Testing HTTPS with OpenSSL

wechris.github.io/tips-tutorials/openssl/https/docker/2018/01/26/Testing-HTTPS-with-OpenSSL

26 January 2018

Posted by Christian Weiß on January 26, 2018

OpenSSL

Introduction

OpenSSL is a versatile command line tool that can be used for a large variety of tasks related to Public Key Infrastructure (PKI) and HTTPS (HTTP over TLS).

The Docker Image

The docker image creates a self-signed certificate ('/CN=localhost') for test purpose and start the `s_server` with the self-signed certificate. The internal port: 8443. I use the container to verfiy and test a SSL/TSL connection with OpenSSL to my HTTPS clients. Sometimes I receive some error like 'SSL Handshake Failure' and the docker image helps me to analyze the HTTPS problems.

Clone the reporitory wechris/docker-openssl-sslcheck

The Dockerfile:

```
FROM alpine:3.5

LABEL version="1.0"
LABEL description="Docker Images with OpenSSL \
The s_server command implements a generic SSL/TLS server \
which listens for connections on a given port using SSL/TLS."

# install openssl and some helpful tools
RUN apk add --no-cache vim curl openssl wget

## And create self-signed ssl keys for test purposes (bind mount proper ones to
running container)
RUN openssl req -new -x509 -nodes -out /home/ssl.crt -keyout /home/ssl.key -days
3650 -subj '/CN=localhost'

##
CMD openssl s_server -accept 8443 -cert /home/ssl.crt -key /home/ssl.key -www

# http and httpd ports. You can map these to whatever host ports you want with -p
EXPOSE 8443

# Default env vars for httpd. You can override these at runtime if you want to
ENV SERVERNAME localhost
ENV ADMINEMAIL root@localhost
```

Build the docker image:

```
docker build -t wechris/opensslserver .
```

and start the container:

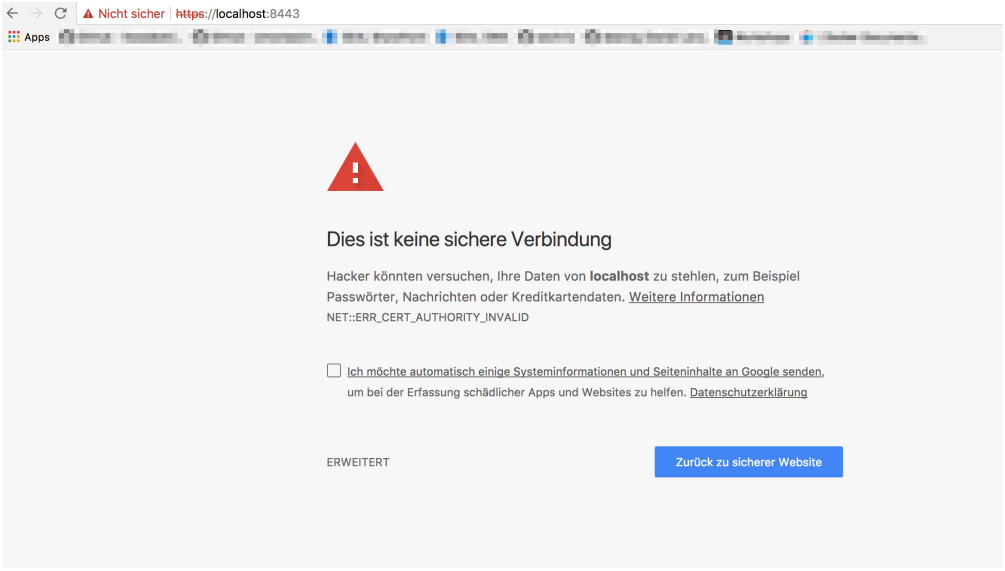
```
docker run -it -p 8443:8443 --name opensslserver wechris/opensslserver
```

Accessing the s_server via web browser

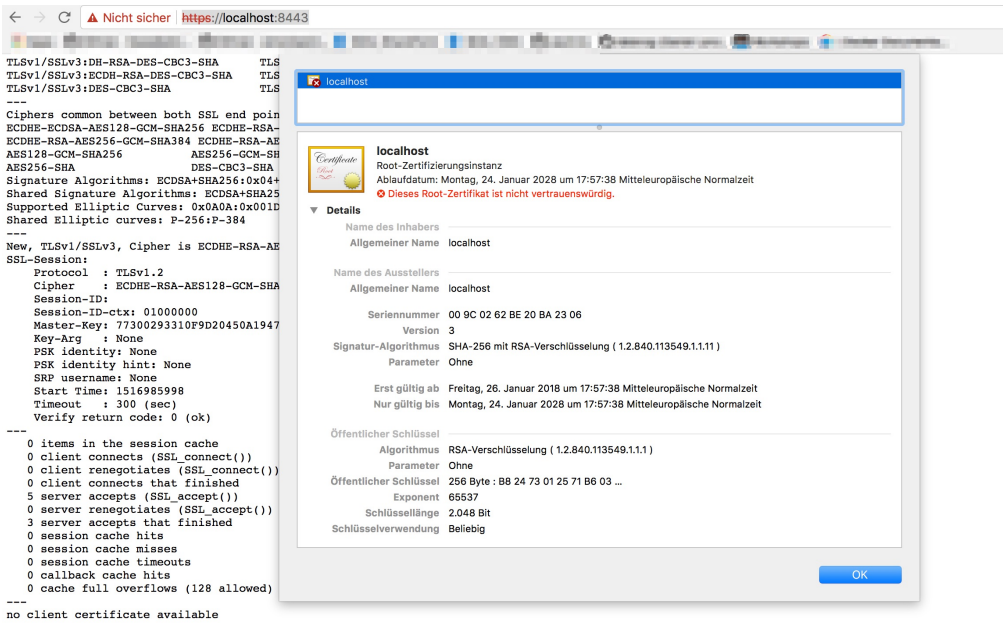
To test openssl s_server enter the following URL in the web browser:

https://localhost:8443

If everything went right, we will see a privacy error, this is because we're using a self-signed certificate (created in the preparation)



After passing the privacy error, we will see the response from the openssl s_server internal webserver:



Accessing the s_server via openssl s_client

OpenSSL comes with a client tool that can be used to connect to a secure server. The `s_client` command implements a generic SSL/TLS client which connects to a remote host using SSL/TLS. It is a very useful diagnostic tool for SSL servers.

It is possible to access the container to test the HTTPS server or another one with the OpenSSL s_client inside the container

```
docker exec -it opensslserver /bin/sh
```

...or start a s_client from your host system if openssl is installed.

Test the docker image with the OpenSSL `openssl s_client -connect localhost:8443`

```
$ openssl s_client -connect localhost:8443
CONNECTED(00000005)
depth=0 CN = localhost
verify error:num=18:self signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
---
Certificate chain
 0 s:/CN=localhost
   i:/CN=localhost
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIC+zCCAeOgAwIBAgIJAJwCYr4guiMGMA0GCSqGSIb3DQEBCwUAMBQxEjAQBgNV
BAMMCWxvY2FsaG9zdDAeFw0xODAxMjYxNjU3MzhaFw0yODAxMjQxNjU3MzhaMBQx
EjAQBgNVBAMMCWxvY2FsaG9zdDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoC
ggEBALgkcwElcbYDXV2x8DZY2gOBHbaXc1otHiUQrl3M3OxbAkAzlrfntV0UDIRB
TldgJ100Kiek1YumvXyn/KIPiOMyDxlf4DPUgRb6RB8MquNjtnNKl/tutX6kULUd
Tmj6jRmXddKtbtdNo3IvbqadYqbc1VIaQBxLCtUHnFAOZYGXZkJa3SA5bs0yBUt9
XPJEq7+lOmu5kzsuycqEwStC+p1VKTH8x38emBKhwW5RViLSvxuWuHc+V3GZG8ti
4LlTe/2WxboyOfPgVO70OMeee8eOMbLSWSbgCUiY6bMt6Wxl8q/nW9vJFn53CfBH
IN4u6Yj8IxsdMqGCEXRT3Z10EncCAwEAAaNQME4wHQYDVR0OBBYEFJVFe58IA9y6
3/f9GcPPgAgHlH+zMB8GA1UdIwQYMBaAFJVFe58IA9y63/f9GcPPgAgHlH+zMAwG
A1UdEwQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAJGa14srMkJKLbhgGxSJtW6w
sk/7fSngMyFcP3oHUVKwO9iArI94xOkb2lVIvaLmFPogW+DVjwUTKT8fhB6+ytVr
PlLLDyca8AvtoTUG2QAC/3vCbmmGTvMGSlxS9vUA2Rja1t58Rcgceij7QbhhQFsr
3ZuXXuZz5TagwnTFC3PZe70o2Him2pSZX0RGV3g9Du6bKysGOmzhWODynKnbzCx5
eg144NurPMLed4HqkoCSm2BMc2x7wBUaT+H/9epCWMmagjdYZz1peGVvs2b0vD+t
jaczYQfD6zwWeiGuFS28LT4uem9Ujn0eQSaoJxJfZlhDO//QLlVf7fcAaZ5R7eY=
-----END CERTIFICATE-----
subject=/CN=localhost
issuer=/CN=localhost
---
No client certificate CA names sent
---
SSL handshake has read 1421 bytes and written 444 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES256-GCM-SHA384
    Session-ID: B23ED9CD2067C6438E6550DF54FFEB396E92DDBB9F8F8703934D225C80151006
    Session-ID-ctx:
    Master-Key:
AE358E3A66E965405EB34711BA38C30157BF52231F5EB1F6D0BFBC1C268C2A85024AE0BD67B3752A444

    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
    0000 - 6b c0 1d 1a 5f a8 8a 39-ee 1d 26 0f 21 a0 5a ff   k..._..9..&.!.Z.
    0010 - f3 a1 f3 8a 5f 81 fa 12-0d 6f 62 3c c4 a4 c4 42   ...._....ob<...B
    0020 - fb ea de 33 25 ef cc f6-ba f4 bd bd 7f 1b 3e 4c   ...3%.........>L
    0030 - f3 97 68 c9 03 7c 43 97-e9 c4 9f 55 48 9c 14 17   ..h..|C....UH...
    0040 - c2 a3 36 4b c2 6f 7b 05-66 29 89 8e 43 36 26 0c   ..6K.o{.f)..C6&.
```

```
    0050 - 15 12 aa 46 07 66 a5 7b-86 7a 49 c6 a2 06 f9 9d   ...F.f.{.zI.....
    0060 - ce 50 3d 55 d0 3d de 3a-b2 03 6a 67 f7 d2 21 f6   .P=U.=.:..jg..!.
    0070 - 82 e9 48 16 42 38 1d b6-35 0d d6 f5 49 7f b6 09   ..H.B8..5...I...
    0080 - a2 4c 82 23 88 93 99 37-79 b1 1b 57 77 97 64 4c   .L.#...7y..Ww.dL
    0090 - 25 53 93 7d 00 72 a9 f8-21 2a 94 26 63 2f 84 d5   %S.}.r..!*.&c/..

    Start Time: 1516986082
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
```

reference:

- openssl s_server
- openssl s_client