Final Report - Life of a web request

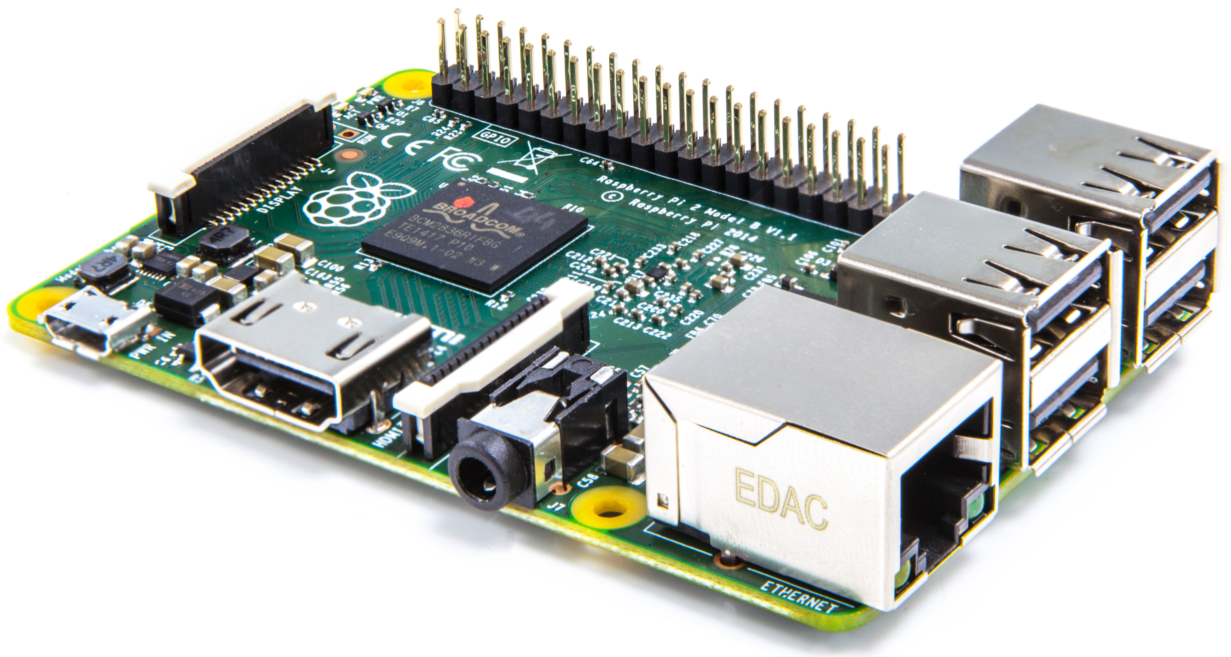
Valentin Facko, Theresia Deiss, Ozren Mendas

Introduction

The web is used to simplify communications between people and machines around the world. The goal of our project is to get a better understanding of the web. We wanted to learn how to create our own web page. To perform this mission we had to learn how to program and how to design our own web page.

Materials and Method

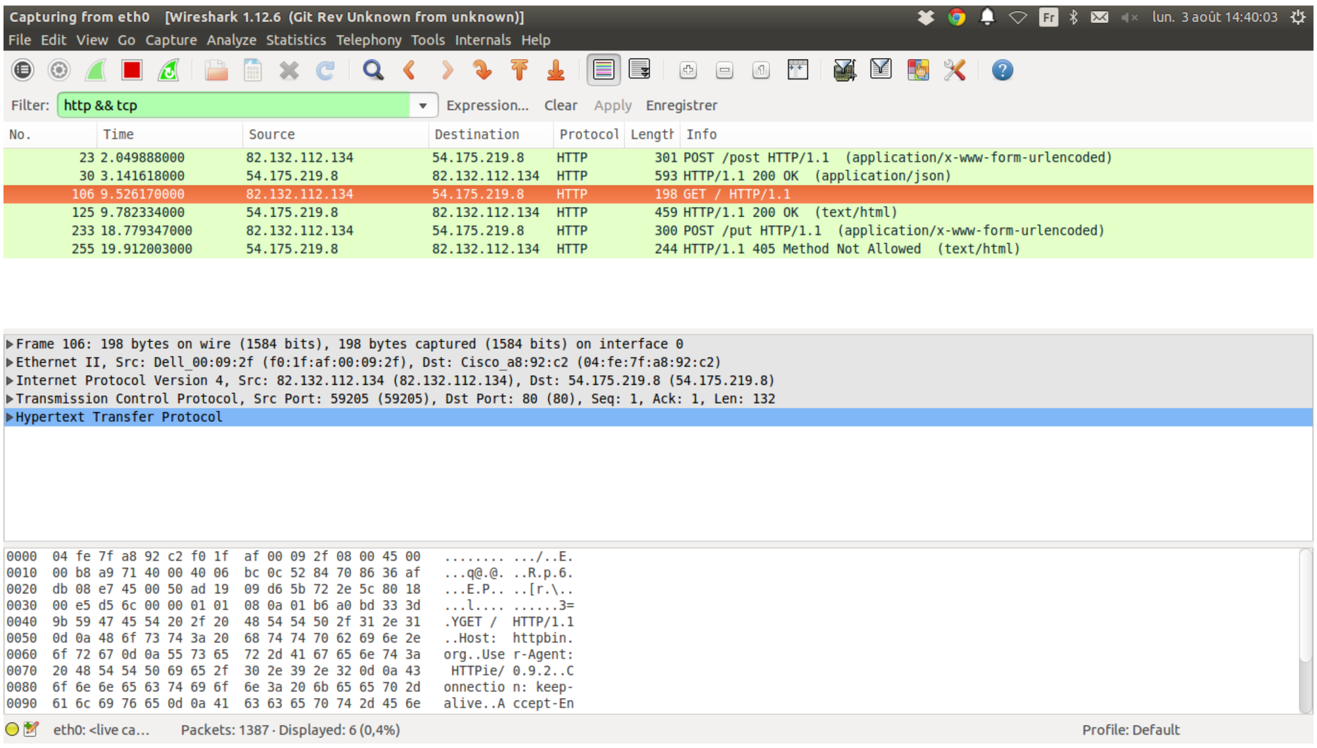
Raspberry Pi



For creating a simple website the only equipment we needed was a computer, a stable internet connection and basic knowledge of programming. To make it as simple as possible we used the raspberry pi, which is a cheap computer using the operating system raspbian.

Wireshark

The first step of our project was to understand the basic communication between computers and the internet. For that we used the program wireshark, which makes it possible to see the transferred information between computers, in particular the ones from the HTTP protocol. In our researches we got to know, that HTTP is based on different verbs, such as GET for getting a document and POST for posting data on the web. For a wider understanding we also dealt with the DNS and IP-address, which make it possible to address your request and getting your answer correctly.


Capturing from eth0 [Wireshark 1.12.6 (Git Rev Unknown from unknown)]
lun. 3 août 14:40:03
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Enregistrer

Filter: http && tcp

```
No.    Time          Source          Destination     Protocol Length Info
    23 2.049888000   82.132.112.134  54.175.219.8    HTTP       301 POST /post HTTP/1.1  (application/x-www-form-urlencoded)
    30 3.141618000   54.175.219.8    82.132.112.134  HTTP       593 HTTP/1.1 200 OK  (application/json)
   106 9.526170000   82.132.112.134  54.175.219.8    HTTP       198 GET / HTTP/1.1
   125 9.782334000   54.175.219.8    82.132.112.134  HTTP       459 HTTP/1.1 200 OK  (text/html)
   233 18.779347000  82.132.112.134  54.175.219.8    HTTP       300 POST /put HTTP/1.1  (application/x-www-form-urlencoded)
   255 19.912003000  54.175.219.8    82.132.112.134  HTTP       244 HTTP/1.1 405 Method Not Allowed  (text/html)
```

```
▶Frame 106: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
▶Ethernet II, Src: Dell_00:09:2f (f0:1f:af:00:09:2f), Dst: Cisco_a8:92:c2 (04:fe:7f:a8:92:c2)
▶Internet Protocol Version 4, Src: 82.132.112.134 (82.132.112.134), Dst: 54.175.219.8 (54.175.219.8)
▶Transmission Control Protocol, Src Port: 59205 (59205), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 132
▶Hypertext Transfer Protocol
```

```
0000  04 fe 7f a8 92 c2 f0 1f  af 00 09 2f 08 00 45 00   ........ .../..E.
0010  00 b8 a9 71 40 00 40 06  bc 0c 52 84 70 86 36 af   ...q@.@. ..R.p.6.
0020  db 08 e7 45 00 50 ad 19  09 d6 5b 72 2e 5c 80 18   ...E.P.. ..[r.\..
0030  00 e5 d5 6c 00 00 01 01  08 0a 01 b6 a0 bd 33 3d   ...l.... ......3=
0040  9b 59 47 45 54 20 2f 20  48 54 54 50 2f 31 2e 31   .YGET /  HTTP/1.1
0050  0d 0a 48 6f 73 74 3a 20  68 74 74 70 62 69 6e 2e   ..Host:  httpbin.
0060  6f 72 67 0d 0a 55 73 65  72 2d 41 67 65 6e 74 3a   org..Use r-Agent:
0070  20 48 54 54 50 69 65 2f  30 2e 39 2e 32 0d 0a 43    HTTPie/ 0.9.2..C
0080  6f 6e 6e 65 63 74 69 6f  6e 3a 20 6b 65 65 70 2d   onnectio n: keep-
0090  61 6c 69 76 65 0d 0a 41  63 63 65 70 74 2d 45 6e   alive..A ccept-En
```

eth0: <live ca... Packets: 1387 · Displayed: 6 (0,4%) Profile: Default

Python

As already mentioned the web page should be created with the raspberry pi, on which the python programming language is already installed. Python is an easy to learn language, what makes it interesting for beginners. To get to know python, we started reading the tutorial Learning Python the hard way, so we were able to program functions on our own. For getting content on our website we decided to post the temperatures of our computer, which can be measured with a sensor on the raspberry pi. To show changes on that measurements we decided to measure the current temperature every ten seconds and then upload the last five values, which are stored, on the website.

Database redis

For storing the temperatures of the raspberry pi we decided to use a database. The advantage of a database in contrast to a normal file is that you can query data more efficiently than with a file. Furthermore, the data can be easy manipulated. We decided to use Redis, because of its simplicity and also because mature library already exist to use it in Python.

```
import redis
r = redis.Redis()
r.rpush("temperature list", 42)
```

Flask

Flask is a lightweight web framework which is used to make a website. It is a python library. The way it works is that it handles HTTP requests, dispatches code that generates HTML, and creates an HTTP response with that content.

```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run()
```

Templates

Because we want to upload every ten seconds the current temperatures, it is not possible to create a static website. For generating dynamically every web pages, we use Jinja2 as a template library. When a page is requested, the function that handle it loads the template in memory and then populates the templates with the right value. Because of the stiff measurements the data uploaded on the website are up to date all the time.

For example this Jinja2 template:

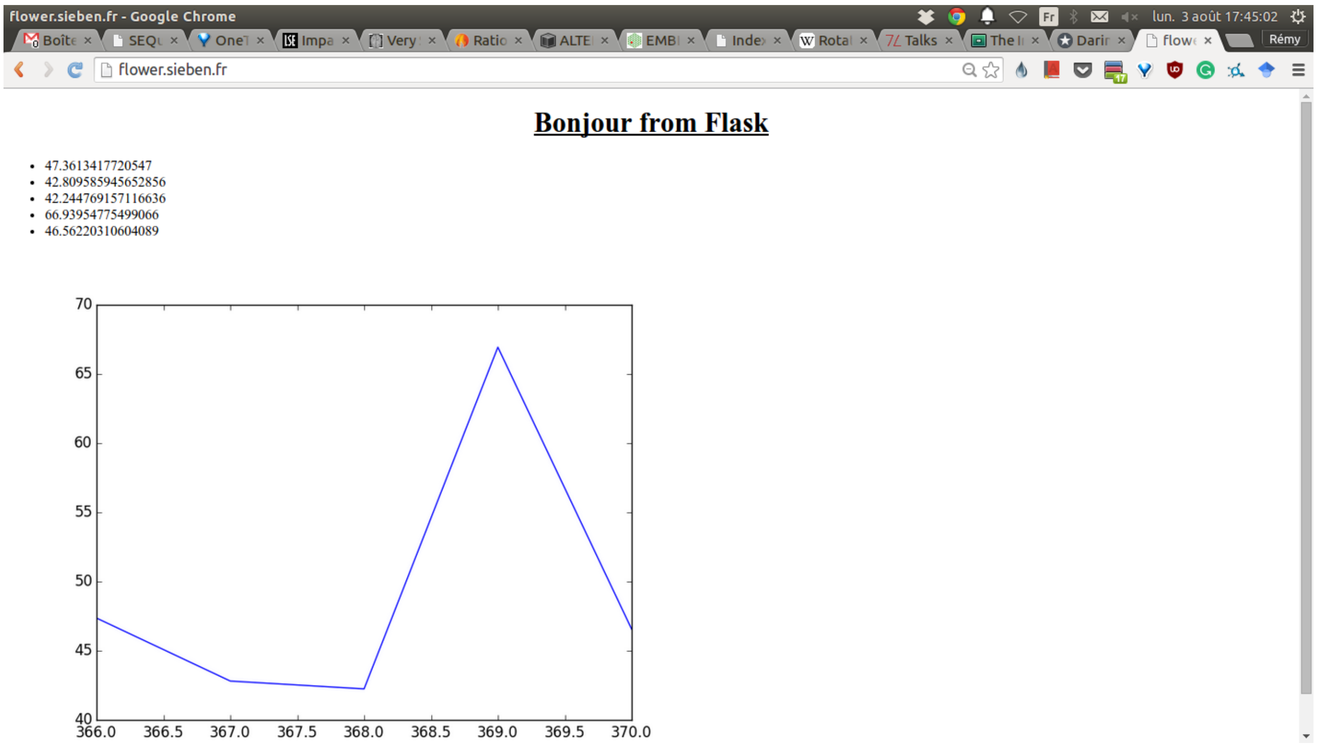
```
```
<ul>

</ul>
```
```

will be transformed into this HTML code ready to be read by the web browser:

```
```
<ul>
  <li>42</li>
  <li>42.69</li>
  <li>42.96</li>
</ul>
```
```

Results



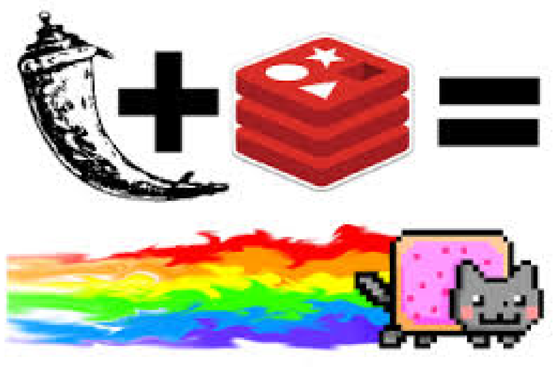
Bonjour from Flask

- 47.3613417720547
- 42.809585945652856
- 42.244769157116636
- 66.93954775499066
- 46.56220310604089

Our final website shows the last five measurements of the temperature of the raspberry pi as well as it compares them in the graph below.

By creating our own website we can quickly see that many programs are available, for free, on the internet, but just a few are really helpful. We used as a tool flask, tembling and the database, because these enabled us to create an easy script, where all steps, functions and store are working together, modify each other and can be dynamically uploaded. Through our lessons in theory we have now a better understanding of the relation between the internet and the web but also their benefit. The web is a very useful tool to look up information on a very cheap way and getting always the latest information.

In addition, we obtain now a knowledge about different programs, stores and languages, their advantages and disadvantages and especially we are able to understand better what we are telling the computer by giving a command and what the computer is telling us by giving back an error. Having experience in Computer Science leads to a logical way of thinking, being able to understand abstract models and transferring this quality also into other disciplines of science.



Conclusion

We managed to finish a web page that fulfilled our goal. As a next step, we could improve the design of the website and make it look pretty on more devices such as mobile phones and tablets. We could also include as a further step a full-duplex connection such as the ones provided by websockets. Which makes it possible to receive new updates without having to explicitly send a new request for them.

References

- The Internet by Jessica McKellar

Raspberry Pi

- how to make a sensor

Python programming

- Python
- Learn Python the hard way

- Flask
- Jinja2

Redis

- Redis
- Try redis

HTML

- Reveal.js
- Mozilla Developer Network

---

remy.leone@telecom-paristech.fr

sieben
remyleone