INTEL CORPORATION

COSBench User Guide

Version 2.2.0

Wang, Yaguang

1/29/2013

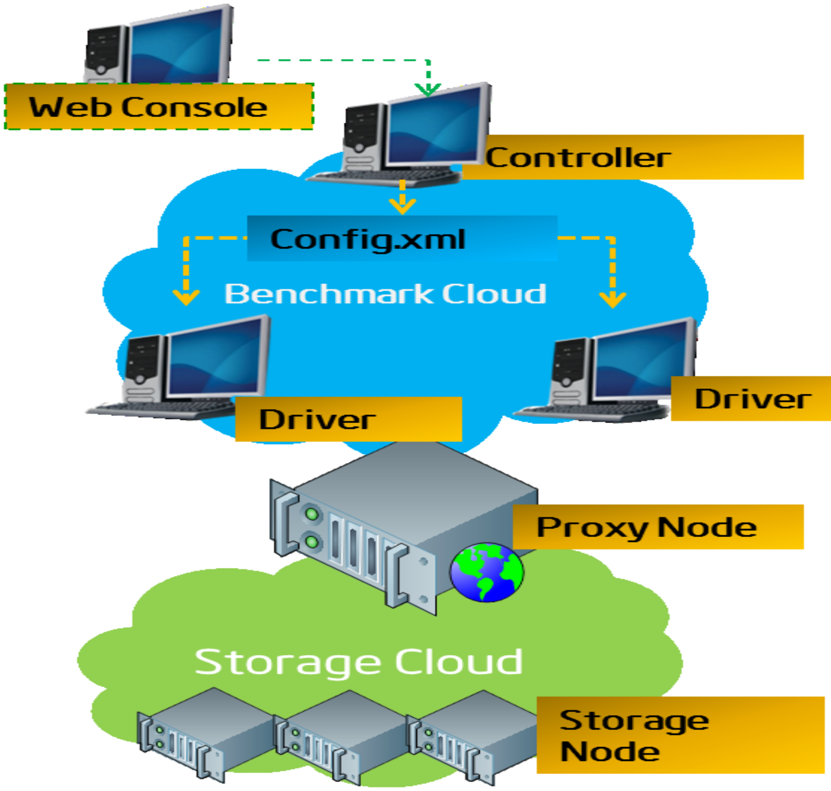
This document describes how to install, configure and run COSBench (a cloud storage benchmark tool) step by step, and explains how to define workload through configuration file, and then share examples for reference.

Revision	Date	Author	Reviewer	Description
0.5	7/14, 2012			. Initial version
0.6	7/18, 2012			. Add "init", "dispose" stage in amplistor example, and description for special stages.
0.7	7/20, 2012			. Add "nsroot" at storage parameter list to access amplistor v2.5 namespace, by default, it's "/namespace", set to "/manage/namespace" for v2.5.
0.8	7/24, 2012			. Change default listening ports: 8088->18088 8089->18089 9088->19088 9089->19089
0.9	8/1, 2012			. Change example port number to 19088 and 18088 also to avoid confusion.
1.0	8/9, 2012			. Add one section to describe data results, one section for FAQ.
1.1	8/13, 2012			. Add one paragraph in result to explain metrics . Modify amplistor sample to reflect v2.5 needs. . Add parameter list
1.2	8/24, 2012			. Enhance the content based on internal and external users' feedback.
1.3	8/30, 2012			. Add screenshot for redhat package should choose . Change runtime from 60 to 300 for amplistor example to avoid confusion . Remove internal link for package downloading. . Fix one bug in "cleanup" stage in swift sample.
1.4	9/14, 2012			. Fix inconsistence bugs.
1.5	9/17, 2012			. Stick default OS to Ubuntu 12.04.1 LTS desktop.
1.6	11/02, 2012			Major modifications: . Transfer all scripts to Ubuntu 12.04.1 compatible. . Add O/S installation steps. . Add object integrity check parameter . Add details about selector description. . Add details about directory structure. . Move workload configuration section from appendix to main chapter.
1.7	11/13, 2012			Minor modifications: . correct batch script names . Add one item in FAQ for handling "OOM" error.
1.8	11/20, 2012			. Change parameter "url" to "auth_url" for swauth and keystone

				to avoid confusion.
1.9	1/14, 2013			. Add parameter “tenant_name” for keystone. . Add items in FAQ to explain testing with large objects.
2.0	1/25, 2013			. Correct two minor typos. . Add explanation about histogram data. . Reword FAQ #12.
2.1	2/19, 2013			. Constraint supported amplistor version to v2.3 and v2.5. . Minor formatting modifcations.

1. Introduction

1.1 Overview



COSBench is a distributed benchmark tool to test cloud object storage systems, till now it can supports openstack swift/amplidata amplistor v2.3 and v2.5 system.

COSbench consists of two key components:

*** Driver (aka COSBench Driver, or Load Generator), which is**

- Responsible for workload generation, issuing operations to target cloud object storage and collecting performance statistics.
- Can be accessed via http://<driver-host>:18088/driver/index.html

* **Controller (aka COSBench Controller), which is**

- responsible for coordinating driver to collectively execute a workload, collecting and aggregating runtime status or benchmark results from driver instances, accepting workload submission
- can be accessed via http://<controller-host>:19088/controller/index.html

The controller and driver can be deployed on the same node or different nodes, and the node can be a physical machine or vm instance.

1.2 Reference H/W configuration

Below hardware configurations are what used in Intel lab for COSBench validation, just shown them here as reference, and what hardware configurations should be adopted is highly depending on usage scenarios, and our observation is network is normally the concern for COSBench nodes.

Hardware	**Configuration**
Controller	
Processor	2x NHM-EP 2.93GHz
RAM	12GB RAM
Storage	1x 120GB+ disk drive
Network	Intel 82574 1GbE
Driver	
Processor	2x NHM-EP 2.93GHz
RAM	12GB RAM
Storage	1x 50GB+ disk drive
Network	Intel 82599 10GbE

1.3 Prerequisite

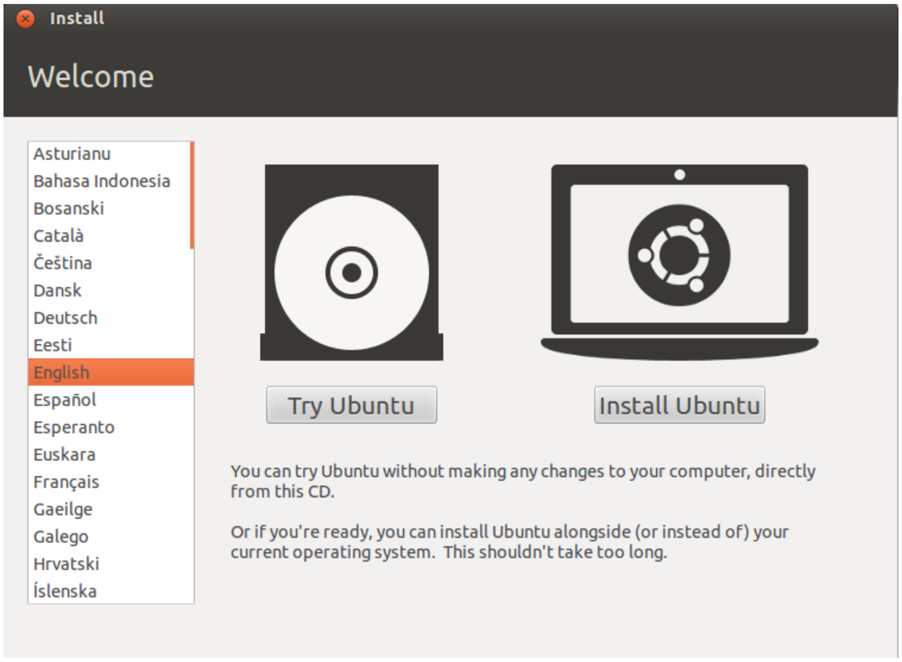
- Ubuntu 12.04.1 LTS Desktop (codename: precise)
- Java Runtime Environment 1.7+
- Curl 7.22.0+
- Free TCP port (ensure those ports accessible outside local):
  - On COSBench controller machine: 19088
  - On COSBench driver machines: 18088
- (NOTE: command line is **bolded** and *italic, and important content will be in yellow color to emphasize.*)

2. Install

2.1 Install O/S

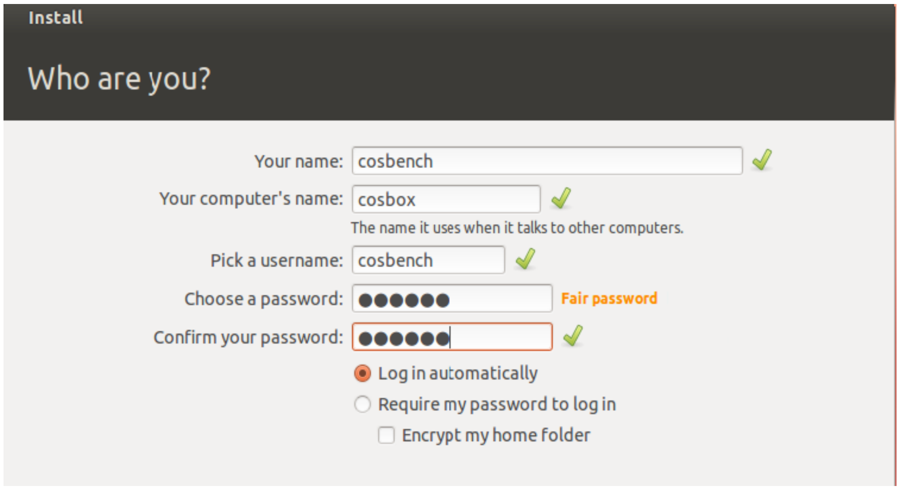
- Download Ubuntu Desktop 12.04.1 LTS.
- Follow Ubuntu installation guide.

- Below are screenshots from major steps during installation, we will create one user named "cosbench", and all other steps are left as default or modified at user's discrimination.



You can try Ubuntu without making any changes to your computer, directly from this CD.

Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.



Your name: cosbench ✓

Your computer's name: cosbox ✓

The name it uses when it talks to other computers.

Pick a username: cosbench ✓

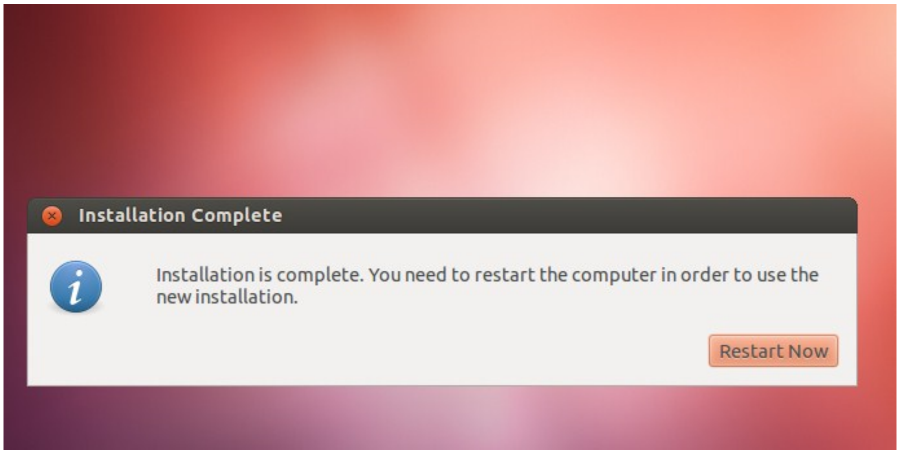
Choose a password: ●●●●●● Fair password

Confirm your password: ●●●●●● ✓

● Log in automatically

○ Require my password to log in

□ Encrypt my home folder



Installation is complete. You need to restart the computer in order to use the new installation.

2.1.1 Install JRE (Java Runtime Environment)

- OpenJDK is the default, and Oracle JRE should also work.
- If internet connection is available, the package can be installed through apt-get as following:

```
cosbench@cosbox:~$ sudo apt-get update
cosbench@cosbox:~$ sudo apt-get install openjdk-7-jre
```

- If no internet connections available on the node, it's possible to install JRE through deb packages, and two deb packages are essential: JRE-LIB, JRE-HEADLESS.
- To install those packages (assuming packages are at */tmp*):

```
cosbench@cosbox:/tmp$ sudo dpkg –i –force depends openjdk-7-jre-lib_7u7-2.3.2a-0ubuntu0.12.04.1_all.deb
(Reading database …

cosbench@cosbox:/tmp$ sudo dpkg –i –force depends openjdk-7-jre-headless_7u7-2.3.2a-0ubuntu0.12.04.1_amd64.deb
Selecting previously unselected package openjdk-7-jre-headless …

cosbench@cosbox:/tmp$ java –showversion
java version "1.7.0_07"
…
```

2.1.2 Install curl

- With internet connection available:

```
cosbench@cosbox:~$ sudo apt-get update
cosbench@cosbox:~$ sudo apt-get curl
```

- If no internet connection is available, need install curl package through deb package.

```
cosbench@cosbox:/tmp$ sudo dpkg –i curl_7.22.0-3ubuntu4_amd64.deb

cosbench@cosbox:/tmp$ curl –V
curl 7.22.0 (x86_64-pc-linux-gnu) …
```

2.2. Install COSBench

2.2.1 Prepare

In our release package, COSBench controller and driver are combined together; there is no separate package for each of them.

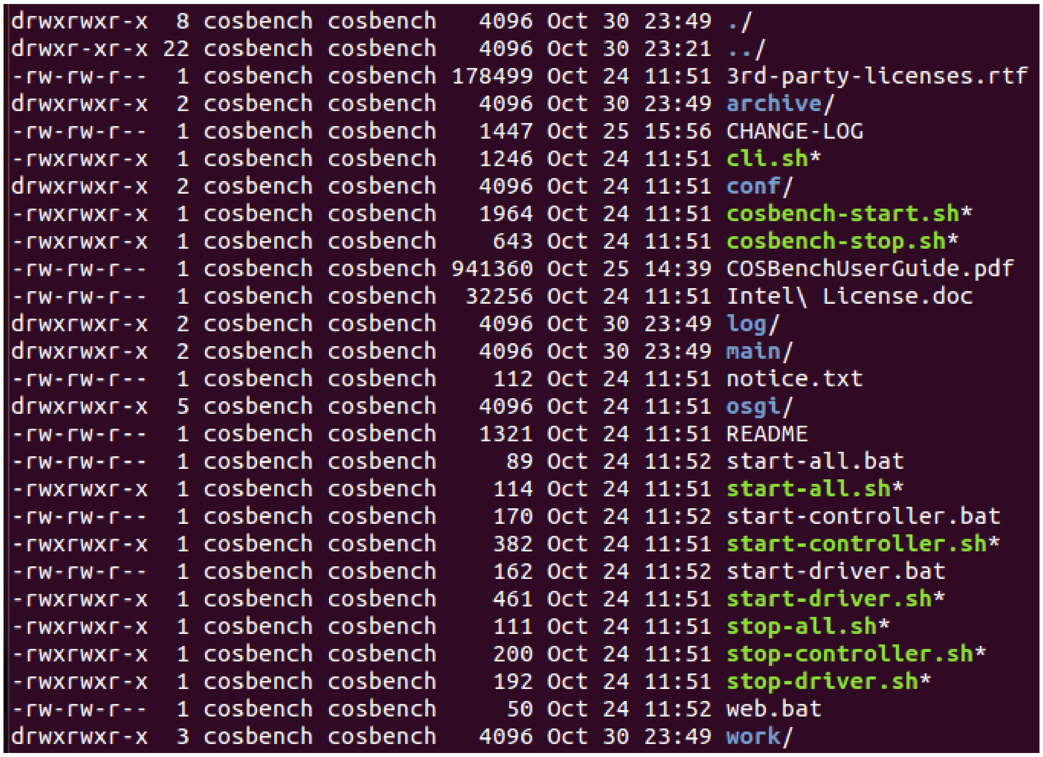
Getting installation package <version>.zip (e.g. 2.1.0.GA.zip) through some way like email, ftp, shared folder, etc., and put it at COSBench package under home directory on controller node.

2.2.2 Install

Follow below commands to finish the installation:

```
cosbench@cosbox:/tmp$ cd ~
cosbench@cosbox:~$ unzip 2.1.0.GA.zip
cosbench@cosbox:~$ rm cos
cosbench@cosbox:~$ ln –s 2.1.0.GA/ cos
cosbench@cosbox:~$ cd cos
cosbench@cosbox:~$ chmod +x *.sh
```

2.3 Directory Structure



```
drwxrwxr-x  8 cosbench cosbench   4096 Oct 30 23:49 ./
drwxr-xr-x 22 cosbench cosbench   4096 Oct 30 23:21 ../
-rw-rw-r--  1 cosbench cosbench 178499 Oct 24 11:51 3rd-party-licenses.rtf
drwxrwxr-x  2 cosbench cosbench   4096 Oct 30 23:49 archive/
-rw-rw-r--  1 cosbench cosbench   1447 Oct 25 15:56 CHANGE-LOG
-rwxrwxr-x  1 cosbench cosbench   1246 Oct 24 11:51 cli.sh*
drwxrwxr-x  2 cosbench cosbench   4096 Oct 24 11:51 conf/
-rwxrwxr-x  1 cosbench cosbench   1964 Oct 24 11:51 cosbench-start.sh*
-rwxrwxr-x  1 cosbench cosbench    643 Oct 24 11:51 cosbench-stop.sh*
-rw-rw-r--  1 cosbench cosbench 941360 Oct 25 14:39 COSBenchUserGuide.pdf
-rw-rw-r--  1 cosbench cosbench  32256 Oct 24 11:51 Intel\ License.doc
drwxrwxr-x  2 cosbench cosbench   4096 Oct 30 23:49 log/
drwxrwxr-x  2 cosbench cosbench   4096 Oct 30 23:49 main/
-rw-rw-r--  1 cosbench cosbench    112 Oct 24 11:51 notice.txt
drwxrwxr-x  5 cosbench cosbench   4096 Oct 24 11:51 osgi/
-rw-rw-r--  1 cosbench cosbench   1321 Oct 24 11:51 README
-rw-rw-r--  1 cosbench cosbench     89 Oct 24 11:52 start-all.bat
-rwxrwxr-x  1 cosbench cosbench    114 Oct 24 11:51 start-all.sh*
-rw-rw-r--  1 cosbench cosbench    170 Oct 24 11:52 start-controller.bat
-rwxrwxr-x  1 cosbench cosbench    382 Oct 24 11:51 start-controller.sh*
-rw-rw-r--  1 cosbench cosbench    162 Oct 24 11:52 start-driver.bat
-rwxrwxr-x  1 cosbench cosbench    461 Oct 24 11:51 start-driver.sh*
-rwxrwxr-x  1 cosbench cosbench    111 Oct 24 11:51 stop-all.sh*
-rwxrwxr-x  1 cosbench cosbench    200 Oct 24 11:51 stop-controller.sh*
-rwxrwxr-x  1 cosbench cosbench    192 Oct 24 11:51 stop-driver.sh*
-rw-rw-r--  1 cosbench cosbench     50 Oct 24 11:52 web.bat
drwxrwxr-x  3 cosbench cosbench   4096 Oct 30 23:49 work/
```

2.3.1 License (Intel License.doc)

The license document for COSBench should already be acknowledged before using COSBench.

2.3.2 Scripts

script	description
start-all.sh stop-all.sh	Start/stop both controller and driver on current node.
start-controller.sh stop-controller.sh	Start/stop controller only on current node.
start-driver.sh stop-driver.sh	Start/stop driver only on current node.
cosbench-start.sh cosbench-stop.sh	Internal scripts called by above scripts.
cli.sh	Manipulate workload through command line.

A few windows based batch scripts are also included, and they are just for demonstration only to give quick impression about the tool on windows.

script	description
start-all.bat	Start both controller and driver on current node.

start-controller.bat	Start controller only on current node.
start-driver.bat	Start driver only on current node.
Web.bat	Open controller web console through local installed browser.

2.3.3 Sub-directories

Sub-directory	description
archive	Stores all generated results, see results section
conf	Configuration files, including cosbench’s configurations and workload configurations.
log	runtime log files, the important one is system.log
osgi	COSBench libraries and third-parties libraries are here.
main	The osgi launcher is here.

2.4 Sanity check

The steps will launcher controller and driver on current node and run mock test to ensure installation is correct.

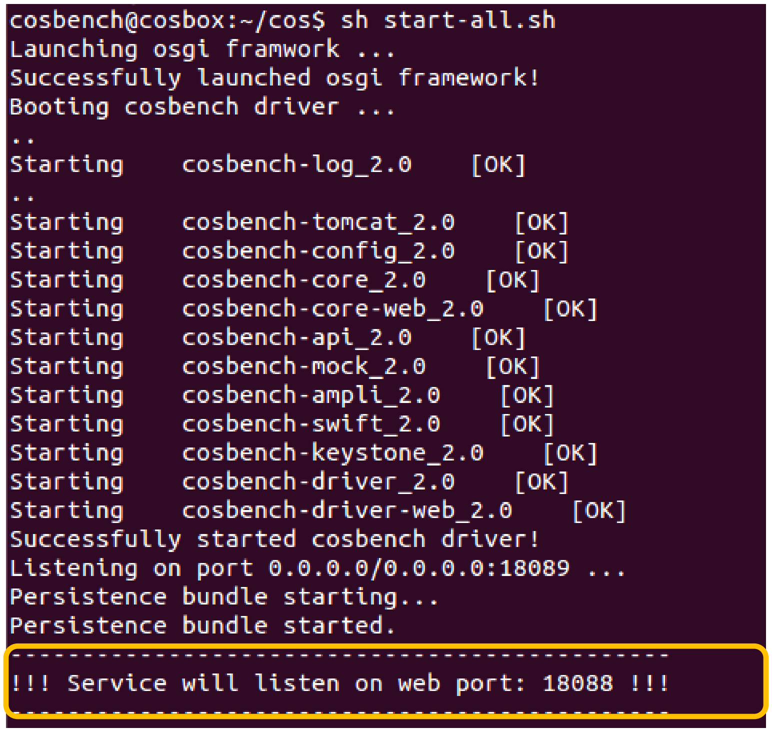
2.4.1 Launch

To avoid http requests routing, need bypass proxy setting:

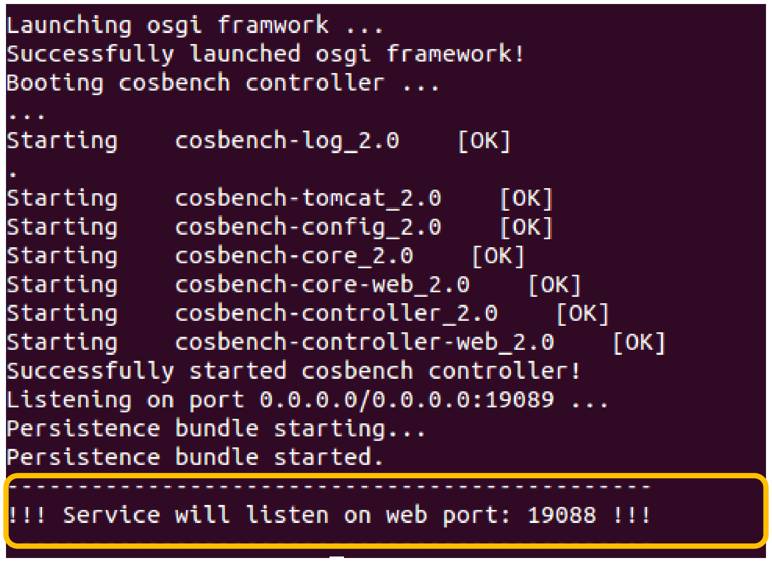
```
cosbench@cosbox:~$ unset http_proxy
```

Then start up COSBench driver and controller on current node. By default, COSBench driver will listen on port 18088, and COSBench controller will listen on port 19088.

```
cosbench@cosbox:~$ sh start-all.sh
```



```
cosbench@cosbox:~/cos$ sh start-all.sh
Launching osgi framwork ...
Successfully launched osgi framework!
Booting cosbench driver ...
..
Starting    cosbench-log_2.0    [OK]
..
Starting    cosbench-tomcat_2.0    [OK]
Starting    cosbench-config_2.0    [OK]
Starting    cosbench-core_2.0    [OK]
Starting    cosbench-core-web_2.0    [OK]
Starting    cosbench-api_2.0    [OK]
Starting    cosbench-mock_2.0    [OK]
Starting    cosbench-ampli_2.0    [OK]
Starting    cosbench-swift_2.0    [OK]
Starting    cosbench-keystone_2.0    [OK]
Starting    cosbench-driver_2.0    [OK]
Starting    cosbench-driver-web_2.0    [OK]
Successfully started cosbench driver!
Listening on port 0.0.0.0/0.0.0.0:18089 ...
Persistence bundle starting...
Persistence bundle started.
----------------------------------------------
!!! Service will listen on web port: 18088 !!!
----------------------------------------------
```



```
Launching osgi framwork ...
Successfully launched osgi framework!
Booting cosbench controller ...
...
Starting    cosbench-log_2.0    [OK]
.
Starting    cosbench-tomcat_2.0    [OK]
Starting    cosbench-config_2.0    [OK]
Starting    cosbench-core_2.0    [OK]
Starting    cosbench-core-web_2.0    [OK]
Starting    cosbench-controller_2.0    [OK]
Starting    cosbench-controller-web_2.0    [OK]
Successfully started cosbench controller!
Listening on port 0.0.0.0/0.0.0.0:19089 ...
Persistence bundle starting...
Persistence bundle started.
----------------------------------------------
!!! Service will listen on web port: 19088 !!!
```

2.4.2 Check

```
cosbench@cosbox:~$ netstat –an |grep LISTEN |grep 19088 # check controller.
tcp        0      0 :::19088                :::*       LISTEN

Cosbench@cosbox:~$ netstat –an |grep LISTEN |grep 18088 # check driver
tcp        0      0 :::18088                :::*       LISTEN
```

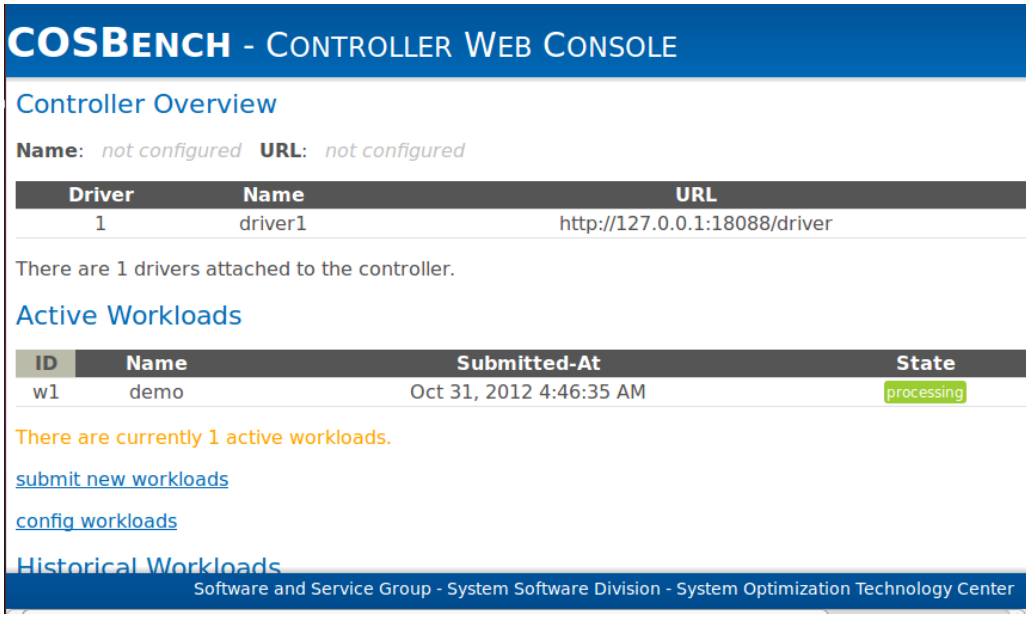
2.4.3 Test

```
Cosbench@cosbox:~$ sh cli.sh submit conf/workload-config.xml # run mock test.
Accepted with ID: w1

cosbench@cosbox:~$ sh cli.sh info
Drivers:
driver1 http://127.0.0.1:18088/driver
Total: 1 drivers

Active Workloads:
W1      Thu Jul 12 04:37:31 MST 2012    PROCESSING
```

Open http://127.0.0.1:19088/controller/index.html in browser to monitor running status. One workload in "processing" state is listing at "active workloads" section.



COSBENCH - CONTROLLER WEB CONSOLE

Controller Overview

Name: *not configured* **URL**: *not configured*

Driver	Name	URL
1	driver1	http://127.0.0.1:18088/driver

There are 1 drivers attached to the controller.

Active Workloads

ID	Name	Submitted-At	State
w1	demo	Oct 31, 2012 4:46:35 AM	processing

There are currently 1 active workloads.

submit new workloads

config workloads

Historical Workloads

Software and Service Group - System Software Division - System Optimization Technology Center

Congratulations! COSBench is successfully installed on current node. It's optionally to cancel workload and stop cosbench as following:

```
cosbench@cosbox:~$ sh cli.sh cancel w1
W1      Thu Jul 12 23:34:14 MST 2012    CANCELLED

cosbench@cosbox:~$ sh stop-all.sh
Stopping cosbench controller ...
Successfully stopped cosbench controller.

=====================================================
Stopping cosbench driver ...
Successfully stopped cosbench driver.
```

2.5 Deploy

- Copy <version>.zip to the rest cosbench nodes through scp, shared folder, etc.
- Do as Installing COSBench does, including sanity check step on all those new nodes.
- If all are done and sanity check is successful, then deployment is completed.

3. Configure & Run

3.1 General

COSBench controller and driver depend on different system configuration files to start up, and those configuration files are only for COSBench itself, not about workload configuration.

The following table gives a quick overview of all the configurations COSBench expects.

Configuration	Description	File Path
controller	the configuration for a controller; read by the controller during its initialization	conf/controller.conf
driver	the configuration for a driver; read by the driver during its initialization	conf/driver.conf
workload	the configuration for a workload being submitted	submitted via controller's web interface

3.2 Controller Configuration

3.2.1 conf/controller.conf

- *Mandatory* for COSBench controller, configuration is an INI format file
- Example

```
[controller]
concurrency=1
drivers=1
log_level = INFO
log_file = log/system.log

[driver1]
name=driver1
```

```
url=http://192.168.10.1:18088/driver
```

- Explanation

[controller]

Parameter	Type	Default	Comment
drivers	integer	1	the number of drivers controlled by this controller
concurrency	Integer	1	how many workloads can be executed simultaneously
log_level	String	“INFO”	“TRACE”, “DEBUG”, “INFO”, “WARN”, “ERROR”
log_file	String	“log/system.log”	Where the log file will store at

The driver section for the ith driver should be named **driver<i>** in order to be recognized.

[driver<i>]

Parameter	Type	Default	Comment
name	String		A label used to identify the driver node. Note that The name of driver is not necessary to be the node’s hostname
url	string		The address to access the driver node.

3.3 Driver Configuration

3.3.1 conf/driver.conf

- *Optional for COSBench driver, COSBench driver can still start up without this configuration file, the only drawback is web console can’t correctly label the driver node. Configuration is an INI format file.*
- *Example*

```
[driver]
name=driver1
url=http://192.168.0.11:18088/driver
```

- *Explanation*

[driver]

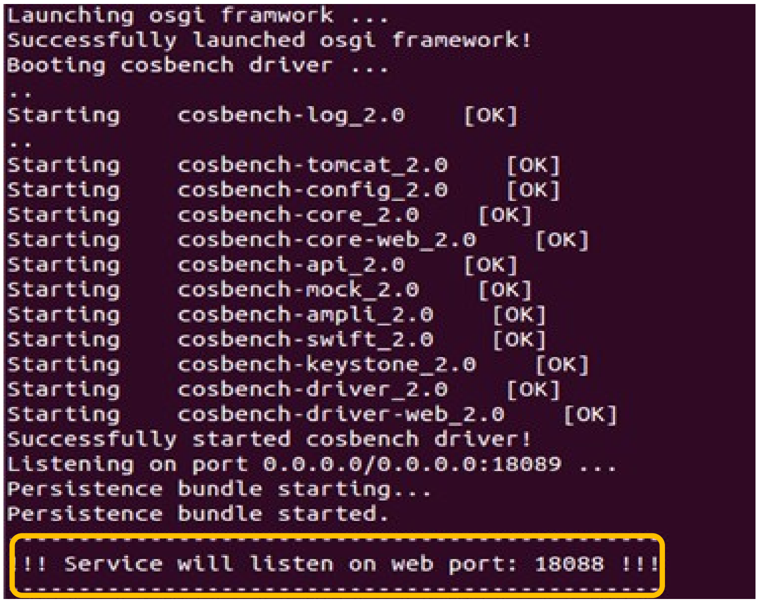
Parameter	Type	Default	Comment
name	String		A label used to identify the driver node. Note that The name of driver is not

			necessary to be the node’s hostname
url	string		The address to access the driver node.

3.4 Start drivers

- Edit conf/driver.conf on driver nodes if user likes to.
- By default, COSBench driver will listen on port 18088.
- Launch Driver on all driver nodes.

```
sh start-driver.sh
```

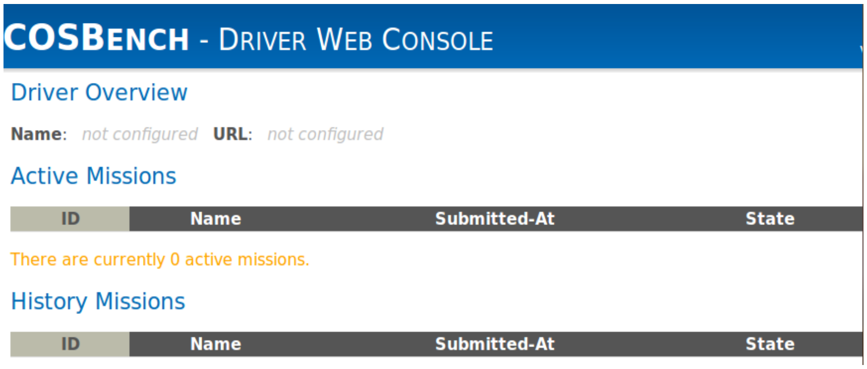


```
Launching osgi framwork ...
Successfully launched osgi framework!
Booting cosbench driver ...
..
Starting    cosbench-log_2.0    [OK]
..
Starting    cosbench-tomcat_2.0    [OK]
Starting    cosbench-config_2.0    [OK]
Starting    cosbench-core_2.0    [OK]
Starting    cosbench-core-web_2.0    [OK]
Starting    cosbench-api_2.0    [OK]
Starting    cosbench-mock_2.0    [OK]
Starting    cosbench-ampli_2.0    [OK]
Starting    cosbench-swift_2.0    [OK]
Starting    cosbench-keystone_2.0    [OK]
Starting    cosbench-driver_2.0    [OK]
Starting    cosbench-driver-web_2.0    [OK]
Successfully started cosbench driver!
Listening on port 0.0.0.0/0.0.0.0:18089 ...
Persistence bundle starting...
Persistence bundle started.
----------------------------------------------------
!! Service will listen on web port: 18088 !!!
----------------------------------------------------
```

- Ensure all drivers are accessible from controller with http connection.
  - By connecting with curl, one valid html file is expected in console, and one valid html file is expected in console.

    ```
    curl http://<driver-host>:18088/driver/index.html
    ```

  - By opening http://driver:18088/driver/index.html in web browser, will see below web page shown.



COSBENCH - DRIVER WEB CONSOLE

Driver Overview

Name: *not configured* **URL**: *not configured*

Active Missions

ID	Name	Submitted-At	State

There are currently 0 active missions.

History Missions

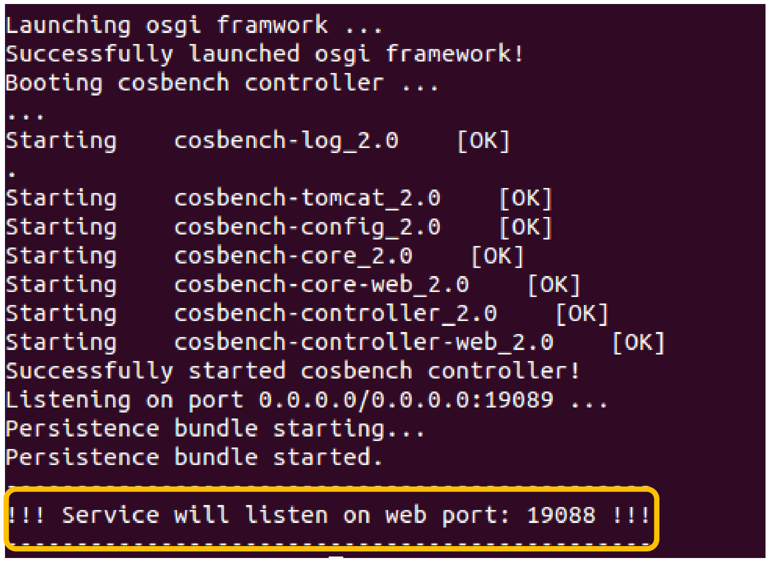
ID	Name	Submitted-At	State

- If any errors or unexpected results shown please check your system configurations, one common issue is firewall filtering or http proxy routing.

3.5 Start controller

- Edit conf/controller.conf on cosbench controller machine.
- By default, COSBench controller will listen on port 19088**.**
- Launch Controller on the controller node.

```
sh start-controller.sh
```

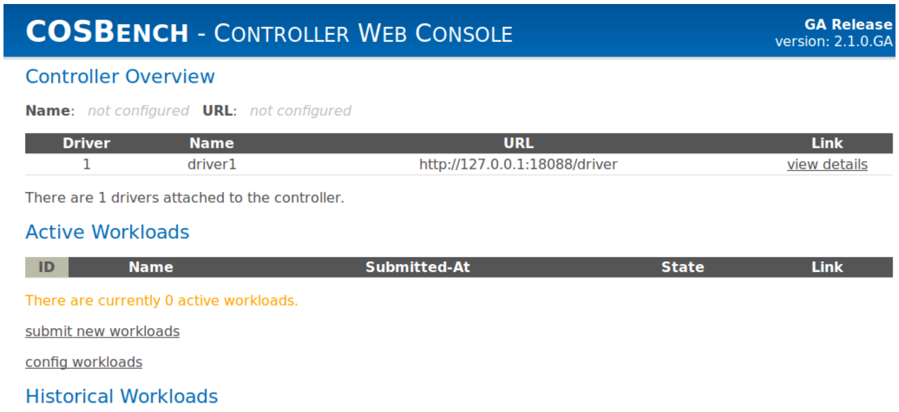


```
Launching osgi framwork ...
Successfully launched osgi framework!
Booting cosbench controller ...
...
Starting    cosbench-log_2.0    [OK]
.
Starting    cosbench-tomcat_2.0    [OK]
Starting    cosbench-config_2.0    [OK]
Starting    cosbench-core_2.0    [OK]
Starting    cosbench-core-web_2.0    [OK]
Starting    cosbench-controller_2.0    [OK]
Starting    cosbench-controller-web_2.0    [OK]
Successfully started cosbench controller!
Listening on port 0.0.0.0/0.0.0.0:19089 ...
Persistence bundle starting...
Persistence bundle started.
----------------------------------------------
!!! Service will listen on web port: 19088 !!!
----------------------------------------------
```

- Ensure the controller is started successfully.
    - By connecting with curl, one valid html file is expected in console, and one valid html file is expected in console.

```
curl http://<controller-host>:19088/controller/index.html
```

- By opening http://<controller-host>:19088/controller/index.html in web browser, will see below web page shown.



COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Controller Overview

Name: *not configured* **URL:** *not configured*

Driver	Name	URL	Link
1	driver1	http://127.0.0.1:18088/driver	view details

There are 1 drivers attached to the controller.

Active Workloads

ID	Name	Submitted-At	State	Link

There are currently 0 active workloads.

submit new workloads

config workloads

Historical Workloads

3.6 Submit workload

There are a few templates in conf/ directory for reference:

workload-config.xml is a template with comments to describe how to configure for different storage type. It will access mock storage to help for sanity check.

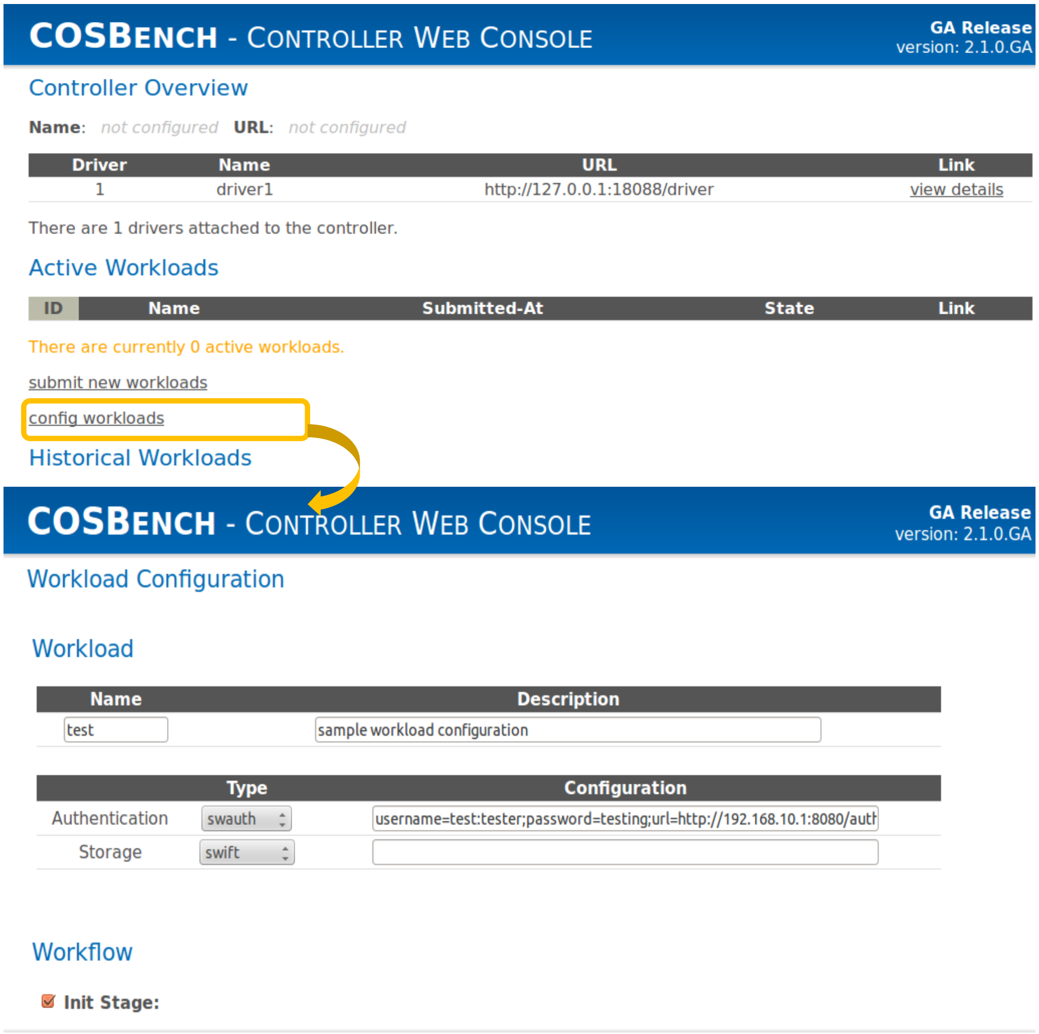
swift-config-sample.xml is a template for openstack swift storage system.

ampli-config-sample.xml is a template for amplidata amplistor storage system.

3.6.1 Define

User can define its own workload based on them, see Workload Configuration to more details about how to write workload config file.

Or it’s possible to define workload through one config editing page on controller web console, so far this page provide basic workload configuration options, please refer to Workload Configuration to customize generated xml file for maximum flexibility.



COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Controller Overview

Name: *not configured* **URL**: *not configured*

Driver	Name	URL	Link
1	driver1	http://127.0.0.1:18088/driver	view details

There are 1 drivers attached to the controller.

Active Workloads

ID	Name	Submitted-At	State	Link

There are currently 0 active workloads.

submit new workloads

config workloads

Historical Workloads

COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Workload Configuration

Workload

Name	Description
test	sample workload configuration

	Type	Configuration
Authentication	swauth	username=test:tester;password=testing;url=http://192.168.10.1:8080/auth
Storage	swift	

Workflow

☑ **Init Stage:**

3.6.2 Submit

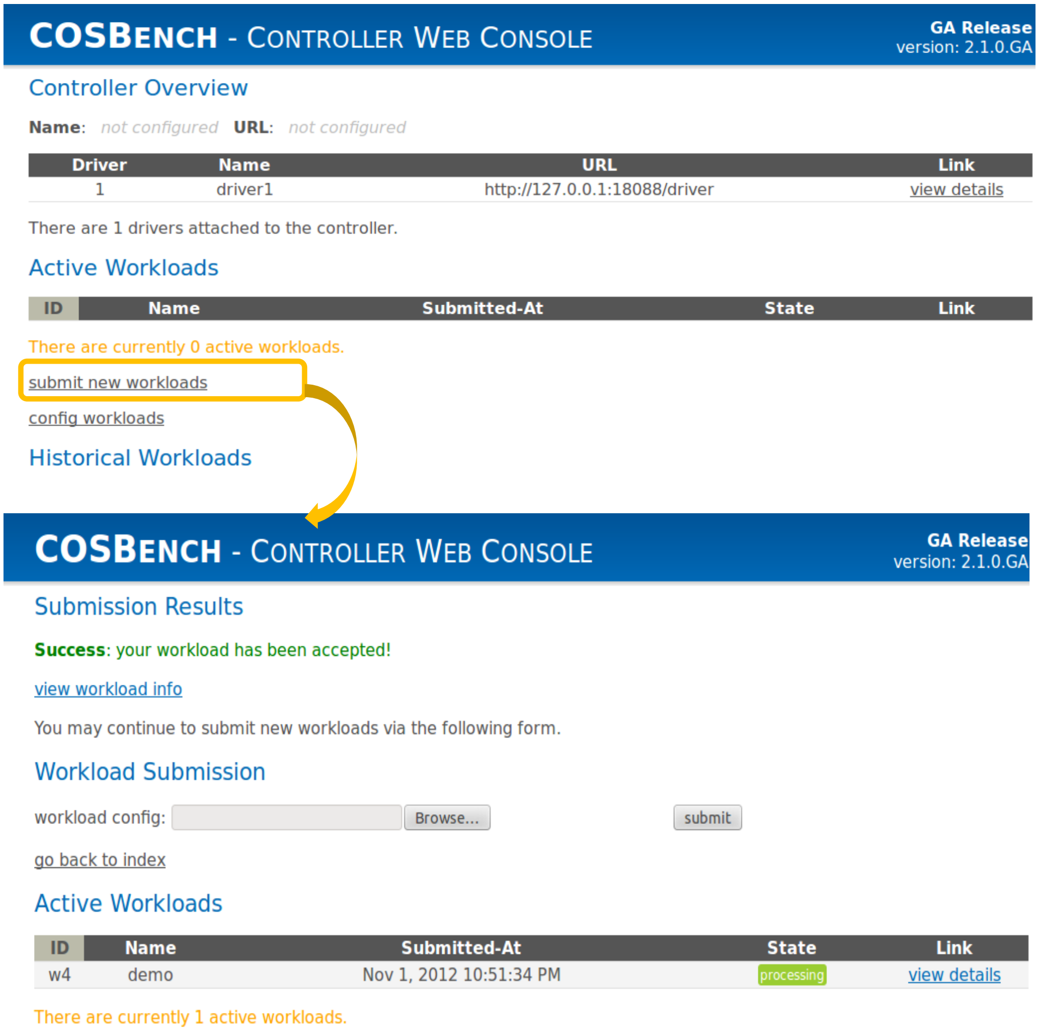
Two ways to submit workloads into COSBench:

3.6.2.1 by command line

```
sh cli.sh submit conf/config.xml
```

3.6.2.1 by web console

Open http://<controller-host>:19088/controller/index.html in browser to monitor running status.



COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Controller Overview

Name: *not configured* **URL**: *not configured*

Driver	Name	URL	Link
1	driver1	http://127.0.0.1:18088/driver	view details

There are 1 drivers attached to the controller.

Active Workloads

ID	Name	Submitted-At	State	Link

There are currently 0 active workloads.

submit new workloads

config workloads

Historical Workloads

COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Submission Results

Success: your workload has been accepted!

view workload info

You may continue to submit new workloads via the following form.

Workload Submission

workload config:

go back to index

Active Workloads

ID	Name	Submitted-At	State	Link
w4	demo	Nov 1, 2012 10:51:34 PM	processing	view details

There are currently 1 active workloads.

3.6.3 Check

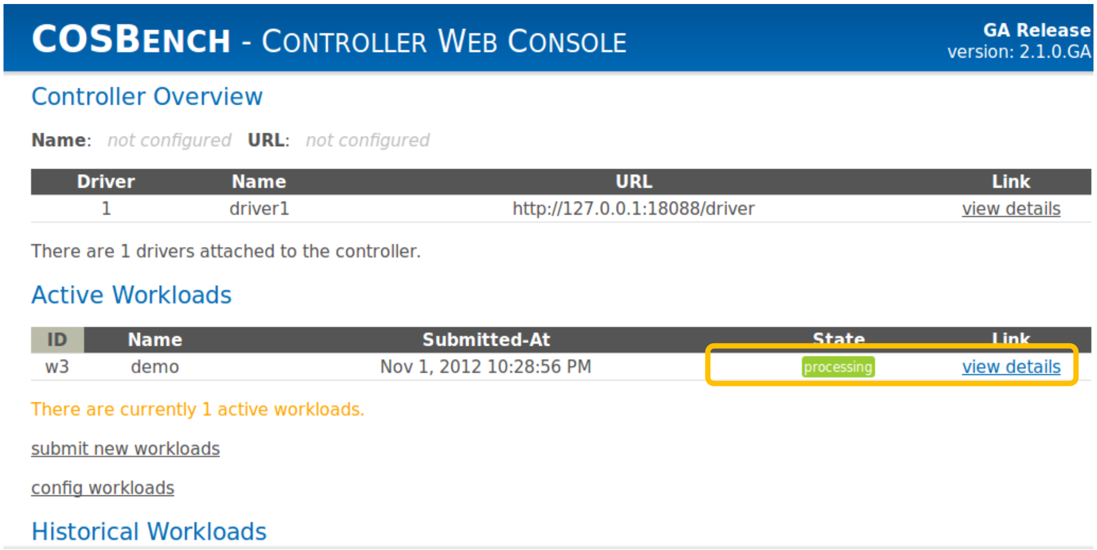
There are also two ways to check workload status:

3.6.3.1 by command line

```
sh cli.sh info
```

3.6.2.3 by web console

Open http://<controller-host>:19088/controller/index.html in browser to monitor running status.



COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

Controller Overview

Name: *not configured* **URL**: *not configured*

Driver	Name	URL	Link
1	driver1	http://127.0.0.1:18088/driver	view details

There are 1 drivers attached to the controller.

Active Workloads

ID	Name	Submitted-At	State	Link
w3	demo	Nov 1, 2012 10:28:56 PM	processing	view details

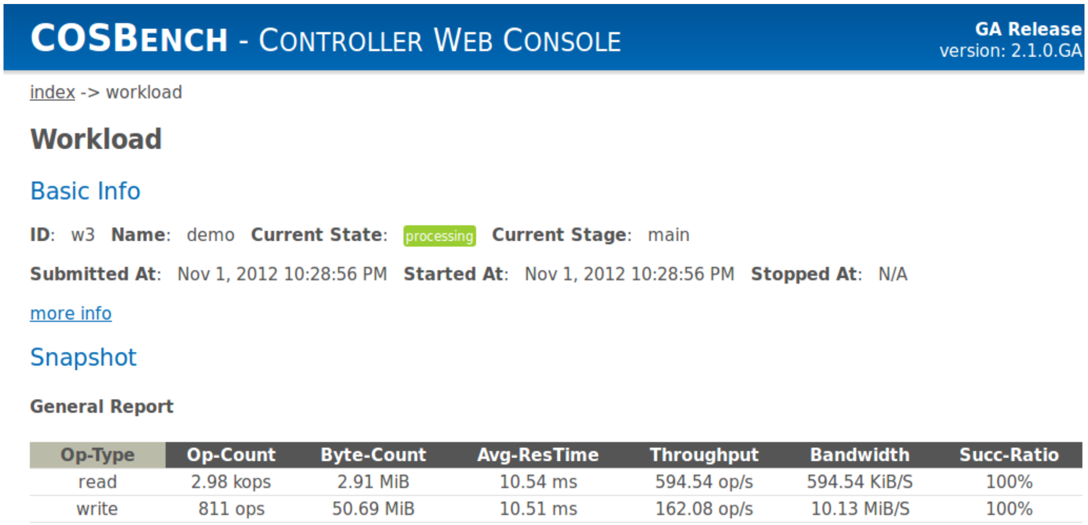
There are currently 1 active workloads.

submit new workloads

config workloads

Historical Workloads

Click details on active workload, you will see runtime performance data.



COSBENCH - CONTROLLER WEB CONSOLE

GA Release
version: 2.1.0.GA

index -> workload

Workload

Basic Info

ID: w3 **Name**: demo **Current State**: processing **Current Stage**: main

Submitted At: Nov 1, 2012 10:28:56 PM **Started At**: Nov 1, 2012 10:28:56 PM **Stopped At**: N/A

more info

Snapshot

General Report

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Throughput	Bandwidth	Succ-Ratio
read	2.98 kops	2.91 MiB	10.54 ms	594.54 op/s	594.54 KiB/S	100%
write	811 ops	50.69 MiB	10.51 ms	162.08 op/s	10.13 MiB/S	100%

3.7 Stop driver, controller

```
ps; sh stop-driver.sh; ps
ps; sh stop-controller.sh; ps
```

If the java process doesn't stop as expected, user needs forcibly stop it by killing the process.

4. Workload Configuration

4.1 Introduction

```
workload
    auth
    storage
```

```
workflow
    workstage
        auth
        storage
        work
            auth
            storage
            operation
```

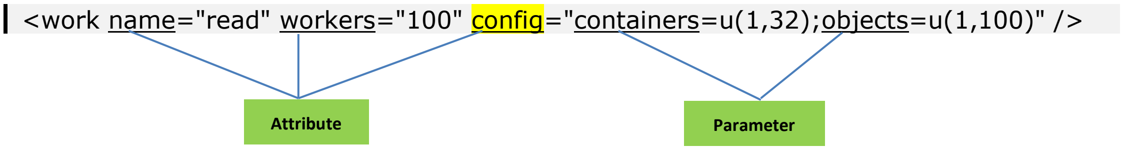
A workload is represented as an XML file having the following structure:

- Workload → work stage → work → operation
- Depending on testing purpose, one workload can define one or more work stages
- Execution of work stage is sequential, while execution of work in the same work stage is in parallel.
- For each work, "workers" is used to tune load.
- Authentication definition (auth) and storage definition (storage) can be defined at a few levels, and lower level definition will overwrite uppers. E.g., if operations will use the definitions for auth and storage at its work instead of those at workload level.

4.2 Selection Expression (aka. Selector)

4.2.1 Overview

- In workload configuration, below elements support one "config" attribute: **auth**, **storage**, **work**, **operation**, the attribute contains optional parameter list with key-value pair form like "a=a_val;b=b_val".



```
<work name="read" workers="100" config="containers=u(1,32);objects=u(1,100)" />
```

- In parameter list, usually used keys are "**containers**", "**objects**", "**sizes**", and they are used to figure out how to select container, object and size. One expression is used to help define selection rule.

- The number in expression has different meaning for object size and object/container. For object size, the number represents a quantity, while for object or container, the number represents a numbering or label.

4.2.2 Selector

Expression	form	Comments
constant	c(number)	Only use specified number. e.g. c(1) means the element numbering will be fixed in one fixed number.

uniform	u(min, max)	Select from [min, max] evenly. e.g. u(1,100) means the element numbering will be selected from the 100 elements evenly. The selection is random, and some numbers may be selected more than one times, while some may be never selected.
range	r(min,max)	Select from [min,max] incrementally. e.g.: r(1,100) means the element numbering will incrementally increase from min to max. Each number will be selected once only. This is only used in special stages (init,prepare,cleanup,dispose).

4.2.3 Available matrix

There are additional constraints for selectors based the element type and work type, below two tables list allowable combinations:

Selector vs Element:

O: OK, X: Not OK	Selector		
Key	constant (c(num))	uniform (u(min,max))	range (r(min,max))
containers	O	O	O
objects	O	O	O
sizes	O	O	X

Selector vs Work:

Key	init	prepare	normal (read)	normal (write)	normal (delete)	cleanup	dispose
containers	r()	r()	c(), u(), r()	c(), u(), r()	c(), u(), r()	r()	r()
objects		r()	c(), u(), r()	c(), u(), r()	c(), u(), r()	r()	
sizes		c(), u()		c(), u()			

4.3 Workload

4.3.1 General form

```
<workload name=”demo” description=”demo benchmark with mock storage” />
```

4.3.2 Attributes

Parameter	Type	Default	Comment
name	string		One name for the workload

description	string		Some additional information

4.4 Auth

4.4.1 General form

```
<auth type="none|mock|swauth|keystone"
config="<key>=<value>;<key>=<value>" />
```

4.4.2 Attributes

Attribute	Type	Default	Comment
type	string	"none	authentication type
config	String		[optional], parameter list

4.4.3 Authentication mechanisms

- ✓ **none:** (do nothing, default)

```
<auth type="none" config="" />
```

Parameter list:

Parameter	Type	Default	Comment
logging	Boolean	false	Print information to logger

- ✓ **mock:** (delay specified time)

```
<auth type="mock" config="" />
```

Parameter list:

Parameter	Type	Default	Comment
token	String	"token"	The token string
delay	Long	20	Delay time in milliseconds.

- ✓ **swauth:** (for openstack swift)

```
<auth type="swauth"
config="username=test:tester;password=testing;url=http://127.0.0.1:8080/auth/
v1.0" />
```

Parameter list:

Parameter	Type	Default	Comment
auth_url	String	"http:/127.0.0.1:8080/auth/v1.0"	The url to auth node.
username	String		The username

			for authentication with account:user form.
password	String		The password for authentication.
timeout	Int	30,000	Connection timeout value in milliseconds.

- ✓ **keystone:** (for openstack swift)

```
<auth type="keystone"
config="username=tester;password=testing;tenant_name=test;url=http://127.0.
0.1:5000/v2.0;service=swift" />
```

Parameter list:

Parameter	Type	Default	Comment
auth_url	String	“http://127.0.0.1:8080/auth/v2.0”	The url to auth node.
username	String		The username for authentication with account:user form.
password	String		The password for authentication.
tenant_name	String		The name of tenant which the user is belonging to.
service	String	“swift”	The service requested.
timeout	Int	30,000	Connection timeout value in milliseconds.

4.5 Storage

4.5.1 General form

```
<storage type="none|mock|swift|ampli" config="<key>=<value>;<key>=<value>" />
```

4.5.2 Attributes

Attribute	Type	Default	Comment

type	string	"none"	Storage type
config	string		[optional] parameter list

4.5.3 Storage systems

✓ **none:** (do nothing, default)

```
<storage type="none" config="" />
```

Parameter list:

Parameter	Type	Default	Comment
logging	Boolean	false	Print information to logger

✓ **mock:** (delay specified time)

```
<storage type="mock" config="" />
```

Parameter list:

Parameter	Type	Default	Comment
logging	Boolean	false	Print information to logger
size	Integer	1024	Object size in bytes
delay	Integer	10	Delay time in milliseconds.
errors	Integer	0	Set error limit to emulate failure.
printing	Boolean	False	Print out data content.

✓ **swift:**

```
<storage type="swift" config="" />
```

Parameter list:

Parameter	Type	Default	Comment
timeout	Int	30,000	Connection timeout value in milliseconds.

✓ **ampli:**

```
<storage
type="ampli"config="host=192.168.10.1;port=8080;nsroot=/namespace;policy=
14195ca863764fd48c281cb95c9bd555" />
```

Parameter list:

Parameter	Type	Default	Comment

timeout	Int	30,000	Connection timeout value in milliseconds.
host	String		The controller node ip to connect
port	Int		The port
nsroot	String	“/namespace”	The namespace root
policy	String		The policy id of the namespace will access.

4.6 WorkStage

4.6.1 General form

```
<workstage name="<name>" >
</workstage>
```

4.6.2 Attributes

Attribute	Type	Default	Comment
name	string		One name for the stage.

4.7 Work

4.7.1 General form

```
<work name="main" type="normal" workers="128" interval="5" division="none"
runtime="60" rampup="0" rampdown="0" totalOps="0" totalBytes="0" config="" > . . .
</work>
```

There are one normal type and 4 special type (init, prepare, cleanup, dispose). Above form is a full set, for different work type, the valid forms are different, below are general rules:

i) *workers* is one key attribute, which normally used to control load.

ii) *runtime* (including *rampup* and *rampdown*), *totalOps* and *totalBytes* are attributes to control how to end the work, called ending options. Only one can be set in a work.

4.7.2 Attributes

Attribute	Type	Default	Comment
name	string		One name for the work
type	String	“normal”	The type of work
workers	Integer		The number of worker to conduct the work in parallel.
interval	Integer	5	The interval between performance snapshots.
division	string	“none”	[“none”|”container”|”object”], The division strategy to

			control how work is spread between worker.
runtime	Integer	0	How many seconds the work will execute
rampup	Integer	0	How many seconds to rampup workload, the time is exclude from runtime.
rampdown	Integer	0	How many second to rampdown workload, the time is excluded from runtime
totalOps	Integer	0	How many operations will execute, it should be multiplier of workers.
totalBytes	Integer	0	How many bytes will transfer, it should be multiplier of the product of workers and size.

4.8 Special work

4.8.1 General form

```
<work type="init|prepare|cleanup|dispose" workers="<number>"
config="<key>=<value>;<key>=<value>" />
```

Comparing to normal work, special work shows a few differences:

i) It internally adopts and calculates “totalOps”, so *no ending option* needs in configuration explicitly.

ii) It has implicitly defined operations, so *no operation* needs.

4.8.2 Supported special work

✓ **init** (creating specific ***containers*** in bulk)

```
<work type="init" workers="4" config="containers=r(1,100)" />
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), r(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix

✓ **prepare** (inserting specific ***objects*** in bulk)

```
<work type="prepare" workers="4"
config="containers=r(1,10);objects=r(1,100);sizes=c(64)KB" />
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix
objects	String		The object selection expression like c(1), u(1,100)
oprefix	String	myobjects_	Object prefix
osuffix	String	<null>	Object suffix
sizes	String		The size selection expression with unit (B|KB/MB/GB), like c(128)KB, u(2,10)MB
chunked	Boolean	False	Upload data in chunked mode or not.
content	String	"random"(default) "zero"	To fill object content with random data or all-zeros.
createContainer	Boolean	False	Create related containers if not exists.
hashCheck	Boolean	False	Do object integrity check related work.

✓ **cleanup** (removing specific ***objects*** in bulk)

```
<work type="cleanup" workers="4" config="containers=r(1,10);objects=r(1,100)"
/>
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix
objects	String		The object selection expression like

			c(1), u(1,100)
oprefix	String	myobjects_	Object prefix
osuffix	String	<null>	Object suffix
deleteContainer	Boolean	False	Delete related containers if exists.

✓ **dispose** (removing specific ***containers*** in bulk)

```
<work type="dispose" workers="4" config="containers=r(1,100)" />
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix

4.9 Operation

4.9.1 General form

```
<operation type="read|write|delete" ratio="<1-100>"
config="<key>=<value>;<key>=<value>" />
```

4.9.2 Attributes

Attribute	Type	Default	Comment
type	string		Operation type
ratio	integer		
division	integer		The division strategy for this operation.
config	string		Parameter list

4.9.3 Supported operations

✓ **container/object naming convention:**

By default, container is named with following form "**mycontainers_**<n>", and object is named with form "**myobjects_**<n>", where <n> is a number which is defined by one selection expression in parameter list.

Container/object naming can be modified through cprefix/csuffix or oprefix/osuffix.

✓ **read**

```
<operation type="read" ratio="70" config="containers=c(1);objects=u(1,100)" />
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix
objects	String		The object selection expression like c(1), u(1,100)
oprefix	String	myobjects_	Object prefix
osuffix	String	<null>	Object suffix
hashCheck	Boolean	False	Do object integrity check related work.

✓ write

```
<operation type="write" ratio="20"
config="containers=c(2);objects=u(1,1000);sizes=c(2)MB" />
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix
objects	String		The object selection expression like c(1), u(1,100)
oprefix	String	myobjects_	Object prefix
osuffix	String	<null>	Object suffix
sizes	String		The size selection expression with unit (B|KB/MB/GB), like c(128)KB, u(2,10)MB
chunked	Boolean	False	Upload data in chunked mode or not.
content	String	“random”(default) ”zero”	To fill object content with random data or all-zeros.

hashCheck	Boolean	False	Do object integrity check related work.

✓ delete

```
<operation type="delete" ratio="10" config="containers=c(2);objects=u(1,1000)"
/>
```

Parameter list:

Parameter	Type	Default	Comment
containers	String		the container selection expression like c(1), u(1,100).
cprefix	String	mycontainers_	Container prefix
csuffix	String	<null>	Container suffix
objects	String		The object selection expression like c(1), u(1,100)
oprefix	String	myobjects_	Object prefix
osuffix	String	<null>	Object suffix

4.9.4 Example

✓ pure read

```
e.g: 100% read, 16 users, 300 seconds
<work name="100r16c30s" workers="16" runtime="300">
    <operation type="read" ratio="100" config="..." />
</work>
```

✓ pure write

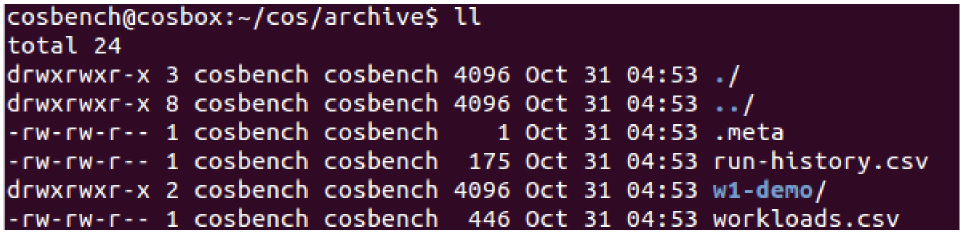
```
e.g.: 100% write, 8 clients, 600 seconds
<work name="100w8c600s workers="8" runtime="600">
    <operation type="write" ratio="100" config="..." />
</work>
```

✓ mixed operations

```
e.g.: 80% read, 20% write, 32 clients, 300 seconds
<work name="80r20w32c300s" workers="32" runtime="300">
    <operation type="read" ratio="80" config="..." />
   <operation type="write" ratio="20" config="..." />
</work>
```

5. Results

All results are stored in "archive" directory.



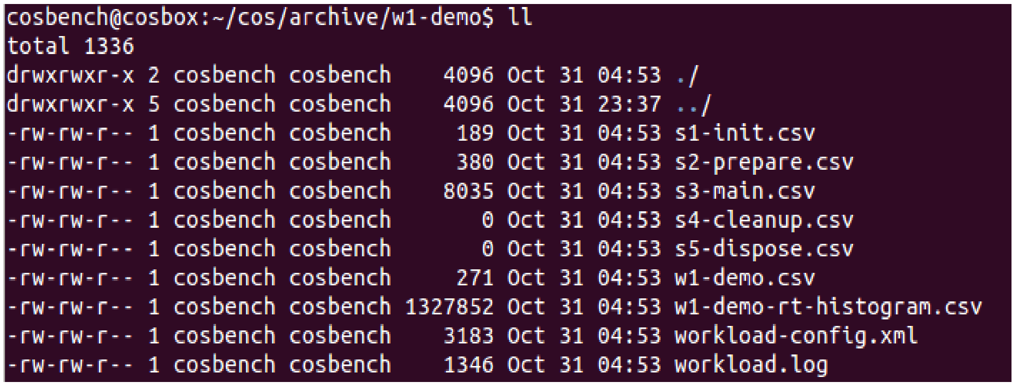
```
cosbench@cosbox:~/cos/archive$ ll
total 24
drwxrwxr-x 3 cosbench cosbench 4096 Oct 31 04:53 ./
drwxrwxr-x 8 cosbench cosbench 4096 Oct 31 04:53 ../
-rw-rw-r-- 1 cosbench cosbench    1 Oct 31 04:53 .meta
-rw-rw-r-- 1 cosbench cosbench  175 Oct 31 04:53 run-history.csv
drwxrwxr-x 2 cosbench cosbench 4096 Oct 31 04:53 w1-demo/
-rw-rw-r-- 1 cosbench cosbench  446 Oct 31 04:53 workloads.csv
```

5.1 Structure

- **.meta**
  - The starting run id.
- **run-history.csv**
  - Record all historical workload runs, including the time and major stages.
- **workload.csv**
  - Record overall performance data for all historical workload runs.
- **Sub-directories**
  - Prefixed with "w<runid>-" store data for each workload run.

5.2 Per Run Data

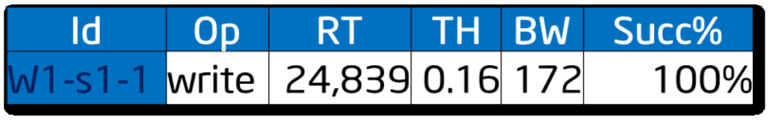
- Below is the a sample per run data list:



```
cosbench@cosbox:~/cos/archive/w1-demo$ ll
total 1336
drwxrwxr-x 2 cosbench cosbench    4096 Oct 31 04:53 ./
drwxrwxr-x 5 cosbench cosbench    4096 Oct 31 23:37 ../
-rw-rw-r-- 1 cosbench cosbench     189 Oct 31 04:53 s1-init.csv
-rw-rw-r-- 1 cosbench cosbench     380 Oct 31 04:53 s2-prepare.csv
-rw-rw-r-- 1 cosbench cosbench    8035 Oct 31 04:53 s3-main.csv
-rw-rw-r-- 1 cosbench cosbench       0 Oct 31 04:53 s4-cleanup.csv
-rw-rw-r-- 1 cosbench cosbench       0 Oct 31 04:53 s5-dispose.csv
-rw-rw-r-- 1 cosbench cosbench     271 Oct 31 04:53 w1-demo.csv
-rw-rw-r-- 1 cosbench cosbench 1327852 Oct 31 04:53 w1-demo-rt-histogram.csv
-rw-rw-r-- 1 cosbench cosbench    3183 Oct 31 04:53 workload-config.xml
-rw-rw-r-- 1 cosbench cosbench    1346 Oct 31 04:53 workload.log
```

5.2.1 Overall performance data (like w1-demo.csv)

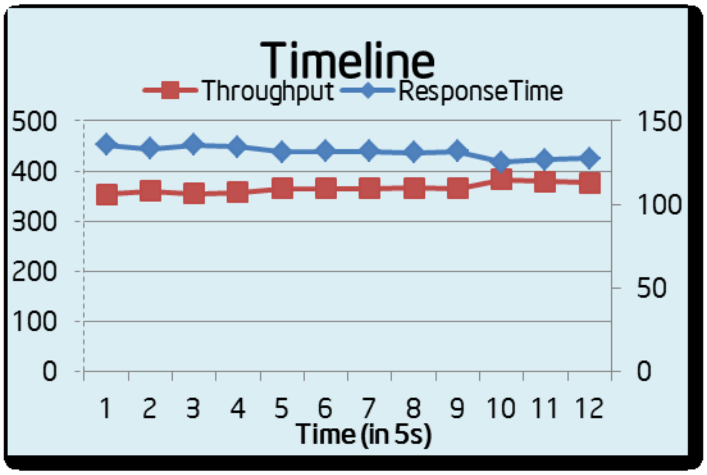
One line per each stage.



Id	Op	RT	TH	BW	Succ%
W1-s1-1	write	24,839	0.16	172	100%

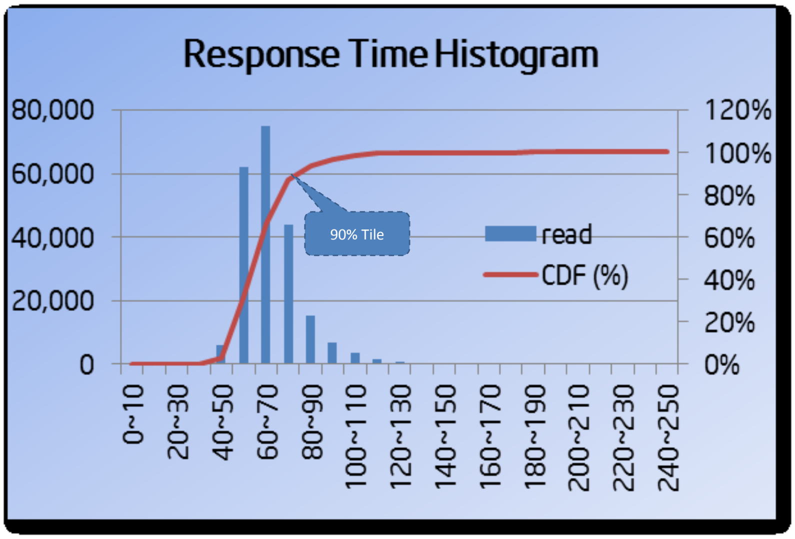
5.2.2 Timeline data (like s3-main.csv)

One file per each stage, can import into excel to draw timeline chart.



5.2.3 Response time histogram data (like w1-demo-rt-histogram.csv)

- One indicator to understand Quality of Service is to know the distribution of response time; histogram data is generated to cover this purpose. The data is bucketed from 0 to 500,000 ms with 10ms as stepping.
- In histogram diagram, the bar represents the count of samples which are falling into each bucket, the curve is the Cumulative Distribution Function (aka CDF), which can help answer question like what's the response time at 90% percentile.



5.2.4 Workload-config.xml

- The workload configuration file used in this run.

5.2.5 Workload.log

- The run time log, which is helpful to help trouble shooting when error rises.

5.3 Metrics

5.3.1 Throughput (Operations/s or Op/s)

- The operations completed in one second
- The reported Throughput is calculated by dividing total successful requests with total run time.

5.3.2 Response Time (in ms)

- The duration between operation initiation and completion.
- The reported Response Time is the average of response time for each successful request.

5.3.3 Bandwidth (MiB/s)

- the total data in MiB transferred in one second,
- the reported bandwidth is calculated by dividing total bytes transferred with total run time,
- The unit is in MiB/s, here 1 MiB = 1024*1024 bytes.

5.3.4 Success Ratio (%)

- the ratio of successful operations,
- The reported success ratio is calculated by dividing total successful requests with total requests.

5.3.5 Other metrics

- Op-count: the total number of operations.
- Byte-count: the total data transferred.

6. FAQ

6.1 General

1. **Is listening port 19088/18088 configurable, how to do?**
   Yes, conf/controller-tomcat-server.xml records the port will be used for controller, and driver-tomcat-server.xml records the port will be used for driver.

2. **What is the difference between "cancelled" and "terminated"**
   "cancelled" means the workload is cancelled by user at running, while "terminated" suggests some errors occur during running, need user's interception to resolve issue.

3. **Can I submit multiple workloads, are they running sequentially?**
   Yes, cosbench can accept multiple workloads in one time, and run them one by one.

4. **Is it possible to cancel one queuing workload?**
   No, cancellation is only for running workload.

5. **Can COSBench be installed on other linux distro like RedHat?**
   Yes, in prior to v2.1, we support redhat EL6 by default, and starting from v2.1, we adopt ubunut 12.04.1 LTS as default OS.

6. **Need a way to re-use the files from a previous test without removing/cleaning up the old files.**
   It's certainly supported, the special stages like init/prepare/cleanup/dispose are all optional, if user wants to keep created objects, he/she can remove related special stages.

7. **Is it possible to define multiple main stages?**
   Yes, and to avoid name confusion, better to name them with different labels.

8. **If there are errors occurred at running workloads, where to see more details?**
   There is one workload.log under that workload's corresponding folder (archive\<workload id>\workload.log), looking at this file to see what's the reason of errors.

9. **Why test is stucked at init stage?**

Check if all cosbench machines are accessible through http connection by using curl to access the url like "curl http://<controller-host>:19088/controller/index.html" or "curl http://<driver-host>:18088/driver/index.html". If firewall is set and has blocked http connection, user needs to open the ports on firewall, for controller node, the ports are 19088 and 19089, for driver nodes, the ports are 18088 and 18089.

10. **Is there tool to distribute cosbench on multiple nodes?**
COSBench itself doesn't provide this kind of tool for package distribution, actually there are many existed solution for this purpose, like scp, shared folder (samba)…

11. **Why cosbench shows one workload test is completed even some errors in workload.log?**
This is by design, for "init/prepare/cleanup/dispose" work, we treat them as supportive work, and there may be side effect if not successfully executes them, so to get "completed" status, the operations embedded into those work should be 100% successful, otherwise, it will be in "terminated" state.

While for normal work, the work is for performance measurement, and should tolerate failures, and "success ratio" is to track failures, so even there are errors, the work is still in "completed" state.

12. **Are there any recommendations for workers number in "init","prepare","cleanup" and "dispose"?**
"init" and "dispose" work are to create or delete containers, the effort is very trivial, it can be done very fast even with 1 worker. Generally, we suggest the number of containers defined in those work should be a multiplier of "workers".
"prepare" and "cleanup" work are to create or delete objects, depending on how many objects to be filled into, the time can be short or long. Generally, we suggest the number of objects defined in those work should be a multiplier of "workers". To shorten the time, user can increase "workers" appropriately to accelerate the procedure.
For main stage, as this stage is for performance measurement, we need tune the workers parameter to control load to storage system, normally the number is big if wanting to stress performance to peak, one best practice is to gradually increase worker number to find out turn-around point of performance.

13. **Received “OutOfMemory” error from driver after running cosbench for a long time**
    In “cosbench-start.sh” script, user can add parameter “-Xmx2g” to limit maximum heap size for the java process to 2 GigaBytes or other number to avoid memory exhausting.

6.2 Swift

1. **For 1G object size, the prepare time takes too long.**
   If total count of large objects is a big number, and the network bandwidth is saturated, long time should be expected. If network bandwidth is not saturated, one action can take is to increase parameter “workers” to a larger number to well utilize bandwidth.

2. **Are there any special changes for operating with large object size like 1GB?**
   For testing with large object size,
   i) Normally longer ramp-up time (the parameter “rampup”) needs to drive performance to peak; longer run time (the parameter “runtime”) needs to get more consistent results. How long both parameters should be set depends on the system under test, and user needs take pre-test to get them.
   ii) Longer time for each operation to complete, and more likely to see timeout error, so it’s necessary to add one “timeout” parameter in “config” part, it’s millisecond in unit. For swift, it looks like:
   ```
   <storage type="swift" config=”timeout=100000” />
   ```
   iii) Another aspect should take care is at the system under test inside, and it’s out of the scope of this document.

6.3 AmpliStor

1. **Where does the system get the string for the policy in the .xml file?**
   Please consult amplidata support to know how to get policy string from amplistor.

2. **Is object range affecting performance?**

From our experience, enlarging object range can improve write performance, as larger range reduces write conflictions.

Appendix A. Example configuration

Swift

The workload configuration describes following test scenario:

i) The test includes five stages: init/prepare/main/cleanup/dispose.
ii) The test creates 32 containers which contains 50 64KB objects.
iii) The operation requests are issued to 3 controller nodes.
iv) The requests include 80% GET(read) operations and 20% PUT(write) operations, and read operations randomly request object in the 50 objects from #1 to #50, while write operations randomly create object with object numbering from #51 to #100.
v) After finished, the test cleans up all those objects and drops all containers.

If expecting to use keystone authentication, just use the commented keystone authentication line as sample.

```
<?xml version="1.0" encoding="UTF-8" ?>
<workload name="swift-sample" description="sample benchmark for swift">

  <storage type="swift" />

  <!-- MODIFY ME -->
  <auth type="swauth"
config="username=test:tester;password=testing;auth_url=http://192.168.10.1:8080/au
th/v1.0" />

  <!-- Keystone Authentication
  <auth type="keystone"
config="username=tester;password=testing;tenant_name=test;auth_url=http://127.0.0.
1:5000/v2.0;service=swift" />
  -->

  <workflow>

    <workstage name="init">
      <work type="init" workers="1" config="containers=r(1,32)" />
    </workstage>

    <workstage name="prepare">
      <work type="prepare" workers="1"
config="containers=r(1,32);objects=r(1,50);sizes=c(64)KB" />
    </workstage>

    <workstage name="main">
      <work name="main" workers="8" rampup="100" runtime="300">
```

```
        <operation type="read" ratio="80"
config="containers=u(1,32);objects=u(1,50)" />
        <operation type="write" ratio="20"
config="containers=u(1,32);objects=u(51,100);sizes=c(64)KB" />
      </work>
    </workstage>

    <workstage name="cleanup">
      <work type="cleanup" workers="1"
config="containers=r(1,32);objects=r(1,100)" />
    </workstage>

    <workstage name="dispose">
      <work type="dispose" workers="1" config="containers=r(1,32)" />
    </workstage>

  </workflow>

</workload>
```

Amplistor

The workload configuration describes following test scenario:

i) The test includes five stages: init/prepare/main/cleanup/dispose.
ii) The test creates 32 containers (namespaces) which contains 50 64KB objects.
iii) The operation requests are issued to 3 controller nodes.
iv) The requests include 80% GET(read) operations and 20% PUT(write) operations, and read operations randomly request object in the 50 objects from #1 to #50, while write operations randomly create object with object numbering from #51 to #100.
v) After finished, the test cleans up all those objects and drops all containers (namespaces).

For amplistor v2.5 release, “*nsroot=/manage/namespace*” is necessary for all namespace related work (init/dispose), for prior to v2.5 release, just remove below “nsroot=/manage/namespace” snippets.

```
<?xml version="1.0" encoding="UTF-8" ?>
<workload name="ampli-sample" description="sample benchmark for amplistor">

  <storage type="ampli"
config="host=192.168.10.1;port=8080;policy=14195ca863764fd48c281cb95c9bd555" />
```

```
  <workflow>

    <workstage name="init">
     <storage type="ampli"
config="host=192.168.10.1;port=8080;nsroot=/manage/namespace;policy=14195ca863764f
d48c281cb95c9bd555" />
     <work type="init" workers="1" config="containers=r(1,32)" />
    </workstage>

    <workstage name="prepare">
      <work type="prepare" workers="1"
config="containers=r(1,32);objects=r(1,50);sizes=c(64)KB" />
    </workstage>

    <workstage name="main">
      <work name="c1p0" workers="16" rampup="100" runtime="300">
        <storage type="ampli"
config="host=192.168.10.1;port=8080;policy=14195ca863764fd48c281cb95c9bd555" />
        <operation type="read" ratio="80"
config="containers=u(1,32);objects=u(1,50)" />
        <operation type="write" ratio="20"
config="containers=u(1,32);objects=u(51,100);sizes=c(64)KB" />
      </work>

      <work name="c2p0" workers="16" rampup="100" runtime="300">
        <storage type="ampli"
config="host=192.168.10.2;port=8080;policy=14195ca863764fd48c281cb95c9bd555" />
        <operation type="read" ratio="80"
config="containers=u(1,32);objects=u(1,50)" />
        <operation type="write" ratio="20"
config="containers=u(1,32);objects=u(51,100);sizes=c(64)KB" />
      </work>
      <work name="c3p0" workers="16" rampup="100" runtime="300">
        <storage type="ampli"
config="host=192.168.10.3;port=8080;policy=14195ca863764fd48c281cb95c9bd555" />
        <operation type="read" ratio="80"
config="containers=u(1,32);objects=u(1,50)" />
        <operation type="write" ratio="20"
config="containers=u(1,32);objects=u(51,100);sizes=c(64)KB" />
      </work>
    </workstage>

    <workstage name="cleanup">
      <work type="cleanup" workers="1"
config="containers=r(1,32);objects=r(1,100)" />
    </workstage>

    <workstage name="dispose">
     <storage type="ampli"
config="host=192.168.10.1;port=8080;nsroot=/manage/namespace;policy=14195ca863764f
d48c281cb95c9bd555" />
```

```
      <work type="dispose" workers="1" config="containers=r(1,32)" />
    </workstage>

  </workflow>

</workload>
```