Usage of the testbed scripts

Listing experiments

Use the `list-experiments.py` script to list current experiments. The script will give you an overview of all the experiment from 1 hour before the current time until 12 hours into the future.

Invoke the script with `./list-experiments.py` if it is located in your current working directory (otherwise, adjust the path accordingly).

The output will look similar to this:

```
20 Nodes 10 Firmware Threads 10 Log Retrieval Threads (ID 25)
[SCHEDULED]:    2022-09-06 14:49:00     ->  2022-09-06 14:51:00

20 Nodes 10 Firmware Threads 10 Log Retrieval Threads (ID 24)   [DONE]:
2022-09-06 13:46:00     ->      2022-09-06 13:47:00
```

Install the `tabulate` package with `pip install tabulate` to get a prettified output.

Listing your own experiments

To list *all of* your own experiments, invoke `./own-experiments.py`. The output will be formatted the same way as from `./list-experiments.py`

Examples

You will find a couple of general examples you can use for your configurations in the examples folder

Queueing an experiment

This is meant to execute an experiment as soon as possible for a given duration.

For this, first create an experiment descriptor file. This file contains information about the name, duration and what nodes are involved. Refer to `experiment-queued.yaml` to get an example for this.

You can specify the duration several ways:
`duration: 600` will run for 600 seconds
`duration: 10:00` will run for 10 minutes
`duration: 12:00:00` will run for 12 hours
`duration: 1:00:00:00` will run for one day.

Once you have done this, make sure that all firmware files you specified in the configuration are available in one folder. The script will search for the exact names used in the configuration.

Invoke the script by using `./queue-experiment.py <experiment-descriptor> <firmware-directory>`.

For example: `./queue-experiment.py experiment-queued.yaml ./firmware`.

You will receive output if the experiment was successfully scheduled and receive a unique ID as well as the scheduled time for the experiment. This ID is important to retrieve the experiment's results later.

Scheduling an experiment

This is meant to execute experiments at a given time.

For this, first create an experiment descriptor file. This file contains information about the name, start and end of the experiment and most importantly what nodes are involved. If you are unsure if the time slot you want to schedule your experiment in is already taken, refer to **Listing experiments**.

Refer to the `experiment-scheduled.yaml` file to get a template for this.

Once you have done this, make sure that all firmware files you specified in the configuration are available in one folder. The script will search for the exact names used in the configuration.

Invoke the script by using `./schedule-experiment.py <experiment-descriptor> <firmware-directory>`.

For example: `./schedule-experiment.py experiment-scheduled.yaml ./firmware`.

You will receive output if the experiment was successfully scheduled and receive a unique ID for the experiment. This ID is important to retrieve the experiment's results later.

Stopping an experiment

Use `./stop-experiment.py <experiment-id>` to stop an experiment that has not finished, yet. If the experiment has already started, log files will still be transferred and you will be able to retrieve them. If the experiment has not yet started, the status will immediately switch to `Cancelled` and you will not be able to retrieve logs.

Retrieving results from an experiment

Results may only be retrieved from experiments that have finished. Use `./download-results.py <experiment-id> [target-directory]` to retrieve an experiment's results. The experiment ID is as returned from the creation of an experiment (otherwise refer to **Listing experiments**). The target directory is optional and, if not provided, will create a folder named `results` in the current working directory and place the results there.

Showing the status of the testbed

Invoke `./status.py` to show the status of all connected nodes of the testbed.