Service Traffic Mix Scenario

This scenario allows to create several traffics, to schedule and launch them with OpenBACH. The four traffics available are:

Traffic	Description	Jobs used
voip	VoIP traffic	voip_qoe_src, voip_qoe_dest
dash	DASH video transfer	dash player&server, dash client
web_browsing	Web browsing	apache2, web_browsing_qoe
data_transfer	File transfer	iperf3

The description of each traffic to be launched has to be put in an extra_args file, which will be described in the next section. This scenario can be found here.

You can launch the service scenario with the following command:

```
python3 generate_service_traffic.py -o -p your_project --extra_args_traffic
path_to_traffic_args_file --entity_pp your_entity run
```

extra_args file description

This file contains all the information needed to generate the traffics. It is composed of several lines, one per traffic. Each line can be split into 2 parts: first the arguments common to all the traffics, then the arguments linked to the traffic declared in this line. An example can be found here.

The first part must be formatted as follows:

```
id traffic_type src_entity dst_entity duration wait_launched wait_finished wait_delay src_ip
dst_ip
```

Here is a description of each element:

- **id**: the id of the flow, must be unique, and should be higher than the id of previous lines
- **traffic_type**: the kind of traffic: voip, dash, web_browsing or data_transfer
- **src_entity**: the name of the source entity
- **dst_entity**: the name of the destination entity
- **duration**: the duration of the transfer for this traffic in seconds (not always used by data transfer: see below)
- **wait_delay**: time to wait in seconds between the completion of *wait_launched* and *wait_finished*, and the launch of this traffic
- **wait_launched**: launch the traffic *wait_delay* seconds after these traffics are launched. These traffics are described by their ids (which has to be declared in the previous lines), separated by a "-". If no traffic, put "None"
- **wait_finished**: launch the traffic *wait_delay* seconds after these traffics are finished. These traffics are described by their ids (which has to be declared in the previous lines), separated by a "-". If no traffic, put "None"
- **src_ip**: the ip address of the traffic source (ignored by web browsing and data transfer traffics)
- **dst_ip**: the ip address of the traffic source (ignored by DASH and web browsing traffics)

For example, if we enter the following line, assuming traffics with id 1, 2 and 3 have already been declared in the file:

```
4 dash src_node dst_node 60 3 1-2 10 192.168.1.1 192.168.1.2
```

This line says: launch a DASH traffic (with id=4) between src_node (whose address is 192.168.1.1) and dst_node (whose address is 192.168.1.2), during 60 seconds. This traffic will be launched 10 seconds after 1 and 2 are finished and 3 is launched.

The second part of the line depends on the traffic chosen, and is described in the following table.

Traffic	Arguments format	Warnings and additional information
voip	dst_port codecs	Actually it is impossible to launch simultaneously to VoIP traffics. Codecs available are G.711.1, G.711.2, G.723.1, G.729.2 and G.729.3
dash	protocol	Protocol value is http/1.1 or http/2
web_browsing	nb_runs nb_parallel_runs	Do not forget to edit the config file with correct address to fetch in the installation folder of the job (/opt/openbach/agent/jobs/web_browsing_qoe)
data_transfer	dst_port file_size tos mtu	If *file_size* is 0, the transfer will last *duration* seconds. Otherwise, it will transfer *file_size* bytes, regardless of the duration

Any traffic not correctly formatted will be ignored.

Statistics returned

The scenario returns the metrics generated by the exploited jobs as a raw csv file. time_series and histograms plots are generated:

- **voip**: one time_series and one histogram per VoIP traffic, and one time_series and one histogram with all VoIP traffics. The metric plotted is the Mean Opinion Score
- **dash**: one time_series and one histogram with all DASH traffics. The metric plotted is the bitrate
- **web_browsing**: one time_series and one histogram per web_browsing traffic, and one time_series and one histogram with all web_browsing traffics. The metric plotted is the Page Load Time
- **data_transfer**: one time_series and one histogram per data_transfer traffic, and one time_series and one histogram with all data_transfer traffics. The metric plotted is the throughput