1 Overview

This article provides some information about *Cli* alongside with examples on how to use it.

2 Introduction

Cli giveus us acces to *pipes* which can manipulate and work with S2 format. Pipes works as followed. They get valid S2 format. They do something with it. Finally they send changed but still valid S2 format to the next Pipe in line. We get desired pipeline by combaning appropriate pipes. The main purpose of Cli is to parse input arguments and based on them, it build pipeline from pipes. In its curent version every pipe can be included at most ones and in predetermed unmutable order. In most cases the order of pipes doesnt matter. In thoes that does the corect order can be achieved by reapingly calling Cli, thought for some very special task it may be more efficent to build a program that can directly use pipes.

In next section we will briefly explain parsing and how Cli build pipes together. In the section to follow we will explain flags and functionalities of coresponding pipe along side with bare bone examples focused on how to use them in Cli. The implementation of pipes will not be discused here. In the last section there will be some more realistic examples.

3 Parsing

For every option that we want to include into our pipeline we must include its flag followed by arguments if option needs them. Order of flags doesnt matter. It is inportant thought that if flag has any arguments, this arguments are directly after flag and in corect order.

If there is problem with the input arguments (flags + arguments) or file acces, Cli will stop with a brif explanation why.

Cli stats by parsing input arguments. After Cli parses flags and their argumentsit it starts including pipes coresponding to flags. Eventhoug all the pipes are complitly mutualy compatible, Cli will look only for pipes that needs input when input is given and vice versa. It isnt mandatory to include output callback, thought without it all the work will be in vain. At the end it runs the pipeline, writes any posible errors that ocured during execution of Cli and in the if everything was successful it will print **CLI finished**. Warning: From point of Cli none existing or pointless pipeline is perfectly fine.

4 Options

The structure of following subsections will be:

- functionalities of the option

- Flag and argument in the following structure:
  - -Flag
  - mandatory argument 1
  - mandatory argument 2

    ...
  - mandatory argument n
  - optional argument
- example.

As we mentioned before Cli builds pipes in predetermined order therefore we will list options in the same order as their pipes (if they represent one) will be in pipeline.

4.1 Help

Option help doesnt actualy corespond to any pipe. Insted it prints basic info on how to use Cli. Any aditional options are discarded.

- -h

Print help :

–h

4.2 Input

Option input also doesnt corespond to any pipe. Its purpuse is to read lines from S2 file saved on disk and give them to the next pipe in line. Directories must be valid. It has optional secondary directory.

- -i
  - primary directory
  - secondary directory

Read file1.s2 from disk :

```
–i ./directory1/directory2/file1.s2
```

4.3 Merge

Option merge merges two S2 files into one S2 file. It needs option input with primary and secondary directory.

- -m

merge two files provided in option input :

–m

4.4 Data

This option filters lines. If we want to discard all coments there must be 1 in argument on first place from right to left. If we want to discard all special messagas there must be 1st in argument on 2nd place from right to left. If we want to discard all meta data there must be 1 in argument on 3rd place from right to left. If we want to discard all packets there must be 1 in argument on 4th place from right to left. Warning in curent version of Data filtering we do not allow discarding meta data.

- -fd
  - data

Discard comments and packets :

```
-fd 0110
```

4.5 Comments

This option filters coments based on the regex provided in argument.

- -fc
  - regex

Keep only comments containing word Hello :

```
-fc (.*s|)Hello(s.*|)
```

4.6 Special messages

This option filters special message. It keeps messagas that have same who and what as in arguments and suits regex. It needs option input.

- -fs
  - who
  - what
  - regex

Keep only messages from T about U containing word Hello :

```
-fs T U (.*s|)Hello(s.*|)
```

4.7 Handles

This option filters packages based on handles. To include handle #i, put 1 in position i+1 (from right to left) in the argument, to exclude it, put 0. It needs option input.

- -fh
  - handles

Keep only packages with handle 0,1 and 4 :

```
-fh 10011
```

4.8 Filter time

This option filters lines with time. We keep only lines inside time interval. Start is included, end is exclusive. If 3rd optional argument is true we give comments and special messagas last known time and filter them accordingly. It needs option input.

- -ft
  - start [s]
  - time [s]
    - aproximate

Keep only lines from 5th second till 15th second :

```
-ft 5 15 true
```

4.9 Change time

This option changes timespams by adding them argument. If argument is too negative (firt line with time would have negative timestamp) it changes it so the first line wth time will have timestamp 0. It needs option input.

- -ct
  - delay [ns]

add timestamps 2s :

```
-ct 2E9
```

4.10 Change datetime

This option changes date and time in meta data. We can only change date time backwards. This option also changes timestamps so the absolute time doesnt change. Argument must be date, time and timezone in ISO format. It needs option input.

- -cdt
  - dateTimeZone

change date time :

```
-cdt 2018-01-31T10:30:10.554+0100
```

4.11 Process time

This option localy changes time with least squares method. It needs option input.

- -p

process time :

```
-p
```

4.12 Statistics

Produces basic statistics about S2 file and saves it into file provided in argument. If argument is omitted statistics will be printed to standard output insted. It needs option input.

- -s
  - directory

print statistics :

```
-s
```

4.13 Output

This option saves results of previous pipes. Based on extension of file provided in argument it will save in either txt,csv or S2 format. If we provide only extension it will print result on standard output in coresponding format. It needs option input.

- -o
  - directory

save result in csv format :

```
-o ./directory/file.csv
```

4.14 Generate

This option tries to generates new S2 file. It considers Disconects are scattered randomly across whole S2 file. When disconect ocurs machine stops recoding and resets counters. Consequently android doesnt get any packets . It needs option output and filter time.

- -g
  - seed for random [long]
  - frequency in Hz [float] (around 128 for PCARD)
  - frequency change [0..1]
  - percentage missing [0..1]
  - normal delay in s [double]
  - big delay chance [0..1]
  - big delay in s [double]
  - number of disconects

save result in csv format :

```
-o ./directory/file.csv
```