105 membrain 3.1.0
STAT pointer_size 64
STAT rusage_user 152136.043805
STAT rusage_system 56005.107929
STAT curr_items 24010321
STAT total_items 159778658
STAT bytes 68554034688
STAT curr_connections 208
STAT total_connections 4021
STAT connection_structures 88
STAT cmd_get 316670533
STAT cmd_set 159778627
STAT get_hits 316362614
STAT get_misses 307919
STAT evictions 132660018
STAT bytes_read 585039132263
STAT bytes_written 1256905395107
STAT limit_maxbytes 68719476736
STAT threads 1
END
```

Schooner Membrain™ Statistics

The following are the Schooner Membrain™ statistics:

```
stats all
STAT pid 30621
STAT uptime 17466
STAT time 1296879637.580330
STAT version 1.4.4-schooner 16105 membrain 3.1.0
STAT pointer_size 64
STAT rusage_user 152836.930254
STAT rusage_system 56512.392810
STAT vol_ctx_sw 319544048, invol_ctx_sw 943219029
STAT page_reclaims 14894424, page faults 1
STAT signals_recv 0, swaps 0
STAT curr_items 24015705
STAT total_items 161351408
STAT bytes 68550572032
STAT curr_connections 208
STAT total_connections 4047
STAT connection_structures 88
STAT cmd_get 318070195
STAT cmd_set 161351371
STAT get_hits 317753976
STAT get_misses 316219
STAT evictions 134224998
STAT bytes_read 589147395006
STAT bytes_written 1260845338749
STAT limit_maxbytes 68719476736
STAT threads 1
STAT last_reset_time 1296862171.0
```

```
STAT Mcd_cmds (GT)318070195 (ST)161351371 (AD)0 (RP)0 (AP)0 (PP)0 (CS)0 (IC)0 (DC)0 (DL)0
(SN)0 (SA)0 (FA)0
STAT Mcd_get_key: evs=691334227 min=16 max=250 avg=98.757274 geo=64.943544 std=83.525954
STAT Mcd_get_key: (4)120826376 (5)145798441 (6)0 (7)292545633 (8)132163777
STAT Mcd_get_data: evs=691334227 min=2 max=131033 avg=4499.248537 geo=1348.033010
std=7441.630371
STAT Mcd_get_data: (0)0 (1)5944249 (2)11772920 (3)29725748 (4)35681783 (5)4064491
(6)7602208 (7)21073171 (8)26610657 (9)23536415 (10)11596632 (11)4157047 (12)408616314
(13)29385794 (14)39991560 (15)13240310 (16)17523313 (17)811615
STAT Mcd_set_key: evs=332895319 min=16 max=250 avg=91.178272 geo=60.177729 std=81.236290
STAT Mcd_set_key: (4)45066895 (5)104261112 (6)0 (7)126920250 (8)56647062
STAT Mcd_set_data: evs=332895319 min=2 max=131033 avg=5236.494234 geo=317.225716
std=10726.449866
STAT Mcd_set_data: (0)0 (1)5952836 (2)11782916 (3)29761710 (4)35715014 (5)21117052
(6)24672320 (7)24961283 (8)26687640 (9)6683313 (10)11637003 (11)4171510 (12)28790694
(13)29381560 (14)39998657 (15)13241708 (16)17527451 (17)812652
STAT Mcd_get_time: evs=0
STAT Mcd_set_time: evs=0
STAT flash_class_map 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
STAT flash_slab_map 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
STAT flash_space_allocated 68719476736
STAT flash_space_consumed 68550595584
STAT flash_num_objects 24015725
STAT flash_num_created_objects 158259939
STAT flash_num_evictions 134224998
STAT flash_num_overwrites 19216
STAT flash_num_ops 112118063
STAT flash_num_read_ops 112006791
STAT flash_num_get_ops 35596839
STAT flash_num_put_ops 158259939
STAT flash_num_del_ops 0
STAT sdf_cache_hits 282473466
STAT sdf_cache_misses 35596854
STAT sdf_flash_hits 35280617
STAT sdf_flash_misses 316219
STAT sdf_cache_evictions 0
STAT sdf_n_overwrites 1261
STAT sdf_n_in_place_overwrites 20541
STAT sdf_n_new_entries 196610257
STAT sdf_cache_only_items 0
END
```

The interesting additions provided by Schooner Membrain™ "stats all" are:

- Mcd_cmds - counts for the following commands:
    - GT - get
    - ST - set
    - AD - add
    - RP - replace
    - AP - append
    - PP - prepend
    - CS - cas
    - IC - increment
    - DC - decrement
    - DL - delete
    - SN - sync
    - SA - sync_all
    - FA - flush_all
- Mcd_get_key, Mcd_get_data, Mcd_set_key, Mcd_set_data
    - evs - events
    - min - minimum size
    - max - maximum size

    - avg - average size
    - geo - geometric mean
    - std - standard deviation
    - data buckets in power of 2 size
- flash_space_allocated
    - secondary storage allocated
- flash_space_consumed
    - secondary storage consumed
- flash_num_objects
    - secondary storage objects
- flash_num_created_objects
    - secondary storage object creations
- flash_num_evictions
    - secondary storage evictions
- flash_num_overwrites
    - secondary storage object overwrites
- flash_num_ops
    - secondary storage operations
- flash_num_read_ops
    - secondary storage reads
- flash_num_put_ops
    - secondary storage writes
- flash_num_del_ops
    - secondary storage deletes
- sdf_cache_hits
    - DRAM cache hits
- sdf_cache_misses
    - DRAM cache misses
- sdf_cache_evictions
    - DRAM cache evictions
- sdf_flash_hits
    - Secondary storage get hits
- sdf_flash_misses
    - Secondary storage object misses
- sdf_n_overwrites
    - Cached object overwrites