PERFSONAR

Lab 5: Configuring Regular Tests Using pScheduler CLI Part II

Document Version: 02-23-2020

Award 1829698
“CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers”
Contents

Overview ........................................................................................................................................... 3
Objectives .......................................................................................................................................... 3
Lab topology ...................................................................................................................................... 3
Lab settings ....................................................................................................................................... 4
Lab roadmap ....................................................................................................................................... 4
1 Introduction ..................................................................................................................................... 5
  1.1 The pScheduler command ........................................................................................................... 6
2 Running tasks from other perfSONAR nodes .................................................................................... 7
3 Repeating tasks ................................................................................................................................. 8
4 Exporting and importing tasks to JSON ........................................................................................... 9
5 Viewing the schedule ......................................................................................................................... 11
  5.1 pScheduler monitor ..................................................................................................................... 11
  5.2 pScheduler schedule .................................................................................................................... 12
6 Canceling tasks ................................................................................................................................. 13
References .......................................................................................................................................... 17
Overview

This lab continues the description of pScheduler commands, and how to use it to run measurement tests between perfSONAR nodes. This lab is focused on running a pScheduler task from other nodes, repeating, exporting and importing tasks. In addition, the tools to visualize the schedule are presented. Finally, the user will learn about the procedure to cancel a task.

Objectives

By the end of this lab, the user will:

1. Understand pScheduler commands.
2. Run tasks from other perfSONAR nodes.
3. Repeat a specific task.
5. Use the visualization tools.
6. Cancel a specific task.

Lab topology

Figure 1 illustrates the topology used for this lab. The topology includes three perfSONAR nodes labeled perfSONAR1, perfSONAR2, perfSONAR3 and a Client host. The perfSONAR nodes run a Linux CentOS 7, and the Client runs a lightweight Linux distribution (Lubuntu). The Client host is used to access perfSONAR graphical user interface.
Lab settings

The information in Table 1 provides the credentials to access to perfSONAR nodes.

Table 1. Credentials to access perfSONAR1, perfSONAR2 and perfSONAR3.

<table>
<thead>
<tr>
<th>Device</th>
<th>IP Address</th>
<th>Account</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>perfSONAR1</td>
<td>192.168.1.10</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>perfSONAR2</td>
<td>192.168.2.10</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>perfSONAR3</td>
<td>192.168.3.10</td>
<td>admin</td>
<td>admin</td>
</tr>
</tbody>
</table>

Lab roadmap

The lab includes the following tasks:

1. Section 1: Introduction.
2. Section 2: Running tasks from other perfSONAR nodes.
3. Section 3: Repeating tasks.
4. Section 4: Exporting and importing tasks.
5. Section 5: Viewing the schedule.

1 Introduction

pScheduler is responsible for managing the execution of network measurements, or more generally tasks, in perfSONAR. When the user wants to run a network measurement on perfSONAR, it is performed through pScheduler command-line. pScheduler is part of the scheduling layer, as it is shown in the figure 2. The scheduling layer is responsible for:

- Finding timeslots to run the tools while avoiding scheduling conflicts that would negatively impact results.
- Executing the tools and gathering results.
- Sending to the results to the archiving layer (if needed).

pScheduler handles the coordination, execution and optionally storage of the task requested. Many of the tools pScheduler executes could be run independently of pScheduler. However, pScheduler provides additional features that the tools by itself does not provide. These features are listed below:

- **Measurement Integrity**: pScheduler maintains a schedule of all measurements to be run and will not allow any measurements to run simultaneously, if doing so, it would adversely affect the result in a significant way. For instance, it will not run two throughput tests at the same time as the competition for resources could
affect the results of each. In contrast, it will run latency tests in the background as the low resource consumption does not significantly affect results of parallel tests.

- **Simplified Coordination:** In addition to simplify coordination during task execution, pScheduler will contact each end device and handle bringing up any daemons as required. It also has a plug-in architecture that allows to send the result elsewhere, such as a long-term storage system, as well the measurement completes.

- **Access Control:** pScheduler has a limits system that allows the definition of rules about who can run what type of measurements and other rules as how long a test can run, and which tests can run in a specific node.

- **Diagnostics:** pScheduler provides the tools to visualize the schedule. It specifies when a task ran, runs or will run. Additionally, it keeps for some amount of time information about the outcome, including whether the result was a failure or not, which can be useful for diagnosing issues with in a network.

In addition to these foundational features, pScheduler allows plug-ins for new tests, tools and archivers to be written. This means that pScheduler allows extensions to perform new type of measurements or other functions as well as to have their results sent to new types of storage and/or analysis tools.

### 1.1 The pScheduler command

The user interacts with perfSONAR using **pscheduler** command. The pScheduler command is the primary way from the command-line to create new pScheduler tasks. The basic syntax is as follows:

```
pscheduler command [args] (1)
```

- **pscheduler:** command used to interact with perfSONAR.
- **command:** it describes the type of test that will be performed, these commands could be task commands or administrative and diagnosis commands, each command has its lists of arguments **args**. The task commands are listed as follows:

  - **task:** give pScheduler a task that consists of making one or more measurements.
  - **result:** fetch and display the results of a single, previously-concluded run by its URL.
  - **watch:** attach to a task identified by URL and show run results as they become available.
  - **cancel:** stop any future runs of a task.

The following commands are for diagnosis and administrative:

  - **ping:** determine if pScheduler is running on a host.
  - **clock:** checks and compare the clocks on pScheduler hosts.
  - **debug:** Enable debugging on the internal part of pScheduler.
• **diags**: Produce a diagnostic dump for the perfSONAR team to use in resolving problems.

For more information about pScheduler tasks, diagnosis and administrative commands, the user get access to the help typing on the perfSONAR command-line:

```
pscheduler --help
```

To get more details about a specific command, using the format of the command (1) type:

```
pscheduler [command] --help
```

## 2 Running tasks from other perfSONAR nodes

pScheduler determines where to submits a task based on the test parameters. Where a task needs to be submitted is called the lead participant. For many tests run by perfSONAR, a **--source** switch which specifies where the test should originate and is also the lead participant. In this section the user will run a throughput test using pScheduler commands. This test will be submitted by perfSONAR1 node (192.168.1.10), however, the tests will run between perfSONAR2 (192.168.2.10) and perfSONAR3 (192.168.3.10) nodes.

**Step 1.** On the topology, click on perfSONAR1 then, enter the username **admin** and password **admin**. Note that the password will not be displayed while typing it. Proceed similarly with perfSONAR2 and perfSONAR3 nodes.

```
CentOS Linux 7 (Core)
Kernel 3.10.0-517.1.3.el7.x86_64 on an x86_64

perfsonar1 login: admin
Password: ...
Welcome to the perfSONAR Toolkit v4.1.5-1.el7
You may create accounts to manage this host through the web interface by running the following as root:

```
/usr/lib/perfsonar/scripts/uimtkit-configure.py
```

The web interface should be available at:

```
https://[host address]/toolkit
[admin@perfsonar1 ~]$ ...
```

**Step 2.** In perfSONAR1 command line, follow command format (1) and type:

```
pscheduler task throughput --source 192.168.2.10 --dest 192.168.3.10
```

- **pscheduler**: is the command to interact with perfSONAR.
- **task**: is a pScheduler command to specify a measurement test.
- **throughput**: specifies the test.
- **--source**: is to specify where the test should originate, in this case it is perfSONAR1 node (192.168.1.10)
• `-dest` is the destination node, in this case is the perfSONAR3 node (192.168.3.10).

Shortly after starting the test submission, the user will see that the tool used to run the test is iperf3. The results above, lists the throughput every second (Interval), the number of retransmissions (Retransmits) and current windows size. At the end, it is summarized the time interval when the test took place, in this case form 0 seconds to 10 seconds, the throughput is 6.66 Gbps and the number of retransmissions is 1535.

In this example, the command above is run on perfSONAR1 node (192.168.1.10), then the node will submit the task to perfSONAR2 node (192.168.2.10) and the test will be run between perfSONAR2 (192.168.2.10) and perfSONAR3 (192.168.3.10).

3 Repeating tasks

A task can be configured to run periodically. In this section, it is shown step by step how to repeat throughput and RTT tasks using pScheduler command. First the user will configure pScheduler to run a throughput task every 30 seconds. Then, the user will run an RTT task every 45 seconds. Any pScheduler task can be configured to run repeatedly by adding options to the task command:

- `-start TIMESTAMP`: it runs the first iteration of the task at TIMESTAMP.
- `-repeat DURATION`: Repeat runs at intervals of DURATION.
- `-max-runs N`: Allow the task to run up to N times.
- `-until TIMESTAMP`: Repeat runs of the task until TIMESTAMP.
Lab 5: Configuring Regular Tests Using pScheduler CLI Part 2

- **--slip DURATION**: Allow the start of each run to be as much as DURATION later than their ideal scheduled time. If the environment variable PSCHEDULER_SLIP is present, its value will be used as a default. Failing that, the default will be PT5M. Notice that the slip value also applies to non-repeating tasks.

- **--sliprand**: Randomly choose a timeslot within the allowed slip instead of choosing earliest available.

**Step 1.** In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler task --repeat PT20M --max-runs 10 rtt --dest 192.168.2.10
```

- **pscheduler**: is the command to interact with perfSONAR.
- **--repeat PT20M**: is a pScheduler command that configure the task to be repeated every 20 minutes.
- **task**: is a pScheduler command to specify a measurement test.
- **throughput**: is the test type.
- **--dest** is the destination node, in this case it is perfSONAR2 node (192.168.2.10). Notice that the source node is not explicit, this means that the source node is perfSONAR1 (192.168.1.10).

The figure above shows the first measurement of the round-trip time. Notice that the task is going to be repeated 10 times in 20 minutes.

**Step 2.** To return to the CLI, press **Ctrl+C**. Notice that the task will keep running.

4 **Exporting and importing tasks to JSON**
The user can export a pScheduler task to a Java Script Object Notation (JSON) file. The JSON version of a task specification can be sent to the standard output without scheduling using the `--export` command.

**Step 1.** In *perfSONAR1* command line, follow the command format (1) and type:

```bash
pscheduler task --repeat PT3M --export throughput --source 192.168.1.10 --dest 192.168.2.10 > my_task_1
```

- `pscheduler`: is the command to interact with perfSONAR.
- `task`: is a pScheduler command to specify a measurement test.
- `--repeat PT3M`: is a pScheduler command that configure the task to be repeated every 3 minutes.
- `--export`: is to indicate that the task will not be executed but stored.
- `throughput`: is the test type.
- `--source`: is to specify where the test should originate, in this case it is perfSONAR1 node (192.168.1.10).
- `--dest` is the destination node, in this case it is perfSONAR2 node (192.168.2.10).
- `> my_task_1`: is to create a file where the task is going to be stored.

**Step 2.** In order to visualize the file, type `cat my_task_1`. A JSON file will be displayed. This file contents a pScheduler task, however this task is not running. Notice also that the task might be invalid because tasks are not validated until they are submitted for scheduling.

**Step 3.** A JSON file that was previously exported or generated elsewhere can be imported using the `--import` command. In *perfSONAR1* command line, follow the command format (1) and type:

```bash
pscheduler task --import my_task_1
```

- `pscheduler`: is the command to interact with perfSONAR.
- `task`: is a pScheduler command to specify a measurement test.
• **my_test_1** is the file that contains the task.

Step 4. To return to the CLI, press `Ctrl+C`. Notice that the task will keep running.

5 Viewing the schedule

In this section, it is presented two visualization tools, pScheduler monitor and pScheduler schedule. The tests scheduled in the last section still running. The user will use pScheduler commands to visualize the schedule.

5.1 pScheduler monitor

The `pscheduler monitor` command provides top-like output of what the schedule is doing in near real time. It takes the following form:

**Step 1.** In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler monitor
```
The user will see the scheduled tests. These tests have a status depending on whether they have already run or are still waiting to do so. Possible status values are:

- **Pending**: This run is scheduled to execute at some point in the future.
- **On Deck**: This run is scheduled to execute and will begin execution very soon.
- **Running**: This run is in the middle of execution.
- **Cleanup**: This run completed execution and is doing some final operations.
- **Finished**: The run has already executed and finished successfully.
- **Overdue**: The run was scheduled to execute at a certain time in the past but did not. It may get executed soon if it is not beyond a certain threshold.
- **Missed**: The run was scheduled but did not execute at its given time. This can happen if the scheduler was not running at the allotted time or the task was paused.
- **Failed**: The run failed to complete for some reason.
- **Non-Starter**: The run could not be scheduled because there were no timeslots that could accommodate the constraints.
- **Canceled**: The task was cancelled before the run was executed.

**Step 2.** To exit from pScheduler monitor, press `Ctrl+C`.

### 5.2 pScheduler schedule
The `pscheduler schedule` command asks pScheduler to fetch scheduled task runs from the past, present or future and display them as text.

**Step 1.** In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler schedule
```

**Step 2.** To exit from pScheduler schedule, press `Ctrl+C`.

### 6 Canceling tasks

So far there are two pscheduler tasks running. In this section, the user will cancel the scheduled Round-Trip Time (RTT) and throughput tasks which are running.

**Step 1.** In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler schedule --filter-test rtt
```
Step 2. In perfSONAR1 command line, follow the command format (1) and type:

\texttt{pscheduler cancel https://localhost/pscheduler/tasks/[url]}

Replace \texttt{[url]} with the first three characters of the last task URL. In this example, the first three characters of the task is \texttt{631}, these characters may vary then, press \texttt{Tab} key to autocomplete the following characters. \texttt{Press Enter} to cancel the task.

Step 3. In perfSONAR1 command line, type the command \texttt{pscheduler monitor} to visualize the schedule.
The user will notice that all the Round-Trip Time (RTT) tasks are finished and there are not more tasks like this scheduled.

**Step 4.** In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler schedule --filter-test throughput
```

The user will see the scheduled task for round-trip time (RTT) tests.
Step 5. In perfSONAR1 command line, follow the command format (1) and type:

```
pscheduler cancel https://localhost/p scheduler/tasks/[url]
```

Replace `[url]` with the first three characters of the last task URL. In this example, the first three characters of the task is `f44`, these characters may vary then, press Tab key to autocomplete the following characters. Press Enter to cancel the task.

Step 6. In perfSONAR1 command line, type the command `pscheduler monitor` to visualize the schedule.
The user will notice that all the tasks are finished and there are not more tasks scheduled.

This concludes lab 5.

References