

# **Compte rendu**

## **Développement Avancé : TP 4**

*Laurent Giustignano*

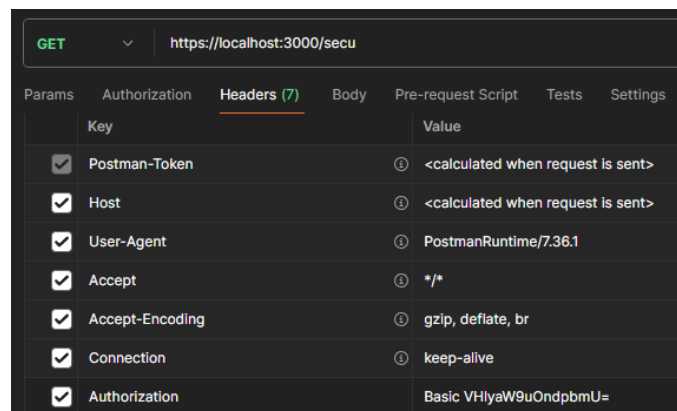


Par Steven TEA 303

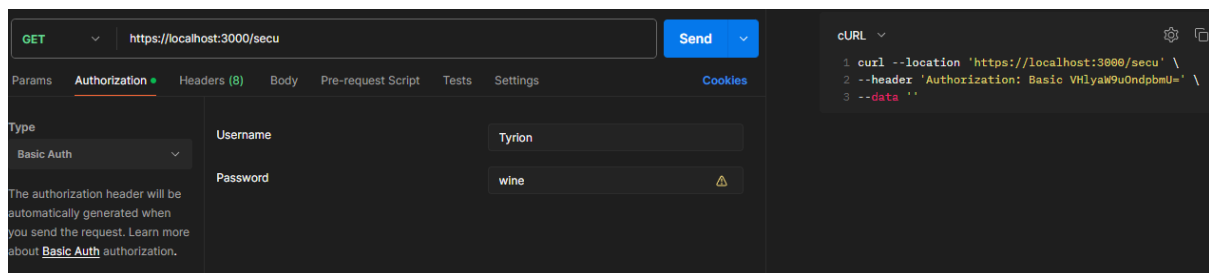
## Étape 1

Pour la première étape du TP, nous essayons d'intégrer un système d'authentification avec une basic auth. J'ai donc essayé d'aller à la route /secu, qui nécessite une authentification basique.

Je m'y suis pris de différentes manières, tout d'abord, j'ai utilisé le site <https://www.base64encode.org/fr/>, pour convertir en base64 avec le username/password, donc convertir "Tyrion:wine" qui donne "VHlyaW9uOndpbmU=", pour mettre dans le header Authorization Basic VHlyaW9uOndpbmU=.



J'ai ensuite fait avec une manière moins compliqué, en utilisant le basic auth dans la catégorie Authorization, en mettant mon username "Tyrion" et le password "wine", la conversion est faite automatiquement par Postman.



Ces deux méthodes fonctionnent, mais la deuxième est privilégiée pour sa simplicité.

La méthode after de Fastify, permet que le contenu dans la méthode soit exécuté après les méthodes appelées auparavant. Dans notre cas, l'utilisation de la méthode after sert à ce que l'enregistrement de fastifyBasicAuth s'effectue avant l'ajout de la route /secu car, cette route utilise le basic auth, et nécessite donc son chargement au préalable.

## Étape 2

Maintenant nous allons mettre le protocole https sur notre serveur web. Pour créer un certificat, nous utilisons dans un premier temps, générer une clé privée 2048bits, puis

créer un fichier de demande de signature de certificat, dans mon cas, le fichier s'appelle server.req.

```
steven@steven-virtual-machine:~$ openssl req -new -key server.key -out server.req
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Ile de France
Locality Name (eg, city) []:Montgeron
Organization Name (eg, company) [Internet Wdgets Pty Ltd]:StevenCorp
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:Steven
Email Address []:steven@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:steven
An optional company name []:
steven@steven-virtual-machine:~$ ll
total 96
drwxr-xr-x 16 steven steven 4096 févr. 17 01:14 ./
drwxr-xr-x  3 root  root  4096 déc. 10 02:53 ../
-rw-r--r--  1 steven steven 742 févr. 14 14:40 .bash_history
-rw-r--r--  1 steven steven 220 déc. 10 02:53 .bash_logout
-rw-r--r--  1 steven steven 3771 déc. 10 02:53 .bashrc
drwx----- 12 steven steven 4096 févr. 14 14:26 .cache/
drwx----- 13 steven steven 4096 déc. 25 17:39 .config/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Desktop/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Documents/
drwxr-xr-x  2 steven steven 4096 déc. 30 15:11 Downloads/
drwx-----  3 steven steven 4096 déc. 10 02:57 .local/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Music/
drwxr-xr-x  3 steven steven 4096 déc. 25 17:39 Pictures/
-rw-r--r--  1 steven steven 807 déc. 10 02:53 .profile
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Public/
-rw-r--r--  1 steven steven 12 févr. 14 14:40 .python_history
-rw-r--r--  1 steven steven 1704 févr. 17 01:13 server.key
-rw-rw-r--  1 steven steven 1070 févr. 17 01:14 server.req
drwx-----  5 steven steven 4096 déc. 25 17:38 snap/
-rw-r--r--  1 steven steven  0 févr. 14 14:19 .sudo_as_admin_successful
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Templates/
drwxrwxr-x  3 steven steven 4096 févr. 14 14:18 tmp/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Videos/
drwxrwxr-x  6 steven steven 4096 févr. 14 14:37 .vscode-server/
-rw-rw-r--  1 steven steven 183 févr. 14 14:36 .wget-hsts
```

Une fois la clé privée et la demande de signature créées, nous allons auto signer un certificat. Donc, en entrant la clé privée, la demande et une durée d'expiration, un fichier de sortie est généré, server.crt. Je peux désormais tester le certificat, en ouvrant un serveur SSL/TLS sur mon ordinateur.

```
steven@steven-virtual-machine:~$ openssl x509 -req -days 365 -in server.req -signkey server.key -out server.crt
Certificate request self-signature ok
subject=C = FR, ST = Ile de France, L = Montgeron, O = StevenCorp, CN = Steven, emailAddress = steven@gmail.com
steven@steven-virtual-machine:~$ ll
total 100
drwxr-xr-x 16 steven steven 4096 févr. 17 01:14 ./
drwxr-xr-x  3 root  root  4096 déc. 10 02:53 ../
-rw-r--r--  1 steven steven 742 févr. 14 14:40 .bash_history
-rw-r--r--  1 steven steven 220 déc. 10 02:53 .bash_logout
-rw-r--r--  1 steven steven 3771 déc. 10 02:53 .bashrc
drwx----- 12 steven steven 4096 févr. 14 14:26 .cache/
drwx----- 13 steven steven 4096 déc. 25 17:39 .config/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Desktop/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Documents/
drwxr-xr-x  2 steven steven 4096 déc. 30 15:11 Downloads/
drwx-----  3 steven steven 4096 déc. 10 02:57 .local/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Music/
drwxr-xr-x  3 steven steven 4096 déc. 25 17:39 Pictures/
-rw-r--r--  1 steven steven 807 déc. 10 02:53 .profile
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Public/
-rw-r--r--  1 steven steven 12 févr. 14 14:40 .python_history
-rw-rw-r--  1 steven steven 1285 févr. 17 01:14 server.crt
-rw-r--r--  1 steven steven 1704 févr. 17 01:13 server.key
-rw-rw-r--  1 steven steven 1070 févr. 17 01:14 server.req
drwx-----  5 steven steven 4096 déc. 25 17:38 snap/
-rw-r--r--  1 steven steven  0 févr. 14 14:19 .sudo_as_admin_successful
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Templates/
drwxrwxr-x  3 steven steven 4096 févr. 14 14:18 tmp/
drwxr-xr-x  2 steven steven 4096 déc. 10 02:57 Videos/
drwxrwxr-x  6 steven steven 4096 févr. 14 14:37 .vscode-server/
-rw-rw-r--  1 steven steven 183 févr. 14 14:36 .wget-hsts
```

En utilisant Postman, nous pouvons afficher de nombreuses informations sur le certificat. Il y a tout d'abord, les méthodes de chiffrement supportées par le serveur, ici, TLS version 1.3, 1.2, SSL version 3, etc...

```
GET https://192.168.40.129:4567

Params Authorization Headers (7) Body Pre-request Script Tests Settings
body Cookies Headers (1) Test Results
Pretty Raw Preview Visualize HTML

4 s_server -accept 4567 -cert server.crt -key server.key -www -state
5 Secure Renegotiation IS NOT supported
6 Ciphers supported in s_server binary
7 TLSv1.3 :TLS_AES_256_GCM_SHA384 TLSv1.3 :TLS_CHACHA20_POLY1305_SHA256
8 TLSv1.3 :TLS_AES_128_GCM_SHA256 TLSv1.2 :ECDHE-ECDSA-AES256-GCM-SHA384
9 TLSv1.2 :ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2 :DHE-RSA-AES256-GCM-SHA384
10 TLSv1.2 :ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2 :ECDHE-RSA-CHACHA20-POLY1305
11 TLSv1.2 :DHE-RSA-CHACHA20-POLY1305 TLSv1.2 :ECDHE-ECDSA-AES128-GCM-SHA256
12 TLSv1.2 :ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 :DHE-RSA-AES128-GCM-SHA256
13 TLSv1.2 :ECDHE-ECDSA-AES256-SHA384 TLSv1.2 :ECDHE-RSA-AES256-SHA384
14 TLSv1.2 :DHE-RSA-AES256-SHA256 TLSv1.2 :ECDHE-ECDSA-AES128-SHA256
15 TLSv1.2 :ECDHE-RSA-AES128-SHA256 TLSv1.2 :DHE-RSA-AES128-SHA256
16 TLSv1.0 :ECDHE-ECDSA-AES256-SHA TLSv1.0 :ECDHE-RSA-AES256-SHA
17 SSLv3 :DHE-RSA-AES256-SHA TLSv1.0 :ECDHE-ECDSA-AES128-SHA
18 TLSv1.0 :ECDHE-RSA-AES128-SHA SSLv3 :DHE-RSA-AES128-SHA
19 TLSv1.2 :RSA-PSK-AES256-GCM-SHA384 TLSv1.2 :DHE-PSK-AES256-GCM-SHA384
20 TLSv1.2 :RSA-PSK-CHACHA20-POLY1305 TLSv1.2 :DHE-PSK-CHACHA20-POLY1305
21 TLSv1.2 :ECDHE-PSK-CHACHA20-POLY1305 TLSv1.2 :AES256-GCM-SHA384
22 TLSv1.2 :PSK-AES256-GCM-SHA384 TLSv1.2 :PSK-CHACHA20-POLY1305
23 TLSv1.2 :RSA-PSK-AES128-GCM-SHA256 TLSv1.2 :DHE-PSK-AES128-GCM-SHA256
24 TLSv1.2 :AES128-GCM-SHA256 TLSv1.2 :PSK-AES128-GCM-SHA256
25 TLSv1.2 :AES256-SHA256 TLSv1.2 :AES128-SHA256
26 TLSv1.0 :ECDHE-PSK-AES256-CBC-SHA384 TLSv1.0 :ECDHE-PSK-AES256-CBC-SHA
27 SSLv3 :SRP-RSA-AES-256-CBC-SHA SSLv3 :SRP-AES-256-CBC-SHA
28 TLSv1.0 :RSA-PSK-AES256-CBC-SHA384 TLSv1.0 :DHE-PSK-AES256-CBC-SHA384
29 SSLv3 :RSA-PSK-AES256-CBC-SHA SSLv3 :DHE-PSK-AES256-CBC-SHA
30 SSLv3 :AES256-SHA TLSv1.0 :PSK-AES256-CBC-SHA384
31 SSLv3 :PSK-AES256-CBC-SHA TLSv1.0 :ECDHE-PSK-AES128-CBC-SHA256
32 TLSv1.0 :ECDHE-PSK-AES128-CBC-SHA SSLv3 :SRP-RSA-AES-128-CBC-SHA
33 SSLv3 :SRP-AES-128-CBC-SHA TLSv1.0 :RSA-PSK-AES128-CBC-SHA256
34 TLSv1.0 :DHE-PSK-AES128-CBC-SHA256 SSLv3 :RSA-PSK-AES128-CBC-SHA
35 SSLv3 :DHE-PSK-AES128-CBC-SHA SSLv3 :AES128-SHA
36 TLSv1.0 :PSK-AES128-CBC-SHA256 SSLv3 :PSK-AES128-CBC-SHA
37 ---
38 Ciphers common between both SSL end points:
39 TLS_AES_128_GCM_SHA256 TLS_AES_256_GCM_SHA384 TLS_CHACHA20_POLY1305_SHA256
40 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-RSA-AES256-GCM-SHA384
41 ECDHE-ECDSA-AES256-GCM-SHA384 ECDHE-ECDSA-CHACHA20-POLY1305 ECDHE-RSA-CHACHA20-POLY1305
42 ECDHE-ECDSA-AES128-SHA ECDHE-RSA-AES128-SHA ECDHE-ECDSA-AES256-SHA
43 ECDHE-RSA-AES256-SHA AES128-GCM-SHA256 AES256-GCM-SHA384
44 AES128-SHA AES256-SHA
```

## Étape 3

Pour la dernière étape de ce TP, j'ai tout d'abord commencé par générer les clés de chiffrement en utilisant la fonction présente dans le fichier .Readme du projet. J'ai complété le fichier jwt.js pour y ajouter les clés privée et publique dans le serveur Fastify.

Étant donné qu'il y a deux serveurs à lancer, un sur le port 3000 pour l'authentification et le 4000 pour la page d'accueil, j'ai eu des difficultés concernant le partage des données, comme la liste des utilisateurs, j'ai notamment pensé à faire un serveur redis que les deux serveurs lisent pour récupérer les nouveaux utilisateurs, cependant, en regardant mieux les consignes, j'ai saisi que le serveur data au port 4000 ne nécessitait pas de connaître la liste des utilisateurs.



Finalement, nous affichons le message adapté en fonction du champ "role" dans l'objet JSON chiffré en JWT.