Caliper User Manual	Public
1.0	Total 19 pages

Caliper [Open source benchmarking framework]
User Manual

Shyju PV (shyju.pv@gmail.com)

Table of Contents

1 What is Caliper?

Caliper is a benchmarking framework for **platforms**, integrated with industry standard tools and test cases. New test tools/benchmarks can be easily integrated into caliper to extend it further. It provides detailed performance assessments for **server platforms** based on ARM64 and x86_64. The **uniqueness** of Caliper is a consolidated score based evaluation. The best performance in each test will get a score of 100. All other test scores are calculated relative to this. This provides a clear relative positioning of each target platform. It enables easy analysis of the performance levels and bottlenecks.

How to use Caliper Report

- Caliper executes a set of benchmarking and performance test tools on the target and the results are represented in html web page.
- The caliper report can be used by the developers and architects to identify the bottleneck in the platform. So they can tune the system for better performance.
- Caliper gives the relative positions of the platforms evaluated this makes the comparison of the platforms easy with respect to one another and assess the maturity of the same.

2 Caliper Organization

Caliper mainly builds on the existing open source benchmarks and test suites. It can be divided into two parts:

- **Functional testing** – Caliper's functional testing mainly uses the LTP (Linux Test Project), which contains a lot of tests focused on systems' functions and features testing at the kernel level.
- **Performance testing** – Caliper's performance assessment includes the following aspects: CPU, memory, network, storage, application and so on. Caliper mainly includes well known open source tools and benchmarks to evaluate the performance. Certain use-case scenarios or workloads are also included to study the system behaviour. For example: the compilation performance, compression decompression performance etc. Caliper is also flexible to add any tool binary and execute it along with other set of benchmarking tools if the source code is not open.

3 Caliper Setup

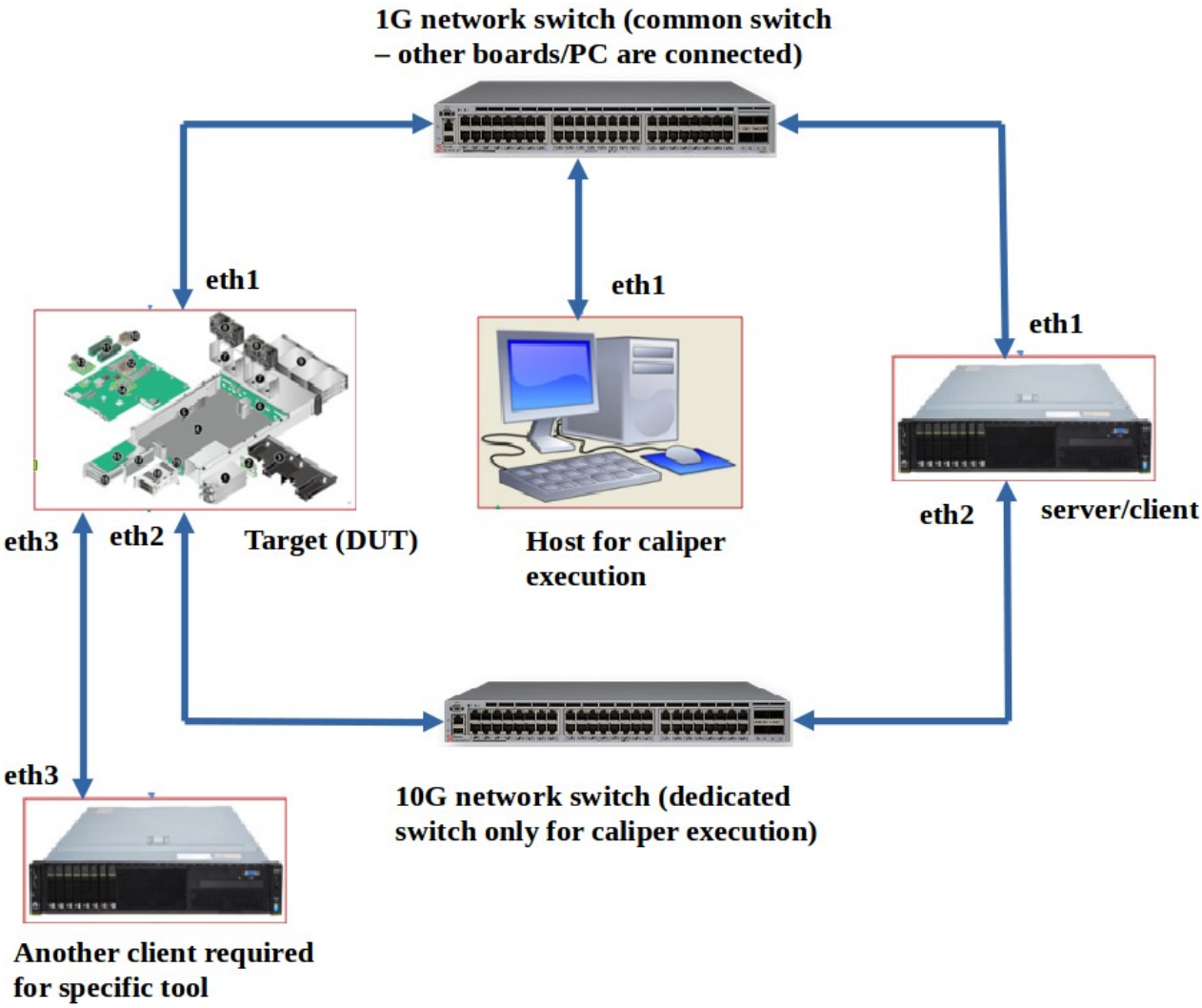


Figure 1: Caliper Setup

Host: caliper script will be installed and triggered on this system

Target (DUT): All the benchmarking tools will executes on this system.

Server/Client: Depending on the requirement of the use case, this system will work as

- Server: Where target would connect for specific services.

  Example: Network daemon services like iperf and netperf

- Client: Where the system will request application services from the targets.

  Example: Sysbench client where the target will be bombarded with queries from the client.

This system named as "**TestNode**" in caliper framework.

3.1 OS Requirements

Host OS Requirements: Caliper supports x86_64 architecture and Ubuntu operating system. Supported Ubuntu versions are 14.04, 16.04 and 16.04.1.

Target OS Requirements: It supports x86_64 and arm64 architectures and operating systems are Ubuntu and Centos. Supported centos version is 7.0.

TestNode OS Requirements: It supports x86_64 architecture and operating system is Ubuntu.

3.2 Download Caliper

cd $HOME

git clone https://github.com/open-estuary/caliper

This will download a directory named caliper in your current operation directory.

Enter the test suite: ***cd caliper***

3.3 Software Requirements

There are certain dependencies to be installed before caliper runs. This section explains about those dependencies.

3.3.1 Host Dependency

Install **expect** package in the Host PC using following command:

- ***sudo apt-get install expect***

Host PC should not use root login for caliper execution or installation. Normal user login should be used for caliper installation and execution with sudo privileges.

If user is using Ubuntu 16.04 as host PC, following packages are required to install manually:

- ***sudo apt-get install libpng-dev libfreetype6-dev***

Run following commands in the host PC:

- ***cd utils/automation_scripts/Scripts***
- ***./host_dependency.exp <package_installation_choice> <host_pc_password>***

host_pc_password indicates host pc password with sudo privileges.

package_installation_choice indicates whether user wants to install dependency packages. Possible value can be "y" or "n".

Example: ***./host_dependency.exp y password123***

After successful execution of this script, host_dependency_output_summary.txt will be created in host PC at ~/caliper/utils/automation_scripts/Scripts/host_dependency_dir/ to view dependency packages installation summary.

Pre-requisite for target and TestNode platform:

- Install ***openssh-server*** package**.**

Update /etc/ssh/sshd_config file with this command: ***PermitrootLogin yes***

Auto-login to Target and TestNode:

- Generate a public key: ***ssh-keygen -t rsa , please enter for all the prompts.***
- Copy the public key of Host to the target: ***ssh-copy-id -i ~/.ssh/id_rsa.pub root@target_ip***
- Copy the public key of Host to the TestNode: ***ssh-copy-id -i ~/.ssh/id_rsa.pub root@TestNode_ip***
- Copy the public key of Host to the Clients: ***ssh-copy-id -i ~/.ssh/id_rsa.pub root@Client_ip***

Note:

Its better to check the autossh connection by running " ssh root@ipaddress" to verify autossh is enabled from host.

TestNode is a PC/platform to execute network tools (iperf, qperf, etc.) and application tools (nginx, redis).

3.3.2 Target Dependency

Run following commands in the host PC:

- ***cd utils/automation_scripts/Scripts***
- ***./target_dependency.exp <package_installation_choice> <target_user> <target_ip> <target_password> <disk_name>***

package_installation_choice indicates whether user wants to install dependency packages. Possible value can be "y" or "n".

target_user should have root priviledges.

disk_name is the disk to be used for storage subsystem testing (fio and iozone tool). For example, if sdb disk is free then use /dev/sdb as disk name.

Please log in to the target and identify the disk to be used for testing. "lsblk" is a handy command.

Example: ***./target_dependency.exp y root 192.168.40.1 passwd /dev/sdb***

Note: If user faced following type of error: "Could not get lock /var/lib/dpkg/lock" in ubuntu target, then execute below commands on target:

sudo rm /var/lib/apt/lists/lock
sudo rm /var/cache/apt/archives/lock
sudo rm /var/lib/dpkg/lock

After successful execution of this script, target_dependency_output_summary.txt will be created in target. Copy the summary file from target to host PC by following command for further reference:

- ***scp target_user@target_ip:~/target_dependency_dir/target_dependency_output_summary.txt target_dependency_output_summary.txt***

Open this file to view dependency packages installation summary. Please take necessary steps if any of the tools /packages are not correctly installed in the target.

Example:

scp root@192.168.40.1:~/target_dependency_dir/target_dependency_output_summary.txt target_dependency_output_summary.txt

For Ubuntu target, install mysql-server and libmysqlclient-dev package manually with following command in target:

- ***ssh root@<target_ip>***
- ***sudo apt-get install mysql-server libmysqlclient-dev -y***
  give mysql server password as "root"
- ***exit***

For Centos target, install mysql server package manually from below steps in target:

- ***ssh root@<target_ip>***
- ***wget http://www.estuarydev.org/caliper/mysql-5.6.34.tar.gz***
- ***tar -xvzf mysql-5.6.34.tar.gz***

- ***cd mysql-5.6.34***
- ***cmake .***
- ***make***
- ***make install***
- ***useradd -r -g root mysql***
- ***cd /usr/local***
- ***chgrp -R root mysql***
- ***chown -R root mysql***
- ***chown -R mysql mysql/data***
- ***cd mysql***
- ***./scripts/mysql_install_db --user=mysql***
- ***/etc/my.cnf file should be updated with below content:***

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
# Recommended in standard MySQL setup
sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES
[mysqld_safe]
log-error=/var/log/mysqld.log
ipid-file=/var/run/mysqld/mysqld.pid
[client]
#password       = your_password
host            = 127.0.0.1
port            = 3306
socket          = /var/run/mysql/mysql.sock
```

- ***./bin/mysqld_safe &***
- ***./bin/mysql_secure_installation***
- ***Give password as "root".***
- ***Give user input as 'n' for all options.***
- ***sudo yum install mysql-devel***
- ***exit***

FIXME: These steps to be moved to script

Note:

1. On Ubunbtu 16.04 target, openjdk-7-jdk and openjdk-7-jre packages are not supported. So install openjdk-8-jdk and openjdk-8-jdk packages on it.

>> sudo apt-get install openjdk-8-jdk openjdk-8-jdk ***-y***

2. If user wants to install all the packages manually in host, TestNode and target platforms, please refer section 9.

3. Target, host and TestNode should be connected to the Internet. Target should be rebooted before executing caliper.

3.3.3 TestNode Dependency

Run following commands in the host PC:

- ***cd utils/automation_scripts/Scripts***
- ***./TestNode_dependency.exp <package_installation_choice> <TestNode_user> <TestNode_ip> <TestNode_password>***

package_installation_choice indicates whether user wants to install dependency packages. Possible value can be "y" or "n".

TestNode_user should have root priviledges.

Example: ***./TestNode_dependency.exp y root 192.168.40.1 root***

After successful execution of this script, TestNode_dependency_output_summary.txt will be created in TestNode.

Copy the summary file from TestNode to host PC by following command:

- ***scp TestNode_user@TestNode_ip:~/TestNode_dependency_dir/TestNode_dependency_output_summary.txt TestNode_dependency_output_summary.txt***

Open this file to view dependency packages installation summary. For more details of logs please see *~/installation_log.txt* file on TestNode.

Example:

- ***scp root@192.168.40.1:~/TestNode_dependency_dir/TestNode_dependency_output_summary.txt TestNode_dependency_output_summary.txt***

Run this command on TestNode platform:

Please login to TestNode and execute the following commands to run the network services.

iperf3 -s & (if user wants to execute iperf tool)

qperf & (if user wants to execute qperf tool)

3.3.4 Toolchain Requirements

Tool-chain should be installed in the host platform.

To generate the binary file of benchmarking tools for arm64 target, it requires arm tool-chain. Download the ARM-64 tool-chain from below command:

- ***cd $HOME***
- ***wget http://www.estuarydev.org/caliper/gcc-linaro-aarch64-linux-gnu-4.9-2014.09_linux.zip***
- ***unzip gcc-linaro-aarch64-linux-gnu-4.9-2014.09_linux.zip***

Export the toolchain by adding below line in ~/.bashrc file at end of file:

- ***export PATH=$HOME/gcc-linaro-aarch64-linux-gnu-4.9-2014.09_linux/bin:$PATH***
- ***source ~/.bashrc***

3.4 Install Caliper

Move into caliper source directory and execute below command:

- sudo python setup.py install

4 Caliper Configurations

To increase the flexibility caliper has different options that can be customized to suite the test requirements. Please make sure you pay attention to the following configure options.

4.1 Target Configuration

4.1.1 Platform configuration

Configure the ***~/caliper_output/configuration/config/client_config.cfg*** file to set up the "Target" (DUT), "TestNode" and "nginx" clients.

```
[TARGET]
#If Platform_name is not specified then the hostname of the target is taken
Platform_name:
ip:
port:
user:
password:
target_ip_10g:

[TestNode]
ip:
port:
user:
password:
TestNode_ip_10g:
# SSH implementation used by server (ssh or paramiko)
ssh_engine: raw_ssh
# enable OpenSSH connection sharing. Only useful if engine_ssh is 'raw_ssh'
enable_master_ssh: True

[nginx]
no_of_clients:
client_1_ip:
client_1_user:
client_1_password:
client_2_ip:
client_2_user:
client_2_password:
target_ip_1_10g:
target_ip_2_10g:
target_port_1:
target_port_2:
```

The field description of "TARGET", "TestNode" and "nginx" section are as below:

- **TARGET**: Give proper name in "**platform_name**" field. The name should not contain '_' character. Give 1G network IP address (eth1 as per Figure 1) in the "**ip**" field. Give "**port**" as 22, "**user**" as user name and "**password**" as None. Give 10G network IP address (eth2 as per Figure 1) in "**target_ip_10g**" field.

- **TestNode**: Give 1G network IP address (eth1 as per Figure 1) in "**ip**" field. Give "**port**" as unused port number in TestNode. Give "**user**" as user name and "**password**" as user password. Give 10G network IP address (eth2 as per Figure 1) in "**TestNode_ip_10g**" field.

- For "nginx" section information, refer **nginx user manual.**

Note:

1. Guideline to select port number in **TestNode** section:

   First in TestNode platform, check whether "server.py" process is running.

   If the process is running, then capture the port number used by "server.py" process.

   Example: search the process "server.py" using this command: ***ps -ef | grep server.py***

   The output of above command will be: ***root 2914 876 0 ? 00:00:00 python /root/caliper_server/server.py 5000.*** 5000 is the port number used by server.py process. So use 5000 port number in client_config.cfg file.

   If "server.py" process is not running in TestNode, then execute "sudo netstat –tulp | grep LISTEN" command to find out which ports are currently in use. Select the un-used port number and use this port number in client_config.cfg file.

2. For multi-target execution, port number in the **TestNode** section should be same for all target configuration files (client_config.cfg file).

4.1.2 Email configuration

Configure the ***~/caliper_output/configuration/config/email_config.cfg*** file to set up the email information.

```
[email_info]
from = "huaweisend@gmail.com"
to = "huaweireceive@gmail.com"
subject = "email_test"
plaintext = "please find the attached doc"

[login_info]
user = huaweisend
password = huaweisend_password
server = smtp.gmail.com
```

Figure 2: Sample email_config.cfg file

- If user wants to send the mail regarding caliper execution, then configure this file as described in above figure.

4.2 Tool Configuration

The snapshot of test_cases_cfg directory structure is given below:

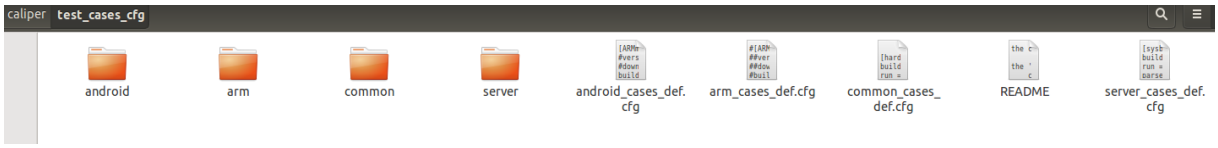


Figure 3: test_cases_cfg directory structure

Configure the *.cfg files located in **~/caliper_output/configuration/test_cases_cfg/XXXX_cases_def.cfg** (XXXX can be server or common) to select the tools you want to run.

To select a tool uncomment the section with that tool name and to deselect the tool comment the section.

```
#nbench is not selected
#[nbench]
#build = nbench_build.sh
#run = nbench_run.cfg
#parser = nbench_parser.py

#lmbench is selected for execution
[lmbench]
build = lmbench_build.sh
run = lmbench_run.cfg
parser = lmbench_parser.py
```

Figure 4: common_cases_def.cfg snapshot

Note: To generate HTML report, hardware_info tool should be executed.

4.3 Test Case Configuration

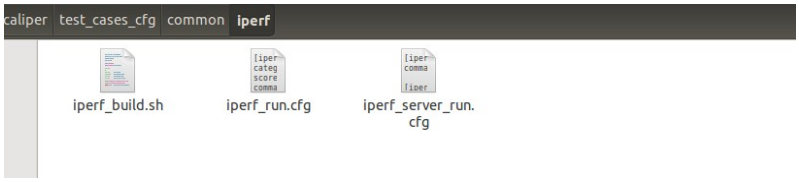


Figure 5: iperf tool config file list

Configure the .cfg files located in

~/caliper_output/configuration/test_cases_cfg/XXXX/*tool_name*/*tool_name*_run.cfg

(XXXX can be server, application and common) to select the test-cases of the tool you want to run. Comment the entire section which you do not want to execute.

```
[lmbench lat]
category = Performance
scores_way =  exp_score_compute 1 -0.5
command = "cd lmbench; ./test.sh latency "
parser = lmbench_lat_parser

#[lmbench bandwidth]
#category = Performance
#scores_way = compute_speed_score 3
#command = "cd lmbench; ./test.sh bandwidth "
#parser = lmbench_bandwidth_parser
```

Figure 6: lmbench_run.cfg file

For some benchmark tools, you need to configure the "**command**" to execute the tool properly. These benchmarks will be listed below.

Sysbench: You need to modify the sysbench_run.cfg in test_cases_cfg/server/sysbench directory to configure your custom information, such as your username and password for your mysql database.

./sysbench.sh root root can be changed to ***./sysbench.sh username password***.

Notes:

1. For fio and iozone tool, individual disk to be mount on /mnt/sdb.
2. To execute hadoop tool, minimum 120 GB of disk should be used. Update the **command** field in ~/caliper_output/configuration/test_cases_cfg/common/hadoop/hadoop_run.cfg file for mount point with relevant disk name.

   Example: command = "if [-f hadoop_tar.gz]; then tar -xvf hadoop_tar.gz; rm hadoop_tar.gz; fi; pushd hadoop; ./hadoop.sh /dev/sdb /mnt/hadoop; popd"
   Here sdb disk is used for hadoop tool execution.

3. User has to format the disk partitions which will be used for fio, Iozone and hadoop tool to ext4 file system. Use ***mkfs.ext4 /dev/<partition name>*** command to format the disk partition.
4. To execute hadoop tool on centos ***platform, /***etc/sudoers file needs to be updated with thi***s command: defaults !requiretty***
5. If caliper is executing for x86_64 target, then make following configuration:
   1. Follow this steps for centos target (To execute compile tool test cases):
      Copy arm64 linaro toolchain from host PC to x86_64 centos target. Export the toolchain path and library path in ~/.bashrc file at the end of file.
      export PATH=gcc-linaro-aarch64-linux-gnu-4.9-2014.09_linux/aarch64linux-gnu/lib:$PATH
      export PATH=gcc-linaro-aarch64-linux-gnu-4.9-2014.09_linux/bin:$PATH

      Note: For Ubuntu target, this step is not required. The toolchain package will be installed by dependency script.

   2. Add this into ~/.bashrc file at the end of file.
      export PATH=/usr/lib64:$PATH (for centos platform)
      export PATH=/lib/x86_64-linux-gnu:$PATH (for ubuntu platform)

export PATH=/usr/lib/x86_64-linux-gnu:$PATH (for ubuntu platform)

6. For **Ubuntu** target, user has to create the soft link for libmysqlclient library to libmysqlclient_r.so.

Example: ln -s <libmysqlclient.so library name> libmysqlclient_r.so
"libmysqlclient.so library name" can be found using commnad: *find / -name libmysqlclient.so**
This may give a list of files with acutal dynamic library and softlinks for the same.

```
root@root1-RH2288H-V3:~/target_dependency_dir# find / -name libmysqlclient.so*
find: '/run/user/1000/gvfs': Permission denied
/usr/lib/x86_64-linux-gnu/libmysqlclient.so
/usr/lib/x86_64-linux-gnu/libmysqlclient.so.20.3.5
/usr/lib/x86_64-linux-gnu/libmysqlclient.so.20
```

Please use 'file' command to find the actual library from the list.

Then create the softlink as given below

ln -s /usr/lib/x86_64-linux-gnu/libmysqlclient.so.20.3.5 /usr/lib/x86_64-linux-gnu/libmysqlclient_r.so.

Please note that the library name would be different in different platforms.
This step is required for sysbench tool execution.

7. For x86_64 target, user has to create softlink for libpcre library to libpcre.so.1.

Example: ln -s <libpcre.so library name> libpcre.so.1
Above procedure can be used.

5 Caliper Execution

For executing caliper you just need to trigger:

caliper –option

Example: ***caliper -brps***

The table describes the options available in caliper. To check if a particular option has executed properly, one can refer to the output column of that option. If the actual output is same as the expected one then the execution is proper else check if the Pre-requisites have been fulfilled or not.

By default –b, -r, -p and -s options are enabled

For Centos target, start mysql server with following command:

- ***cd /usr/local/mysql***
- ***./bin/mysqld_safe &***

5.1 Caliper option description for benchmarking tool execution

Option	Description	Pre-Requisites	Output
-h/ --help	To display the help messages.	----	----

-b/--build	To Build the selected tools incrementally i.e if any tool is already built then skips that tool and uses the already existing binary. The binaries created are then copied to the target.	~/.caliper/benchmarks folder, config folder and the test_cases_cfg folder should be present in caliper_output/workspace/configuration/ directory. Which are the tools need to be build should be selected as described in section 4.2 and 4.3. Auto ssh connection between the target, TestNode and the host should be established as described in section 3.3.1.	• **tool_build.suc** or **tool_build.fail** files in caliper_output/workspace/output/caliper_build/ . • binaries of the tools generated in caliper_output/workspace/binary/target_arch/ .
-B/--no build	To be selected if the tools are not required to be build.	**----**	**----**
-r/--run	To run the binaries of the selected tools that has **tool_build.suc** file in the caliper_build folder.	All required binaries to be build. Check for tool_build.suc file in the caliper_build folder to see the status of the build binaries.	This will generate **tool_output.log** in caliper_output/workspace/output/caliper_exec directory.
-R/--no run	To be selected when execution of the tools is not required.	**----**	**----**
-p/--parse	To parse the output of the selected tools that has **tool_output.log** file in the caliper_exec folder.	**tool_output.log** file in the caliper_exec folder should be present.	This Will generate, • **tool_parser.log** in caliper_output/workspace/output/caliper_exec directory and **final_parsing_logs.yaml** in the caliper_output/workspace/output/ directory. • **<platform>.yaml** and <**platform>_hw_info.yaml** in caliper_output/workspace/output/results/yaml directory.
-P/--no parse	To be selected when parsing the output of the tools is not required.	**----**	**----**
-s/--score	To generate score for the parsed values of the selected tools.	**final_parsing_logs.yaml** in the caliper_output/workspace/output/output folder and the parsed value for the tool in this file should be present.	The result is the <**platform>_score.yaml** in **caliper_output/workspace/output/results/yaml** folder.
-S/--no score	To be selected when scoring for the tools is not required.	**----**	**----**
-e	To send the summary log via email. Ex: **caliper -brpse** or **caliper –brpsef <workspace_name>**	**emal_config.cfg** file in the **caliper_output/workspace/config/email_config.cfg** to be configured with valid email address if "–f" option has been used.	**results_summary.log** file send to receiver.

		emal_config.cfg file in the caliper_output/configuration/config/email_config.cfg to be configured with valid email address if "–f" option has not been used.	
-f/--folder Workspace_name	This option allows the user to specify the name of the output workspace folder as Workspace_name. (this should be the folder of the already executed workspace. System path is not required. Ex: **caliper –brpsf <folder_name>**	If –f option is enabled then the config files in ~/caliper_output/workspace/config and ~/caliper_output/workspace/test_cases_def.cfg should be configured for client information, server information and for the selection of tools.	The folder with the name as Workspace_name is generated in caliper_output directory.

Table 1: caliper options for tool execution

5.2 Caliper option description for html report generation

Option	Description	Pre-Requisites	Output
-w/--webpage	This option is used to generate the HTML report and the consolidated excels for the executed Platforms. **Caliper execution Command:** caliper –BRPSw	In **caliper_output/frontend/frontend/data_files/Input_Logs/** following files to be present: • **<Platform>_score.yaml** of each platform should be placed in **Input_Report** folder • **<Platform>_hw_info.yaml's** of each platform should be placed in **Input_Hardware** folder • **<Platform>.yaml** of each platform should be placed in Input_Consolidated folder • **<Platform>.yaml** of each platform under each iteration should be placed in 5 iteration folders of **Input_Cov.** (This step is only required if user executes caliper for each platform for multiple times) **Example :** • place the **yaml** file of **iteration 1** under the folder **"1"** of **Input_Cov** • place the **yaml** file of **iteration 2** under the folder **"2"** of **Input_Cov** • Continue the same for other plat-	**Test_results.tar.gz** tar file should be generated in **caliper_output/Workspace_name/results/** where Workspace_name is Platform_workspace_timestamp. Untar the **Test_results.tar.gz** file. It contains the following excels and html file. • **index.html** file will be present. Open this file view HTML report. • **Performance-Tests.xlsx** and **Functional-Tests.xlsx** should be present in ~/caliper_output/workspace/output/test_results/static/TestInfo/Report-Data/ to view individual test cases raw value. • **Platform_configuration.xlsx** should be present in ~/caliper_output/workspace/output/test_results/static/TargetInfo/ to view platform information. • **Performance_cov-Tests.xlsx** and

		forms. Following template excels should be present in **caliper_output/frontend/polls/template/polls/** : • Functional_Template.xlsx • Performance_Template.xlsx • Hw_Template.xlsx • Openssl_Template.xlsx **Note: Platform** indicates the name of the particular platform.	**Functional_cov-Tests.xlsx** files will be generated in ~/caliper_output/frontend/polls/static/TestInfo/Iterations/ to view co-variance value for multiple iterations.
-e	Send **Test_results.tar.gz** file, which contains html web report. Ex: **caliper -BRPSwe**	Emal_config.cfg file in the caliper_output/configuration/config/email_config.cfg to be configured with valid email address.	**Test_results.tar.**gz file will be send to receiver.

Table 2: caliper options for HTML report generation

6 Caliper Output

Caliper tool runs in two phases:

1. Execution of benchmarking tools: ***caliper –brps***
2. Generate html report: ***caliper –BRPSw***

6.1 Caliper execution

After the "**caliper –option"** command has been finished; the results are generated in the **~/caliper_output/<folder_name>.**

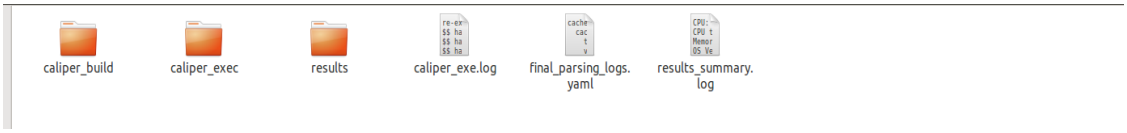
If user executes caliper with "-f <workspace name>" option, then **folder_name** will be same as workspace name.

If user executes caliper without specifying "**-f**" option, then **folder name** will same as **platform_name** provided in ~/caliper_output/configuration/config/client_config.cfg file.

If user executes caliper without specifying "**-f**" option and **platform_name** field in ~/caliper_output/configuration/config/client_config.cfg file is blank, then **folder name** will be host name of the target.

The <**folder name>** directory contains the following directories:

- **binary:** This directory will contain the binary files under the binary/<arch> folder. Where arch is the target architecture Example: x86_64 or arm_64.
- **config:** This directory contains the **client_config.cfg, email_config.cfg, execution_contl.cfg** files, which contains platform login information.
- **test_cases_cfg:** This directory contains the files that will help you to configure the tools to be executed. Refer section 4.2 and 4.3 for configuring the files in this folder.
- **output:** The output directory contains the following directories:



- **caliper_build**: This directory contains the build logs of all the tools built. If the building is successful, then the build logs are named as tools_architecture.suc else it will be named as tools_architecture.fail, for e.g. cachebench_x86_64.suc or cachebench_arm_64.fail.
- **caliper_exec**: This directory contains two files for each tool – Tool_output.log and Tool_parser.log.

  Tool_output.log contains the actual output of the Tool and this output is parsed to get the relevant information and stored in Tool_parser.log
- **results:** The yaml files are stored in results/yaml folder. <Platform>.yaml contains the raw values of test case results of each tool. <Platform>_hw_info.yaml contains the hardware information of the platform. <Platform>_score.yaml contains the values in compressed form.
- **caliper_exe.log:** It captures the start and stop time of each tool and also the total duration for which the tool has executed.
- **results_summary.log:** This file contains the summary of the execution. Information like number of tools selected, built successfully, ran successfully and ran partially is captured in this file.
- **final_parsing_logs.yaml:** This file is the intermediate result of parsing and acts as the input for generating the yamls.

6.2 Caliper report generation

When user executes, caliper –BRPSw to generate web page report, output directory will be generated in ~/caliper_output/<platform_name_WS_current_time>/output/results/test_results.tar.gz.

Extract the file - test_results.tar.gz. test_results directory contains following files:

1. polls: This directory contains html pages for each category.
2. static: This directory contains supported files to generate HTML report.
3. index.html: this is the main page of the caliper report.

Note: After generating HTML report, user has to update the index.html page based on his requirements.

7 FAQ

1. While running caliper with caliper -brps option, if user found following error: "_get_column_letter package not found", then install openpyxl 2.3.0 python package with following command: ***sudo pip install openpyxl==2.3.0***

2. While installing mysql package in Centos target, if user found following error:

"**Can't connect to local MySQL server through socket '/tmp/mysql.sock**"

Then verify that ***/var/lib/mysql/mysql.sock*** file should be present. Link file ***/tmp/mysql.sock*** file should be link to ***/var/lib/mysql/mysql.sock*** file.

3. If user found the error message as "mv: cannot stat '127.0.0.1:8000': No such file or directory" then open the url: **127.0.0.1:8000** in the web browser. The debug message will be shown in the web browser.

8 Known Issues

Issues in dependency package installations through automation scripts.

- When target_dependency.exp is executing, it spawns ssh connection to target platform and install dependency packages on the target.
- If user abruptly stops execution of target_dependency.exp script by mistake, then installation process on the target platform has not been killed automatically.
- So when target_dependency.exp is executed again, the remote process may be holding lock on yum or apt-get package managers. This results in giving the following issue.

```
Loaded plugins: fastestmirror
Existing lock /var/run/yum.pid: another copy is running as pid 1875.
Another app is currently holding the yum lock; waiting for it to exit...
The other application is: yum-builddep
  Memory: 65 M RSS (1.2 GB VSZ)
  Started: Mon Jan 9 15:43:03 2017 - 00:15 ago
  State: Uninterruptible, pid: 1875
```

- Workaround of this issue is as follows:

  On target platform, execute command: ps -ef | grep "/bin/bash ./target_dependency.sh"

  Identify the pid from the above output then kill that process.

9 Appendix

9.1 Host dependency Installation

Host dependency packages are already been installed by automation scripts.

If user wants to install these packages manually, then follow below commands.

```
sudo apt-get update
sudo apt-get install python-pip python-dev build-essential
sudo apt-get install automake autoconf make
sudo apt-get install openssh-server
sudo apt-get install libnuma-dev
sudo apt-get install texinfo
sudo apt-get install nfs-kernel-server
sudo apt-get install libc6
sudo apt-get install libncurses5 unzip
sudo apt-get install libstdc++6
sudo apt-get install lib32z1
sudo apt-get install lib32stdc++6
sudo apt-get install bzr
sudo pip install Django==1.8.4
sudo pip install matplotlib==1.3.1
sudo pip install numpy==1.8.2
sudo pip install openpyxl
sudo apt-get install openjdk-7-jre
sudo apt-get install openjdk-7-jdk

wget http://www.estuarydev.org/caliper/pcre-8.39.tar.gz
tar -zxvf pcre-8.39.tar.gz
cd pcre-8.39
./configure
make -j32
sudo make install
```

9.2 TestNode Dependency Installation

TestNode dependency packages are already been installed by automation scripts.

If user wants to install these packages manually, then follow below commands.

sudo apt-get install netperf
sudo apt-get install iperf3

Start/Restart netperf by running: sudo service netperf restart
Run iperf3 in background by running: iperf3 –s &

wget http://www.estuarydev.org/caliper/redis-3.2.4.tar.gz
tar xvf redis-3.2.4.tar.gz
cd redis-3.2.4/src
make all
mkdir ~/caliper_redis
cp redis-benchmark redis-cli ~/caliper_redis/

9.3 Target Dependency Installation

Target dependency packages are already been installed by automation scripts.

If user wants to install these packages manually, then follow below commands.

9.3.1 Install below packages if target OS is Ubuntu

sudo apt-get update
sudo apt-get install stress
sudo apt-get install build-essential
sudo apt-get install linux-tools-generic
sudo apt-get install linux-tools-common
sudo apt-get install gcc g++
sudo apt-get install nfs-common
sudo apt-get install automake autoconf autogen libtool make
sudo apt-get install openjdk-7-jre
sudo apt-get install openjdk-7-jdk
sudo apt-get install mysql-server libmysqlclient-dev
sudo apt-get install stress-ng
sudo apt-get install expect
sudo apt-get install bzr
sudo apt-get install lshw
sudo apt-get install bridge-utils
sudo apt-get install dmidecode
sudo apt-get install lsdev
sudo apt-get install unzip bc
sudo apt-get install dstat

sudo apt-get install gcc-aarch64-linux-gnu (for x86_64 target)

9.3.2 Install below packages if target OS is CentOS

sudo yum install make
sudo yum install wget
sudo yum install gcc
sudo yum install automake
sudo yum install autoconfc
sudo yum install cmake
sudo yum install net-tools
sudo yum install lshw
sudo yum install bridge-utils
for x86_64 target: sudo yum install java-1.8.0-openjdk.x86_64
for arm64 target: sudo yum install java-1.8.0-openjdk.arm64
for x86_64 target: sudo yum install java-1.8.0-openjdk-devel.x86_64
for arm64 target: sudo yum install java-1.8.0-openjdk-devel.arm64
sudo yum install perl
sudo yum install lksctp-tools
sudo yum install expect

sudo yum install ncurses-devel
sudo yum install yum-utils
sudo yum install dmidecode
sudo yum install libtool unzip bc
sudo yum install dstat psmisc
sudo yum install gcc-c++ libaio

sudo yum install zlib.i686 (for x86_64 target)

stress package installation:
cd /tmp
wget http://www.estuarydev.org/caliper/stress-1.0.4.tar.gz
sudo tar xvzf stress-1.0.4.tar.gz
cd stress-1.0.4
./configure && sudo make && sudo make install

stress-ng package installation:
cd /tmp
wget http://www.estuarydev.org/caliper/stress-ng.zip
sudo unzip stress-ng.zip
cd stress-ng-master
sudo make && sudo make install