lmapd implementation

Vlad Ungureanu
vvu@vdev.ro

May 11, 2015

lmapd

- lmapd is the proof-of-concept execution engine implementation for Large-Scale Measurement of Broadband Performance (LMAP)
- Available at ungureanuvladvictor.github.io/lmapd
- Licensed under GPLv3

Features I

- Event driven execution loop
- Configuration file passed in XML (taken from YANG data-model)
- Execution of actions inside a schedule in the following ways:
  - parallel (all actions are started at the same time)
  - sequential (each action is started after the previous one finished)
  - chained (the output of one action is piped to the next one)
- Reload of the configuration file while running

Features II

- Build system based on `autotools`[1]
- Small memory footprint - 0.46Mb (dynamically compiled)
- Small dependency list:
  - `libxml2`[2] - library for XML parsing
  - `libevent`[3] - library for asynchronous event execution

[1] `http://en.wikipedia.org/wiki/GNU_build_system`
[2] `http://www.xmlsoft.org/`
[3] `http://libevent.org/`

Source Code

- Parser:
  - Parse the XML file
  - Each section of config file mapped to a C struct
  - Easy to extend for new sections of config file
- Runner:
  - 2 event loops that run asynchronous and tick every 1 second
  - Runner callback checks which schedule to execute and forks from main process
  - Cleanup callback moves files from action to destination
- Utils:
  - Replaces SIGUSR1 and SIGHUP handlers
  - SIGHUP - reloads config file and starts loops again
  - SIGUSR1 - outputs statistics about each action in a .csv file for inspection

Future Development

- A testing framework especially for the parser module
- Implementation of:
  - Timing objects: ma-startup-obj, ma-one-off-obj, ma-calendar-obj
  - Suppression: ma-suppression-obj
  - Reporting: ma-report-obj
  - Logging: ma-log-obj
- Better documentation(man page, source-code comments, how-to-use instructions)

Conclusion

- No description on how to send output from one action to another
- The need of a queue/workspace dir is needed from an implementation point of view
- Periodic timing interval specified in milliseconds (too fine grain)
- No clear way to deal with tasks that do not finish