HTTP Log monitoring console program

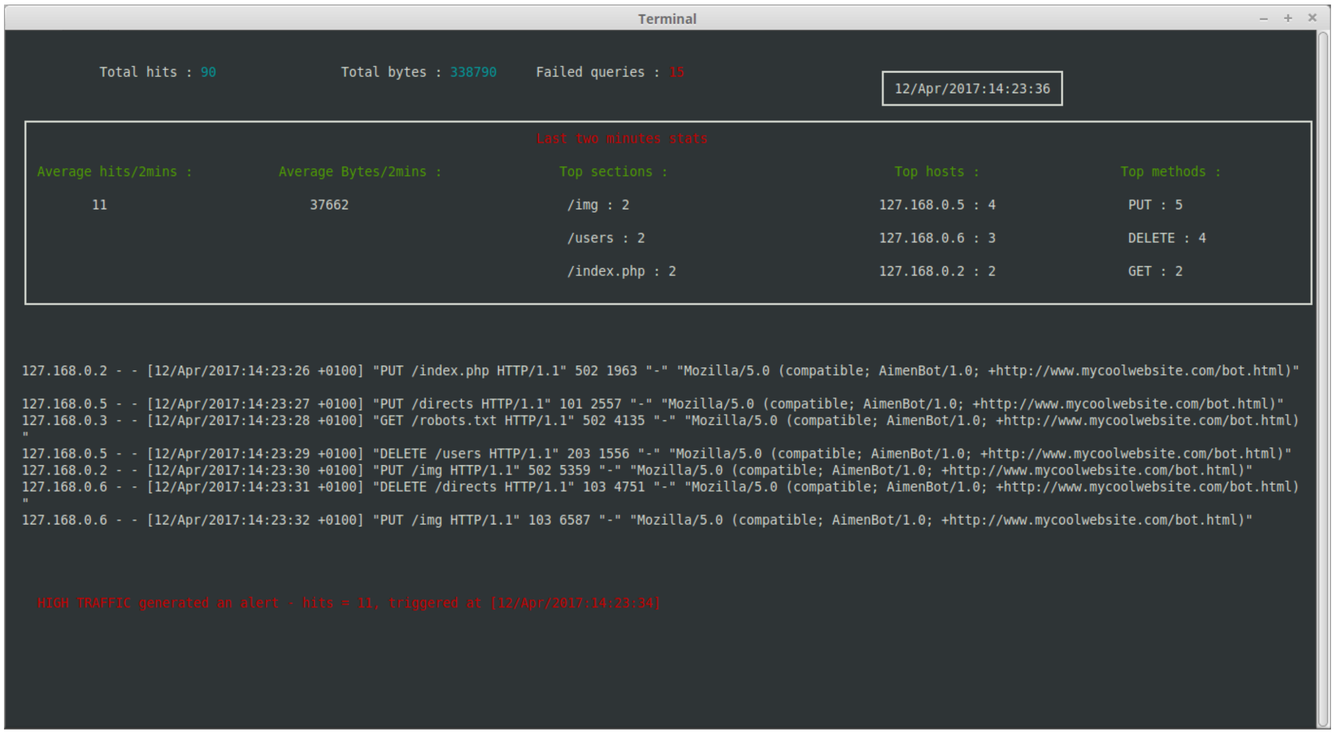
LOUAFI Aïmen

HTTP log monitoring console program

* Create a simple console program that monitors HTTP traffic on your machine:
* Consume an actively written-to w3c-formatted HTTP access log
* Every 10s, display in the console the sections of the web site with the most hits (a section is defined as being what's before the second '/' in a URL. i.e. the section for "http://my.site.com/pages/create' is "http://my.site.com/pages"), as well as interesting summary statistics on the traffic as a whole.
* Make sure a user can keep the console app running and monitor traffic on their machine
* Whenever total traffic for the past 2 minutes exceeds a certain number on average, add a message saying that "High traffic generated an alert - hits = {value}, triggered at {time}"
* Whenever the total traffic drops again below that value on average for the past 2 minutes, add another message detailing when the alert recovered
* Make sure all messages showing when alerting thresholds are crossed remain visible on the page for historical reasons.
* Write a test for the alerting logic
* Explain how you'd improve on this application design

I created this application by using Python 3. This works only on Unix systems (tested on Linux).

Here's a screenshot of the result (launched on Linux Mint terminal)



```
        Total hits : 90              Total bytes : 338790     Failed queries : 15
                                                                                   12/Apr/2017:14:23:36

                                          Last two minutes stats

 Average hits/2mins :          Average Bytes/2mins :          Top sections :                    Top hosts :              Top methods :

       11                             37662                     /img : 2                   127.168.0.5 : 4              PUT : 5

                                                                /users : 2                 127.168.0.6 : 3              DELETE : 4

                                                                /index.php : 2             127.168.0.2 : 2              GET : 2


127.168.0.2 - - [12/Apr/2017:14:23:26 +0100] "PUT /index.php HTTP/1.1" 502 1963 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)"

127.168.0.5 - - [12/Apr/2017:14:23:27 +0100] "PUT /directs HTTP/1.1" 101 2557 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)"
127.168.0.3 - - [12/Apr/2017:14:23:28 +0100] "GET /robots.txt HTTP/1.1" 502 4135 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)
"
127.168.0.5 - - [12/Apr/2017:14:23:29 +0100] "DELETE /users HTTP/1.1" 203 1556 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)"
127.168.0.2 - - [12/Apr/2017:14:23:30 +0100] "PUT /img HTTP/1.1" 502 5359 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)"
127.168.0.6 - - [12/Apr/2017:14:23:31 +0100] "DELETE /directs HTTP/1.1" 103 4751 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)
"
127.168.0.6 - - [12/Apr/2017:14:23:32 +0100] "PUT /img HTTP/1.1" 103 6587 "-" "Mozilla/5.0 (compatible; AimenBot/1.0; +http://www.mycoolwebsite.com/bot.html)"



  HIGH TRAFFIC generated an alert - hits = 11, triggered at [12/Apr/2017:14:23:34]
```

At the top, you have the stats about the whole log file : total hits, total bytes, number of failed queries.

On the middle, you get the stats from the last two minutes, which include the average hits per two minutes, the average bytes, the top sections, top hosts and top methods.

Right under, a few log entries are displayed. These log entries are from the last two minutes.

At the bottom, you get the alerts : the alert generated message and the alert recovered one. You can scroll through them.

The program is divided into 6 python scripts, plus one for the alert test.

- LogRegistry : saves the log entries and the interesting stats. Updates them regularly.
- LogEntry : the class which represents log entries
- LogGen : a random log entries generator
- Display : generates the interface and updates it. Uses python’s curse.
- initialize : resizes the terminal size (for some obscure reason, resizing at the beginning of the program generates a curse error, but running it a second time works well, so I decided to create this initialize script).
- LogMonitor : the main script, which runs all the others.

I also created the AlertTest file : basically, it puts the threshold at 9, generates 10 entries after a short amount of time, and then stops. So you will get an alert for crossing the threshold, and after the two minutes time, the alert will be recovered.

Running the program :

FIRST OF ALL:

You have to execute the initialize.py script. This is important, as not doing it will result into a curses error. This script will resize the terminal so the application can be displayed properly.

If you want to generate log entries by using log gen, you have to execute:

```
python3 –i LogGen.py filename.txt
```

This will generates log entries onto the file: filename.txt

Now, if you want to run the LogMonitor, execute:

```
python3 –i LogMonitor.py –t threshold filename.txt
```

You have to replace threshold by the desired threshold for the alerts, and replace filename by the desired log file.

If you want to try the alert system out, you have to run :

```
python3 –i AlertTest.py
```

Possible improvements :

- Allowing to read from multiple log files.
- Adding a way to detect new log file creation and monitoring them directly.
- You can only monitor files that are on your computer. Allowing the information to be uploaded and accessible from anywhere could be an improvement. Reading the flux directly from the internet.
- Improve the design. Possibly by not using the terminal.
- Implement a way to output the stats, to a website or another application.