Bezpieczeństwo Aplikacji WEB

Zadanie 2

Prowadzący:
Mgr inż. Przemysław Świercz

wtorek 8:15
Miłosz Jagodziński CBE

Apache

Zadanie

Cel:

Bezpieczna konfiguracja serwera webowego
Proszę przygotować konfigurację dla serwera Apache ORAZ Nginx spełniające następujące kryteria:

Zadanie 2: (kontynuacja zadania 1)

Proszę przygotować dwa certyfikaty klienckie dla certyfikatu serwera z Zadania 1 p.1 - User A, User B
Ścieżka /only-user-a (wraz ze wszystkimi pod ścieżkami) ma być dostępna wyłącznie dla klientów z certyfikatem User A
Ścieżka /only-user-b(wraz ze wszystkimi pod ścieżkami) ma być dostępna wyłącznie dla klientów z certyfikatem User B
Ścieżka /user-a-or-b(wraz ze wszystkimi pod ścieżkami) ma być dostępna wyłącznie dla klientów z certyfikatem User A lub User B
(punkt dodatkowy, nieobowiązkowy) Podścieżka /info dla ścieżek z p. 2,3,4 (czyli np. /only-user-a/info) wyświetli informacje o użytkowniku odczytane z jego certyfikatu klienckiego.

Środowisko pracy:

Ubuntu 20.04 Virtualbox, Docker

Przebieg ćwiczenia Apache

W zadaniu tym wykorzystano już istniejące pliki z poprzedniego zadania. Niektóre parametry i komendy trzeba było zmienić, aby móc wykonać zadanie drugie. Również rozbudowano dockerfile.

Tworzenie kluczy i certyfikatów

Pierwszym krokiem było wygenerowanie certyfikatow CA za pomocą komendy:

```
openssl req -newkey rsa:2048 -nodes -keyform PEM -keyout selfsigned-ca.key -x509 -days 3650 -outform PEM -out selfsigned-ca.crt
```

```
milosz@milosz-VirtualBox:~/test$ openssl req -newkey rsa:2048 -nodes -keyform PEM -keyout selfsigned-ca.key -x509 -days 3650 -outform PEM -out selfsigned-ca.crt
Generating a RSA private key
.....+++++
writing new private key to 'selfsigned-ca.key'
-----
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]:PL
State or Province Name (full name) [Some-State]:Polska
Locality Name (eg, city) []:Wroclaw
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Pwr
Organizational Unit Name (eg, section) []:CBE
Common Name (e.g. server FQDN or YOUR name) []:klienci
Email Address []:242027@student.pwr.edu.pl
milosz@milosz-VirtualBox:~/test$
```

Jak widać powyżej uzupełniono już parametry informacyjne.

Następnie wygenerowano klucz serwera dla serwera o nazwie selfsigned.key.

```
openssl genrsa -out selfsigned.key 2048
```

```
milosz@milosz-VirtualBox:~/test$ openssl genrsa -out selfsigned.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
.....+++++
e is 65537 (0x010001)
```

Potem stworzono certyfikat selfsigned.crt. Z hasłem ”milosz”.

```
openssl req -new -key selfsigned.key -out selfsigned.csr
```

```
milosz@milosz-VirtualBox:~/test$ openssl req -new -key selfsigned.key -out selfsigned.csr
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]:PL
State or Province Name (full name) [Some-State]:Polska
Locality Name (eg, city) []:Wroclaw
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Pwr
Organizational Unit Name (eg, section) []:CBE
Common Name (e.g. server FQDN or YOUR name) []:klienci
Email Address []:242027@student.pwr.edu.pl

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:milosz
An optional company name []:milosz
milosz@milosz-VirtualBox:~/test$
```

Następnie wykonano podpis CSR dla Apache.

```
openssl x509 -req -in selfsigned.csr -CA selfsigned-ca.crt -CAkey selfsigned-ca.key -set_serial 100 -days 365 -outform PEM -out selfsigned.crt
```

```
milosz@milosz-VirtualBox:~/test$ openssl x509 -req -in selfsigned.csr -CA selfsigned-ca.crt -CAkey selfsigned-ca.key -set_serial 100 -days 365 -outform PEM -out selfsigned.crt
Signature ok
subject=C = PL, ST = Polska, L = Wroclaw, O = Pwr, OU = CBE, CN = klienci, emailAddress = 242027@student.pwr.edu.pl
Getting CA Private Key
```

Dalej zmodyfikowano poprzedni plik dockerfile z zadania 1, usuwając stare certyfikaty i klucze, a dodając nowe, które zostały wcześniej utworzone.

```
 8         #sciezka do stronki
 9 COPY ./selfsigned.key /usr/local/apache2/conf/selfsigned.key
10 COPY ./selfsigned.crt /usr/local/apache2/conf/selfsigned.crt
11         #wgranie certyfikatów
```

Później przystąpiono do zrobienia podwójnego uwierzytelniania dla klientów. Pierwszy krok na tym etapie to stworzenie certyfikatów i kluczy klienckich. Poniżej polecenie:

`openssl genrsa -out osoba-A.key 2048`

```
milosz@milosz-VirtualBox:~/test$ openssl genrsa -out osoba-A.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
........+++++
......+++++
e is 65537 (0x010001)
```

CSR dla klienta A:

`openssl req -new -key osoba-A.key -out osoba-A.csr`

```
milosz@milosz-VirtualBox:~/test$ openssl req -new -key osoba-A.key -out osoba-A.csr
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]:PL
State or Province Name (full name) [Some-State]:Polska
Locality Name (eg, city) []:Wroclaw
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Pwr
Organizational Unit Name (eg, section) []:CBE
Common Name (e.g. server FQDN or YOUR name) []:klientA
Email Address []:242027A@student.pwr.edu.pl

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:milosz
An optional company name []:klientA
milosz@milosz-VirtualBox:~/test$
```

Poniżej widać jak stworzono podpis dla klienta A przy użyciu polecenia:

`openssl x509 -req -in osoba-A.csr -CA selfsigned-ca.crt -CAkey selfsigned-ca.key -set_serial 101 -days 365 -outform PEM -out osoba-A.crt`

```
milosz@milosz-VirtualBox:~/test$ openssl x509 -req -in osoba-A.csr -CA selfsigned-ca.crt -CAkey selfsigned-ca.key -set_serial 101 -days 365 -outform PEM -out osoba-A.crt
Signature ok
subject=C = PL, ST = Polska, L = Wroclaw, O = Pwr, OU = CBE, CN = klientA, emailAddress = 242027A@student.pwr.edu.pl
Getting CA Private Key
```

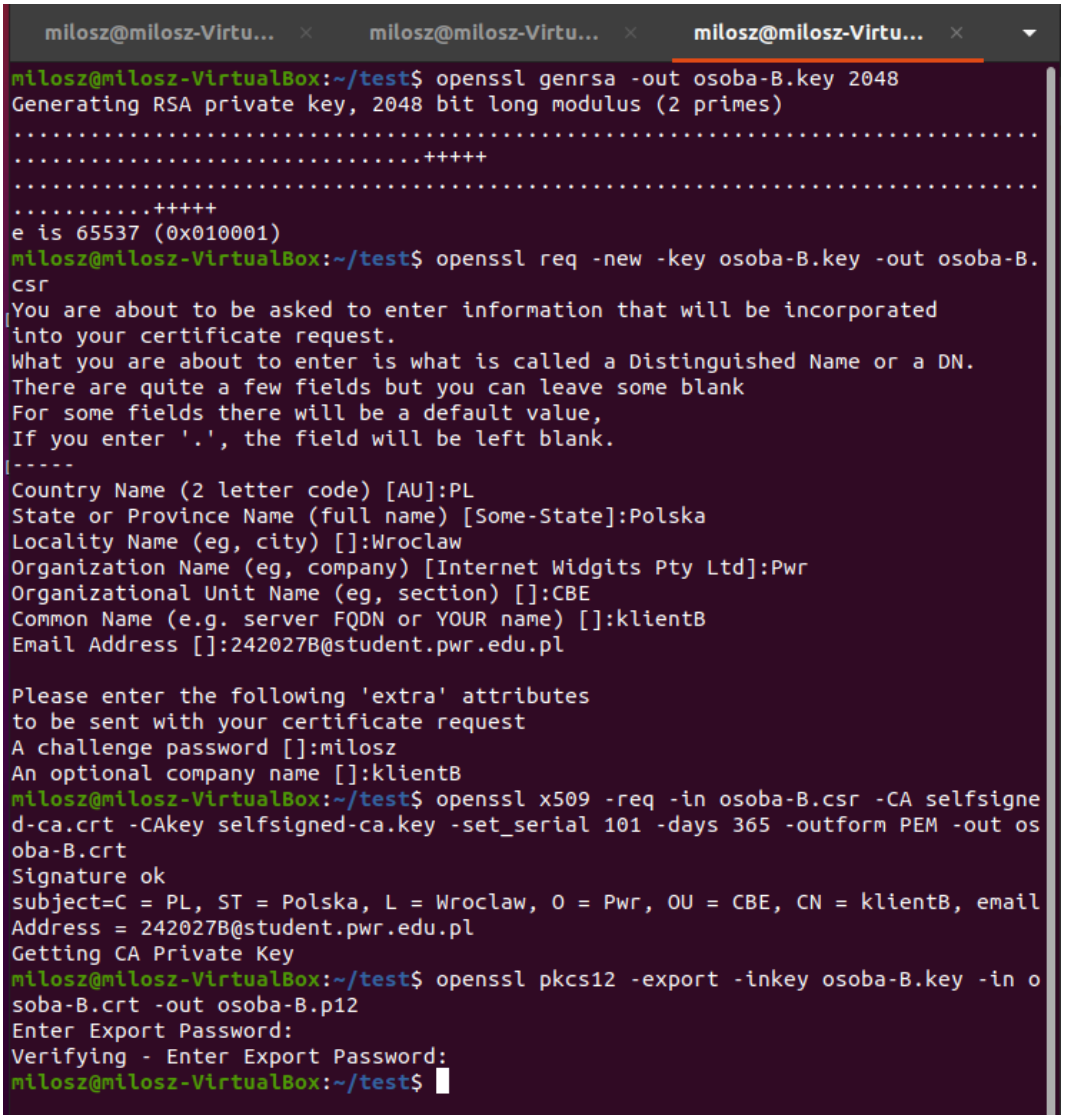
Ostatni krok to połączenie certyfikatu klienta i klucza klienta A w pakiet .p12. Potrzebny jest on dla użytkownika, aby ten mógł potwierdzić swoją tożsamość wchodząc na ścieżkę w przeglądarce. Wpisano hasło miłosz. Polecenie:

`openssl pkcs12 -export -inkey osoba-A.key -in osoba-A.crt -out osoba-A.p12`

```
milosz@milosz-VirtualBox:~/test$ openssl pkcs12 -export -inkey osoba-A.key -in osoba-A.crt -out osoba-A.p12
Enter Export Password:
Verifying - Enter Export Password:
```

Takie same kroki wykonano dla klienta B tylko, certyfikaty i klucze przypisane zostały pod klienta B.

1. `openssl genrsa -out osoba-B.key 2048`
2. `openssl req -new -key osoba-B.key -out osoba-B.csr`
3. `openssl x509 -req -in osoba-B.csr -CA selfsigned-ca.crt -CAkey selfsigned-ca.key -set_serial 101 -days 365 -outform PEM -out osoba-B.crt`
4. `openssl pkcs12 -export -inkey osoba-B.key -in osoba-B.crt -out osoba-B.p12`

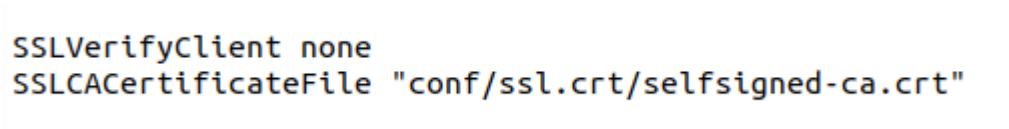


Po utworzeniu certyfikatów i kluczy kliencki przystąpiono do edycji i konfiguracji dockerfile i pliku httpd-ssl.conf. Była to bardzo wymagająca praca poprzez metodę prób i błędów i studiowanie dokumentacji Apache.

Konfiguracja pliku httpd-ssl.conf

Wskazano ścieżkę na nasz wygenerowany certyfikat CA i ustawiono opcję, aby klienci nie byli weryfikowani zawsze.

```
SSLVerifyClient none
SSLCACertificateFile "conf/ssl.crt/selfsigned-ca.crt"
```



Następnie przystąpiono do konfiguracji ścieżek i pod ścieżek. Warto zwrócić uwagę na parametry takie jak:

`SSLVerifyClient` - Parametr dla całego serwera lub katalogów. Umożliwia uwierzytelnienie klienta, gdy ten wysyła żądanie połączenia. Również wymusza renegocjację po odczytaniu żądania HTTPS, ale przed wysłaniem odpowiedzi. Reguła ta pozwala wymagać od klientów własnych certyfikatów.
`SSLVerifyDepth` - Łańcuch wskazujący jak góra i dół sprawdza zabezpieczenia.
`SSLOptions` - Kontroluje różne opcje środowiska wykonawczego dla poszczególnych katalogów. `+FakeBasicAuth` - tłumaczy nazwę wyróżniającą podmiotu certyfikatu X.509 klienta na nazwę użytkownika podstawowej autoryzacji HTTP. Oznacza to, że do kontroli dostępu można użyć standardowych metod uwierzytelniania serwera HTTP. Żadne hasło nie jest uzyskiwane od użytkownika; zastępowany jest ciąg „hasło”.
`SSLRequireSSL` - Ta dyrektywa zabrania dostępu, chyba że HTTP przez SSL (tj. HTTPS) jest włączony dla bieżącego połączenia.
`SSLRequire %{SSL_CLIENT_S_DN_CN} eq "klientA"` - Są to parametry, które musi posiadać klient w swoim poświadczeniu.

only-user-a

```
#A z certami juz konfigurowanymi
     <Location "/only-user-a/">
          SSLVerifyClient require
          SSLVerifyDepth
          SSLOptions +FakeBasicAuth
          SSLRequireSSL
          SSLRequire %{SSL_CLIENT_S_DN_CN} eq "klientA"
     </Location>
```

only-user-b

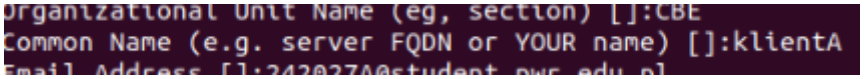
```
#B z certami juz konfigurowanymi
     <Location "/only-user-b">
          SSLVerifyClient require
```

```
            SSLVerifyDepth 5
            SSLOptions +FakeBasicAuth
            SSLRequireSSL
            SSLRequire %{SSL_CLIENT_S_DN_CN} eq "klientB"
      </Location>

user-a-or-b
#A i B z certami już konfigurowanymi
      <Location "/user-a-or-b">
            SSLVerifyClient require
            SSLVerifyDepth 1
            SSLRequire   %{SSL_CLIENT_S_DN_CN}   in   {"klientA",
"klientB"}
      </Location>
```

```
315 #A z certami juz konfigurowanymi
316         <Location "/only-user-a/">
317                 SSLVerifyClient require
318                 SSLVerifyDepth
319                 SSLOptions +FakeBasicAuth
320                 SSLRequireSSL
321                 SSLRequire %{SSL_CLIENT_S_DN_CN} eq "klientA"
322         </Location>
323
324 #B z certami juz konfigurowanymi
325         <Location "/only-user-b">
326                 SSLVerifyClient require
327                 SSLVerifyDepth 5
328                 SSLOptions +FakeBasicAuth
329                 SSLRequireSSL
330                 SSLRequire %{SSL_CLIENT_S_DN_CN} eq "klientB"
331         </Location>
332
333 #A i B z certami juz konfigurowanymi
334         <Location "/user-a-or-b">
335                 SSLVerifyClient require
336                 SSLVerifyDepth 1
337                 SSLRequire %{SSL_CLIENT_S_DN_CN} in {"klientA", "klientB"}
338         </Location>
339
Zwykły tekst ▾   Szerokość tabulacji:
```

Widać jak wskazano na danego klienta. Najważniejszy to parametr `SSLRequire %{SSL_CLIENT_S_DN_CN}`, dzięki niemu, a dokładnie CN wiadomo o jakiego klienta chodzi. Wracamy do naszych ustawień wcześniej. Można wprowadzić dodatkowe parametry w celu lepszego zabezpieczenia. Poniżej parametr Common Name CN, ustawiony wcześniej:



Trzeba było również dodać odpowiednie reguły, aby zadziałało tzn, wskazać wersję TLS:

```
SSLProtocol -all +TLSv1.2
```

```
292 #dodane dodatkowo
293 SSLProtocol -all +TLSv1.2
294 #SSLProtocol all -SSLv3
```

Plik httpd-ssl.conf został skonfigurowany.

Konfiguracja i uruchomienie dockerfile

Gdy skonfigurowano już wszystko przystąpiono z powrotem do dockerfile. Na pierwszy ogień dodano wymagane ścieżki.

```
RUN mkdir /usr/local/apache2/htdocs/only-user-a
RUN mkdir /usr/local/apache2/htdocs/only-user-b
RUN mkdir /usr/local/apache2/htdocs/user-a-or-b
```

Potem dodano nowe i przykładowe pliki .html.

```
27 COPY ./indexklientA.html /usr/local/apache2/htdocs/only-user-a/indexklientA.html
28 COPY ./indexklientB.html /usr/local/apache2/htdocs/only-user-b/indexklientB.html
29 COPY ./indexklientAiB.html /usr/local/apache2/htdocs/user-a-or-b/indexklientAiB.html
```

Wszystko został skonfigurowane można odpalać dockera za pomocą dwóch komend. Pierwsza to pobranie obrazu z naszą konfiguracją. Druga to utworzenie i uruchomienie kontenera.

1. `sudo docker build -t zadanie .`

```
milosz@milosz-VirtualBox:~/apache3$ sudo docker build -t zadanie .
Sending build context to Docker daemon  146.4kB
Step 1/21 : FROM httpd:2.4
 ---> c30a46771695
Step 2/21 : LABEL maintainer = "242027"
 ---> Using cache
 ---> 581067e7d3c5
Step 3/21 : RUN apt-get update
 ---> Using cache
 ---> 6c3d63400323
Step 4/21 : COPY ./index.html /usr/local/apache2/htdocs/
 ---> Using cache
 ---> c3e3a3ce9b8f
Step 5/21 : COPY ./selfsigned.key /usr/local/apache2/conf/selfsigned.key
 ---> Using cache
 ---> 423ad21e18b3
```

```
Removing intermediate container 21a8c9ba8e9b
 ---> 3dcfe4ba9d54
Successfully built 3dcfe4ba9d54
Successfully tagged zadanie:latest
milosz@milosz-VirtualBox:~/apache3$
```

2. `sudo docker run -dit --name dwaapache -p 8082:80 -p 44445:443 zadanie`

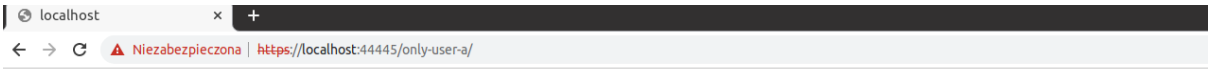
```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ilosz@milosz-VirtualBox:~/apache3$ sudo docker run -dit --name dwaapache -p 8082:80 -p 44445:443 zadanie
```

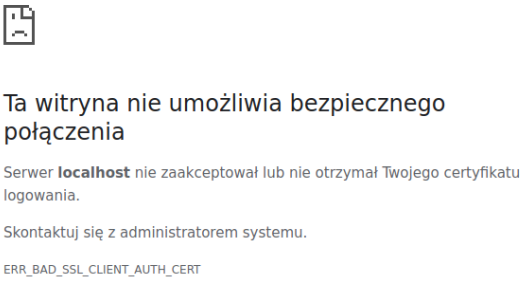
```
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS          PORTS                                                                                  NAMES
449f72fe0e6a   zadanie   "httpd-foreground"   39 seconds ago   Up 2 seconds    0.0.0.0:8082->80/tcp, :::8082->80/tcp, 0.0.0.0:44445->443/tcp, :::44445->443/tcp   dwaapache
milosz@milosz-VirtualBox:~/apache3$
```

Przedstawienie działania certyfikatów i kluczy klienckich

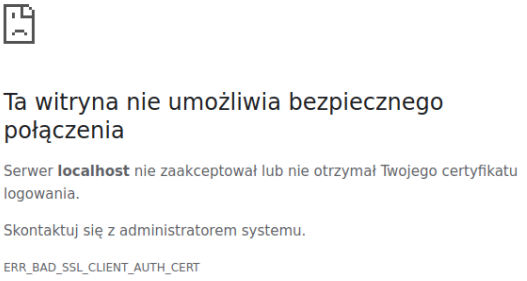
Na ten moment nie dodano jeszcze certyfikatów w przeglądarce, aby pokazać brak dostępu do wyznaczonych ścieżek.

only-user-a

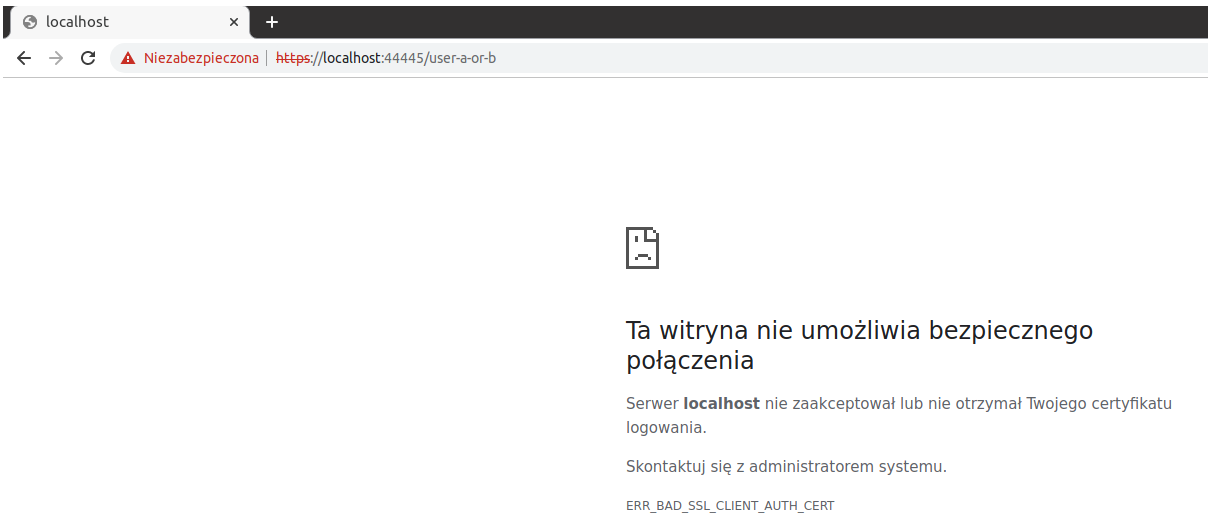




only-user-b

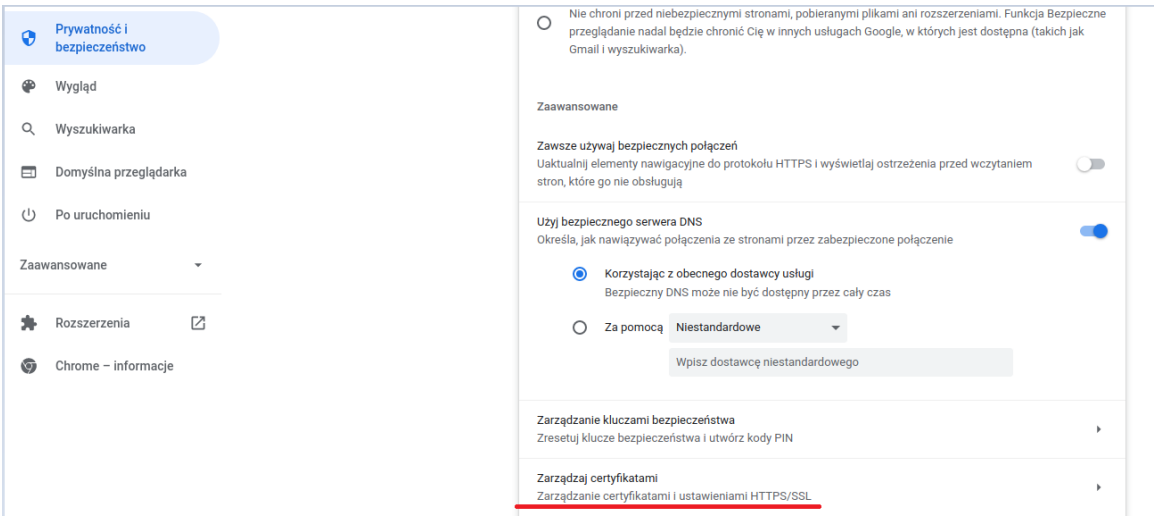


user-a-or-b



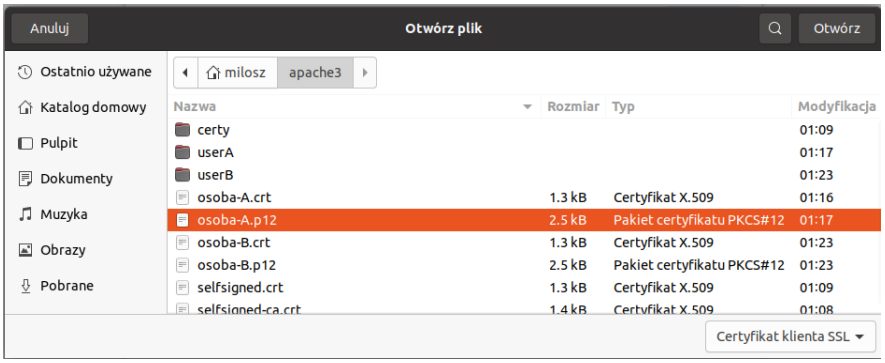
Widać na powyższych obrazkach i na podstawie wyświetlanego komunikatu “ERR_BAD_SSL_CLIENT_AUTH_CERT”, żeby dostać się dalej wymagany jest kliencki certyfikat autoryzujący.

Dalej zaimportowano certyfikaty do przeglądarki Chrome i Firefox w formacie .p12.

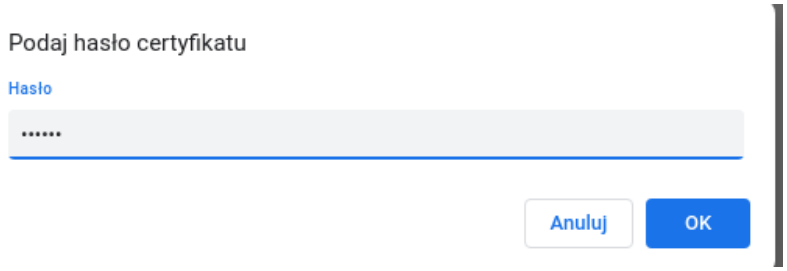


Klient A

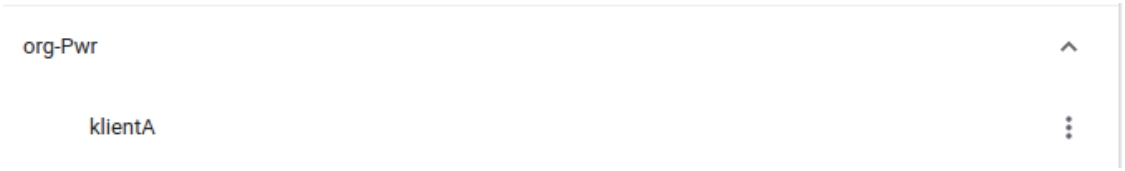
Wpierw sprawdzono klienta A i zaimportowano jego plik .p12.



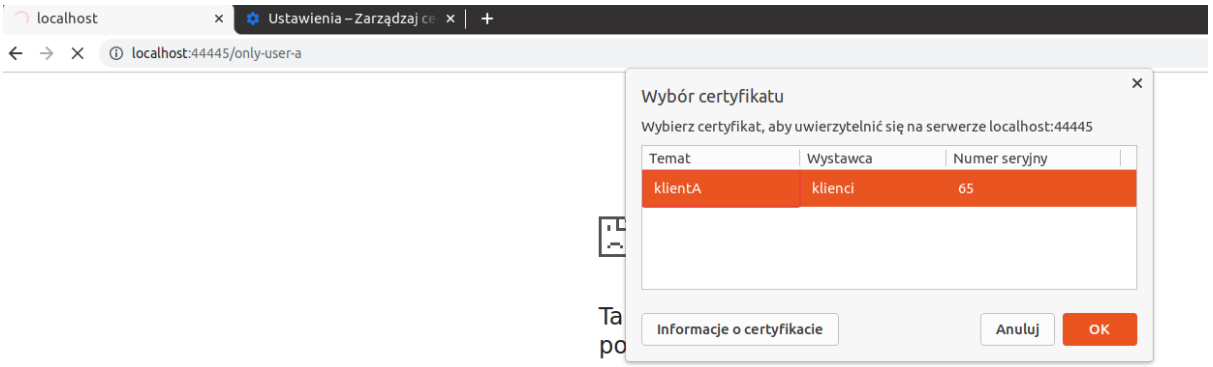
Podano hasło ustanowione wcześniej milosz.



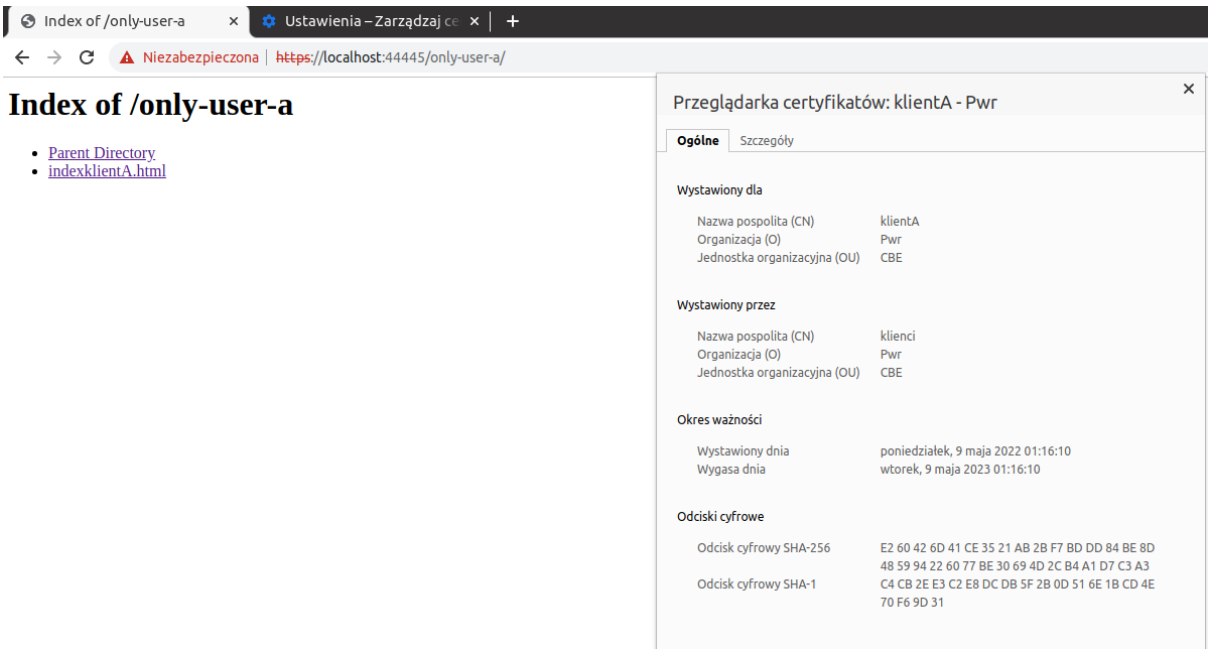
Widać poświadczenia klienta A.



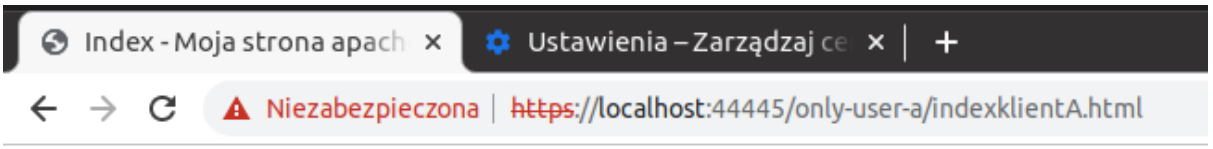
Widać poświadczenie.



Wszystko dobrze działa można wejść do ścieżki only-usera-a, dzięki poświadczeniu klienta A.



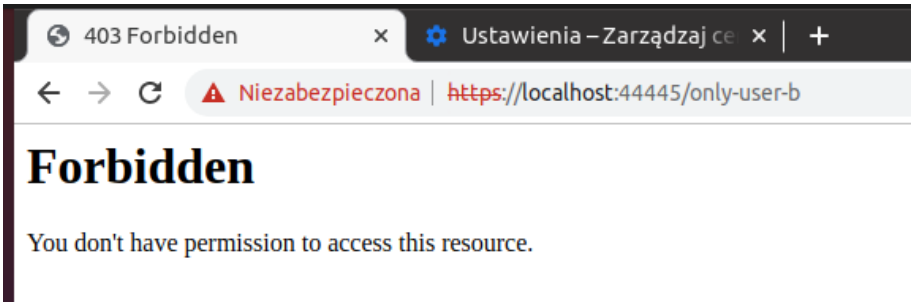
Podścieżka:



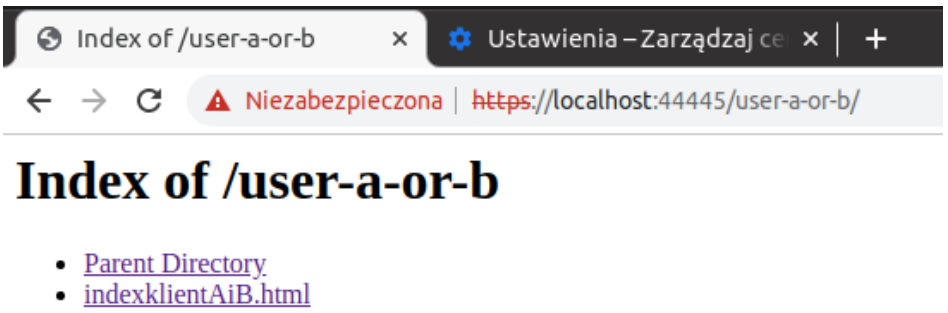
Witaj na mojej stronie projekt WEB klient A

strona na projekt pwr apache Klient A

Natomiast do ścieżki B nie można wejść, ponieważ na ten moment nie posiadano odpowiednich poświadczeń:



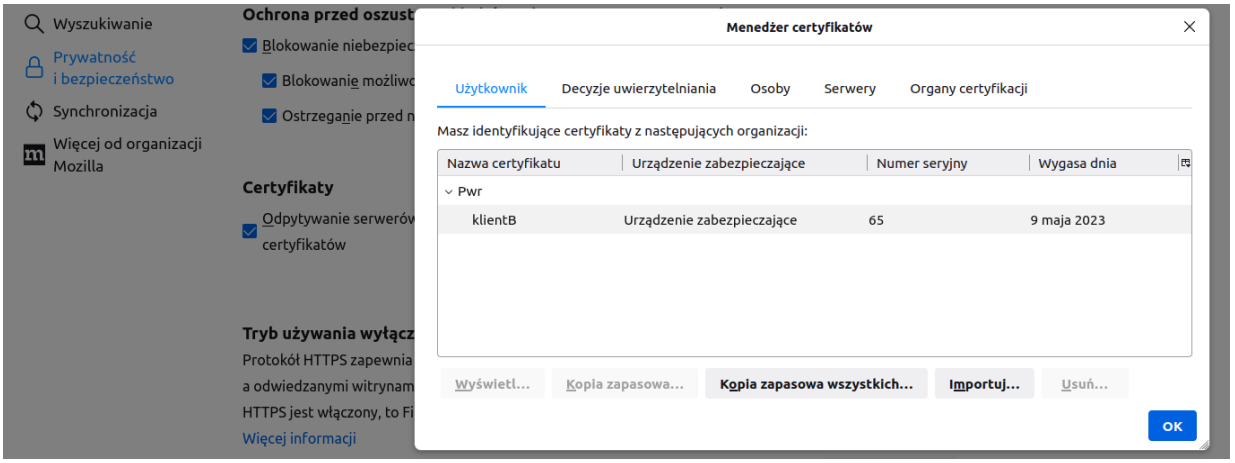
Sprawdzamy ścieżkę dla A i B przy poświadczeniu klienta A:



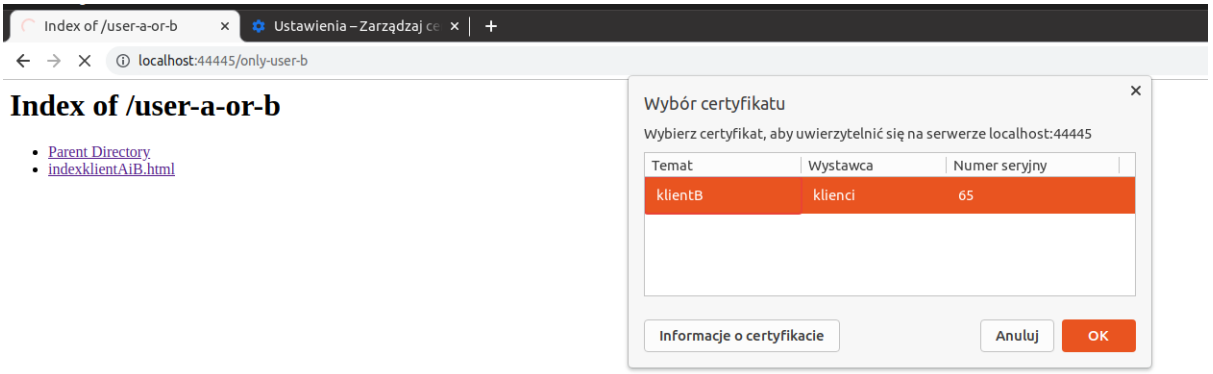
Również klient A posiada dostęp do ścieżki dla dwóch klientów. Wszystko jest tak jak powinno być dla klienta A.

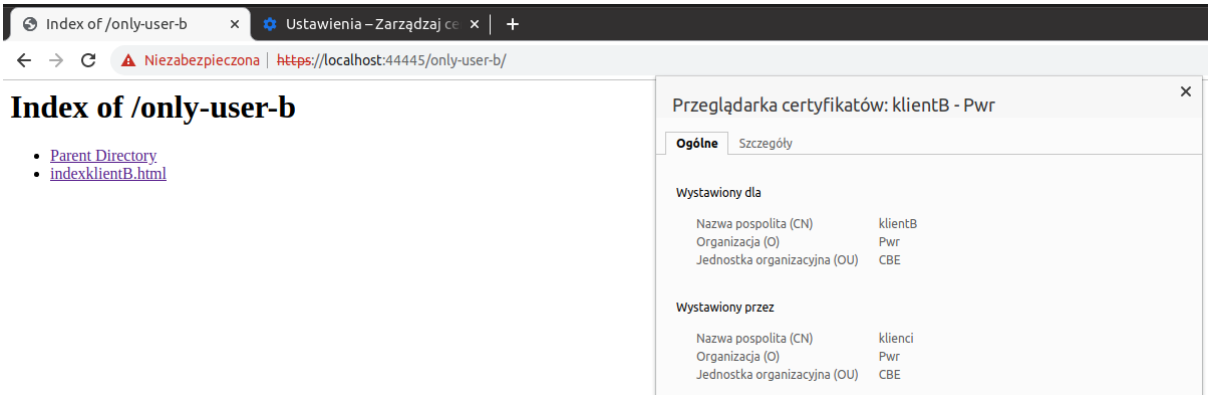
Klient B

Sprawdzamy klienta B:

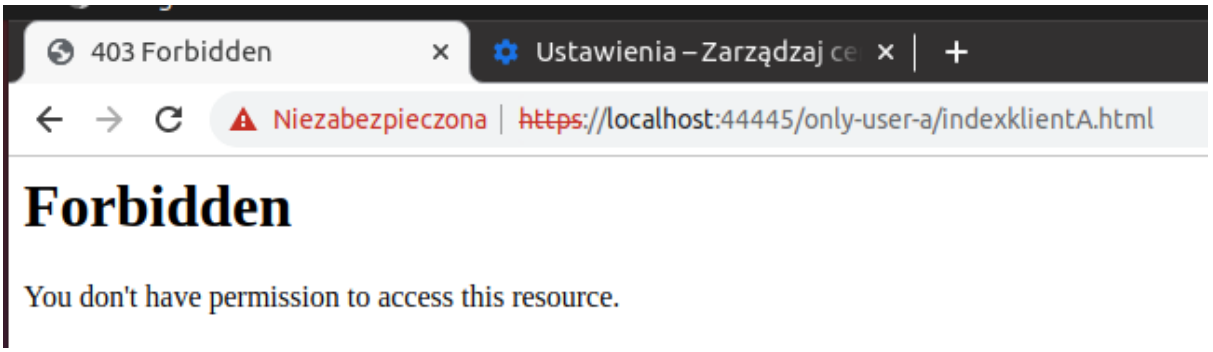


Również klient B ma dostęp do ścieżki dla każdego z użytkowników.





Jak widać gdy mamy certyfikat klienta B, to nie możemy odwiedzić ścieżki i pod ścieżki dla klienta A.



Podsumowanie

Konfiguracja Apache była prostsza od nginx, ale zajęło to trochę czasu. Rozwinięto parametry i reguły plik httpd-ssl.conf. Utworzono 3 dodatkowe ścieżki i nowe certyfikaty dla klientów. Wykonano wiele prób i błędów, aby klienci A i B mogli odwiedzać swoje ścieżki.