

Practical work #3

Michel FACERIAS

February 16, 2022

Abstract

We are going to discover some application protocols. This PW contains 2 sections :

- Discovering HTTP ;
- Discovering SMTP.

Contents

1	Analysis of HTTP exchanges	3
1.1	Grabbing with <i>curl</i>	3
1.1.1	Todo : learning <i>curl</i>	3
1.1.2	Question : What application protocols you can use with <i>curl</i> ?	3
1.1.3	Todo : Focus on HTTP	3
1.1.4	Question : what is the command line to grab <i>http://hal9000.facerias.org/test.html</i> ?	3
1.1.5	Todo : download the file, capturing frames	3
1.1.6	Question : How can you explain the content of the screen afetr download	3
1.1.7	Question : Looking at the capture, can you see same readable information in the HTTP payload and why ?	3
1.1.8	Todo : Redo the same job, but using an other file	3
1.1.9	Question : How can you explain the content of the screen after download ?	3
1.1.10	Question : Looking at the capture can you see same readable information in the HTTP payload and why ?	3
1.2	Advanced <i>curl</i> grabbing	3
1.2.1	Todo : Use <i>curl</i> to see the HTTP header	3
1.2.2	Question : how many types of parts can you see on the console text ?	3
1.2.3	Question : compare what's on the console and the capture payload and explain it	3
2	Sending an email	3
2.1	Learning SMTP ptotocol	4
2.1.1	Todo : get informations about the SMTP protocole	4
2.1.2	Question : what is the minimum dialog needed to send an email ?	4
2.1.3	Question : which transport and port is used for SMTP ?	4
2.1.4	Question : what should be the command to connect to a SMTP server ?	4
2.1.5	Todo : try to send an email, using google account	4
2.1.6	Question : What happens, and why ?	4
2.2	Dealing with SSL/TLS	4
2.2.1	Todo : Send an email to yourself	4

1 Analysis of HTTP exchanges

This case study will help you understand the HTTP protocol.

1.1 Grabbing with *curl*

We are going to use a command line web tool to download files.

1.1.1 Todo : learning *curl*

Open a console and have a look at curl manual using : `man curl`

1.1.2 Question : What application protocols you can use with *curl* ?

1.1.3 Todo : Focus on HTTP

Read carefully the section about HTTP protocol.

1.1.4 Question : what is the command line to grab `http://hal9000.facerias.org/test.html` ?

1.1.5 Todo : download the file, capturing frames

Open *wireshark* and adjust a capture filter.

Then use curl to download `http://hal9000.facerias.org/test.html` and stop the capture.

1.1.6 Question : How can you explain the content of the screen after download

1.1.7 Question : Looking at the capture, can you see same readable information in the HTTP payload and why ?

1.1.8 Todo : Redo the same job, but using an other file

Start a adapted capture and in the console, download `http://hal9000.facerias.org/sources.list`. Then stop the capture.

1.1.9 Question : How can you explain the content of the screen after download ?

1.1.10 Question : Looking at the capture can you see same readable information in the HTTP payload and why ?

1.2 Advanced *curl* grabbing

1.2.1 Todo : Use *curl* to see the HTTP header

Download `http://hal9000.facerias.org/sources.list` again, but adding `-v` option to *curl*.

1.2.2 Question : how many types of parts can you see on the console text ?

1.2.3 Question : compare what's on the console and the capture payload and explain it

Using the previous capture and your last verbose download, compare what's on the screen and the HTTP payload. What is your conclusion ?

2 Sending an email

In this section we are going to learn about SMTP protocol.

2.1 Learning SMTP protocol

2.1.1 Todo : get informations about the SMTP protocole

Use a search engine or go directly to <https://flowmailer.com/en/article/smtp-explained>

2.1.2 Question : what is the minimum dialog needed to send an email ?

2.1.3 Question : which transport and port is used for SMTP ?

2.1.4 Question : what should be the command to connect to a SMTP server ?

2.1.5 Todo : try to send an email, using google account

Try `nc smtp.gmail.com 465` and say `EHLO`.

2.1.6 Question : What happens, and why ?

2.2 Dealing with SSL/TLS

`nc` is not SSL/TLS capable. Instead, you can use `openssl` as a client :

- on dedicated port : `openssl s_client -connect smtp.gmail.com:465`
- on submission port, with STARTTLS command : `openssl s_client -connect smtp.laposte.net:587 -starttls smtp`

All the rest of the dialog is the same as with `nc`. But that's not all you need. After saying `EHLO`, you may add an authentication method, because of anti-spamm rules.

In case you use one of these methods :

- AUTH LOGIN

```
C: AUTH LOGIN
S: 334 VXNlcm5hbWU6                (Username: as B64)
C: TXktVXNlcm5hbWU=              (My-Username as B64)
S: 334 UGFzc3dvcmQ6              (Password: as B64)
C: TXktUGFzc3dvcmQ=              (My-Password as B64)
S: 235 Authentication successful
```

- AUTH PLAIN 1

```
C: AUTH PLAIN
S: 334
C: AE15LVVzZXJhYmU1AE15LVBhc3N3b3JkCg== (\0My-Username\0My-Password as B64)
S: 235 Authentication successful
```

- AUTH PLAIN 2

```
C: AUTH PLAIN AE15LVVzZXJhYmU1AE15LVBhc3N3b3JkCg== (\0My-Username\0My-Password as B64)
S: 235 Authentication successful
```

The `base64` command line tool will help you to code the answer to give to the server.

```
$ echo toto | base64
dG90bwo=

$ echo dG90bwo= |base64 -d
toto
```

2.2.1 Todo : Send an email to yourself

Use all together to send you an email using your EPF account.