MptcpAnalyzer Documentation

Release 0.2

Matthieu Coudron

September 13, 2016

This is the manual of mptcpanalyzer, a python based linux tool to help plot some characteristics of an multipath TCP connection based on network traces (*.pcap files).

This document is written in reStructuredText for Sphinx and is maintained in the `docs/` directory of the package source code.

mptcpanalyzer uses semantic versioning .

You can reference mptcpanalyzer via the following Digital Object Identifier: [![DOI](https://zenodo.org/badge/21021/lip6-mptcp/mptcpanalyzer.svg)](https://zenodo.org/badge/latestdoi/21021/lip6-mptcp/mptcpanalyzer)

Contents:

CHAPTER 1

Introduction

1.1 Features

- list the MPTCP connections in the pcap
- display some statistics on a specific MPTCP connection (list of subflows etc...)

It accepts as input a capture file (*.pcap) and depending on from there can : * pcap to csv conversion * plot data sequence numbers for all subflows * XDG compliance, i.e.,

> **mptcpanalyzer** looks for files in certain directories. will try to load your configuration from *$XDG_CONFIG_HOME/mptcpanalyzer/config*

- caching mechanism: mptcpanalyzer compares your pcap creation time and will regenerate the cache if it exists in *$XDG_CACHE_HOME/mptcpanalyzer/<path_to_the_file>*
- support 3rd party plugins (plots or commands)

Most commands are self documented and/or with autocompletion.

Then you have an interpreter with autocompletion that can generate & display plots such as the following:

![Data Sequence Number (DSN) per subflow plot](examples/dsn.png)

1.2 How to install ?

First of all you will need a wireshark version that supports MPTCP dissection, i.e., wireshark > 2.1.0. If you are on ubuntu, there are dev builds on https://launchpad.net/~dreibh/+archive/ubuntu/ppa/.

Once wireshark is installed you can install mptcpanalyzer via pip:

command:*$ python3.5 -mpip install mptcpanalyzer –user*

python3.5+ is mandatory since we rely on its type hinting features. Dependancies are (some will be made optional in the future):

- stevedore to handle the plugins architecture
- the data analysis library pandas >= 0.17.1
- matplotlib to plot graphs
- (lnumexpr to run specific queries in pandas)

1.3 How does it work (internals) ?

mptcpanalyzer consists of small python scripts. the heavy task is done by wireshark. It relies on tshark (terminal version of wireshark) to convert pcap to csv files.

It accepts as input a pcap (or csv file following a proper format). Upon pcap detection, mptcpanalyzer the formats supported by tshark (terminal version of wireshark).

Usage

This package installs 2 programs: - *mptcpanalyzer* to get details on a loaded pcap.

mptcpanalyzer can run into 3 modes:

1. *Interactive mode* (default): an interpreter with some basic completion will accept your commands.
2. *Batch mode* if a filename is passed as argument, it will load commands from this file.
3. *One-shot mode*, it will consider the unknow arguments as one command, the same that could be used interactively

For example, we can load an mptcp pcap (I made one available on wireshark wiki or in this repository, in the _examples_ folder).

It expects a trace to work with. If the trace has the form *XXX.pcap* extension, the script will look for its csv counterpart *XXX.pcap.csv*. The program will tell you what arguments are needed. Then you can open the generated graphs.

2.1 Interactive mode

Run *$ mptcpanalyzer –load examples/iperf-mptcp-0-0.pcap*. The script will try to generate a csv file, it can take a few minutes depending on your computer. Then you have a command line: you can type **?** to list available commands. You have for instance:

- *lc* (list connections)
- *ls* (list subflows)
- *plot*
- ...

help ls will return the syntax of the command, i.e. *ls [mptcp.stream]* where mptcp.stream is one of the number appearing in *lc* output.

Some more complex commands can be:

```
load --regen examples/iperf-mptcp-0-0.pcap
plot attr 0 Client dsn
plot attr 0 Client dsn  --title "custom title" --out test_with_title.png
plot attr 0 Client dsn  --skip 1 --skip 3 --style examples/red_o.mplstyle --title "Test with matplotl
```

2.2 Batch mode

Commands are the same as in *Interactive mode*, they are just saved in a file.

```
mptcpanalyzer --batch tests/batch_commands.txt -dddd
```

2.3 One-shot mode

Just put your command after your arguments, for instance.

```
mptcpanalyzer --load examples/iperf-mptcp-0-0.pcap`
```

2.4 Tips

mptcpanalyzer is a rather long name so feel free to create an alias for instance **`alias mp="mptcpanalyzer"`**.

To enable debug informations, run **`mptcpanalyzer -dddd`**

If you use zsh, you can enable autocompletion via adding to your .zshrc:

```
compdef _gnu_generic mptcpanalyzer
```

2.5 Conversion from pcap to csv:

mptcpanalyzer comes bundled with an extra program: *mptcpexporter* can convert a pcap to csv (exporting to sql should be easy). Run **`mptcpexporter -h`** to see how it works.

2.6 List of available plots

CHAPTER 3

Configuration

mptcpanalyzer accepts few parameters that can be recorded in a configuration file. The file can be specified on the command line via the '–config' (or '-c') switch:

Listing 3.1: Editing config

```
$ mptcpanalyzer --config myconfig.cfg
```

By default, mptcpanalyzer will try to load the config file in the following order:

1. $XDG_CACHE_HOME/mptcpanalyzer/config, then in
2. $HOME/.config/mptcpanalyzer/config

```
# this is an exhaustive example of a configuration for mptcpanalyzer
# mptcpanalyzer loads automatically the file in "$XDG_CONFIG_HOME/mptcpanalyzer/config"

[DEFAULT]
# If you have several installations of tshark, then you can put the fullpath here
tshark_binary = tshark

# by default, look into  ${XDG_CACHE_HOME:.config}/mptcpanalyzer/config
cache = /tmp

# in case you want to export special options
wireshark_profile = default

# TODO add example
# style0 =
# style0 =
# style2 =
# style3 =
```

- *delimiter* is the csv separator used by tshark when exporting the pcap
- *styleX* follow matplotlib conventions to set lines color/style
- *tshark_bin* in case you want to run a specific tshark binary

CHAPTER 4

How to contribute to mptcpanalyzer ?

There are several things you can do:

- submit bug reports in our tracker
- *Develop an mptcpanalyzer plugin*, if you do, please warn us so that we can add you to the list of plugins
- Send patches to either fix a bug, improve the documentation or flake8 compliance

4.1 Develop an mptcpanalyzer plugin

mptcpanalyzer can load plugins following stevedore's plugin, i.e. mptcpanalyzer will look for specific disttools entry points in order to find and load plugins.

To add a plugin, just mimic what is done for existing plugins, see stevedore's plugin documentation plus check our setup.py:

```
            'mptcpanalyzer.plots': [
                'attr = mptcpanalyzer.plots.dsn:PerSubflowTimeVsAttribute',
                'interarrival = mptcpanalyzer.plots.dsn:InterArrivalTimes',
                'xinterarrival = mptcpanalyzer.plots.dsn:CrossSubflowInterArrival',
                'dss_len = mptcpanalyzer.plots.dsn:DssLengthHistogram',
                'dss = mptcpanalyzer.plots.dsn:DSSOverTime',
                'owd = mptcpanalyzer.plots.owd:OneWayDelay',
                'ns3 = mptcpanalyzer.plots.ns3:PlotTraceSources',
                ],
            # namespace for plugins that monkey patch the main Cmd class
            'mptcpanalyzer.cmds': [
                'stats = mptcpanalyzer.command_example:CommandExample',
              ]
```

mptcpanalyzer will load all plugins residing in these two namespaces:

- **mptcpanalyzer.plots**
- **mptcpanalyzer.cmds**

regardless of which python package they belong

In order to test while modifying mptcpanalyzer, you can install it like this:

```
$ python3.5 setup.py develop --user
```

Note: Add –uninstall to remove the installation.

4.1.1 Develop a new plot

You must create a new class that inherits from `mptcpanalyzer.plot.Plot` (or one of its children). Then you most likely need to override.

4.1.2 Develop a command plugin

Just follow the example in:

```python
from .command import Command

import logging


"""
While in mptcpanalyzer sources, one can do getLogger(__name__) to retrieve a
(sub)logger, your plugin can be in another package and as such, you have to name
the logger explicitly with mptcpanalyzer.**
"""
log = logging.getLogger("mptcpanalyzer")

class CommandExample(Command):
    """
    This is just an example of how to write a plugin that will be automatically
    loaded by mptcpanalyzer.

    """
    def do(self, data):
        """
        :param data: This is the line passed by the user to the interpreter
        """
        print("You wrote: %s" % data)

    def help(self):
        """
        Message printed when the author writes
        """
        print("Prints 'Hello world !' followed by the user message")

    def complete(self, text, line, begidx, endidx):
        """
        To provide autocompletion
        """
        pass
```

4.1.3 How to upload it to pypy (for the forgetful maintainer)

```
$ python3.5 setup.py sdist upload
```

(test first the package locally pip install /path/toarchive.gz)

CHAPTER 5

API

FAQ

1. What if I have several versions of wireshark installed ? Copy the config.example in the repository in *$XDG_CONFIG_HOME/mptcpanalyzer/config* and set the *tshark_binary* value to the full path towards the tshark version that supports mptcp dissection.

2. tshark complains about a corrupted pcap For instance *tshark: The file "/home/user/file.pcap" appears to have been cut short in the middle of a packet.* Analyze your pcap with https://f00l.de/pcapfix/.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search