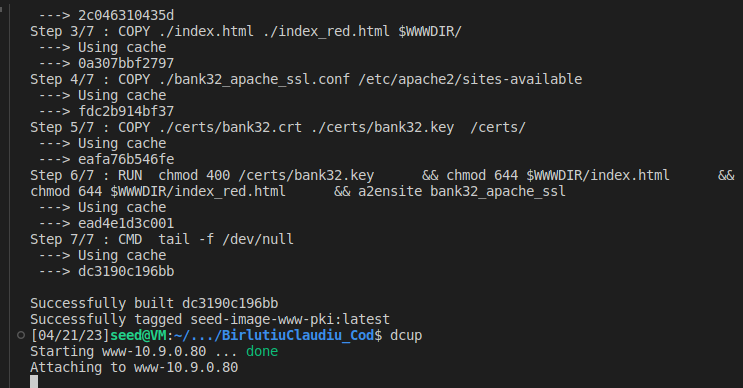
# Raport pentru lucrarea 6: Infrastructura de chei publice (PKI)

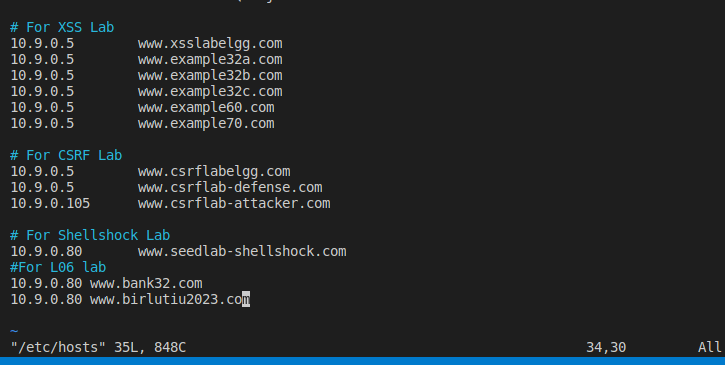
Autor: Birlutiu Claudiu-Andrei

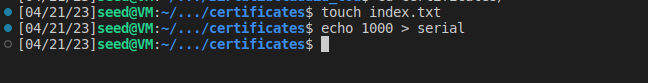
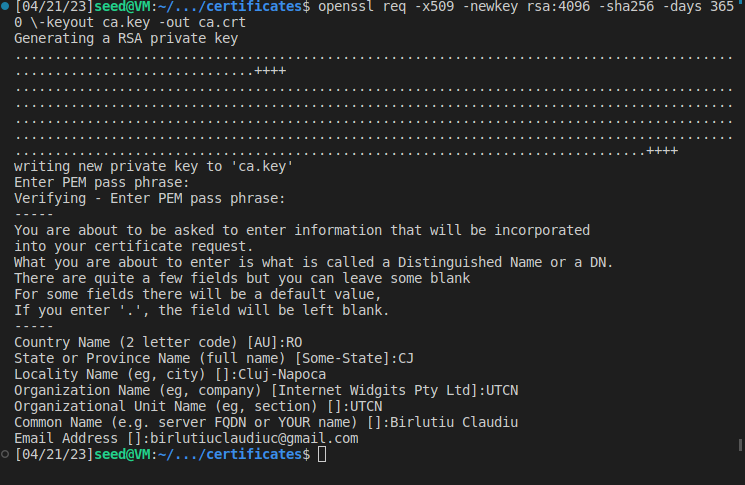
## Sarcina 1: Deveniți o autoritate de certificat (CA)

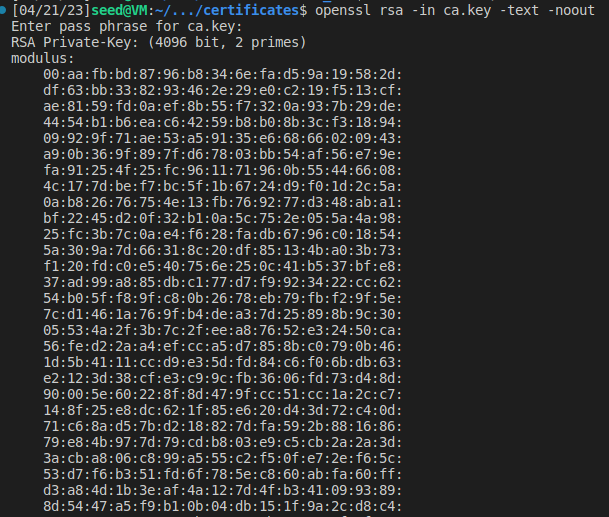
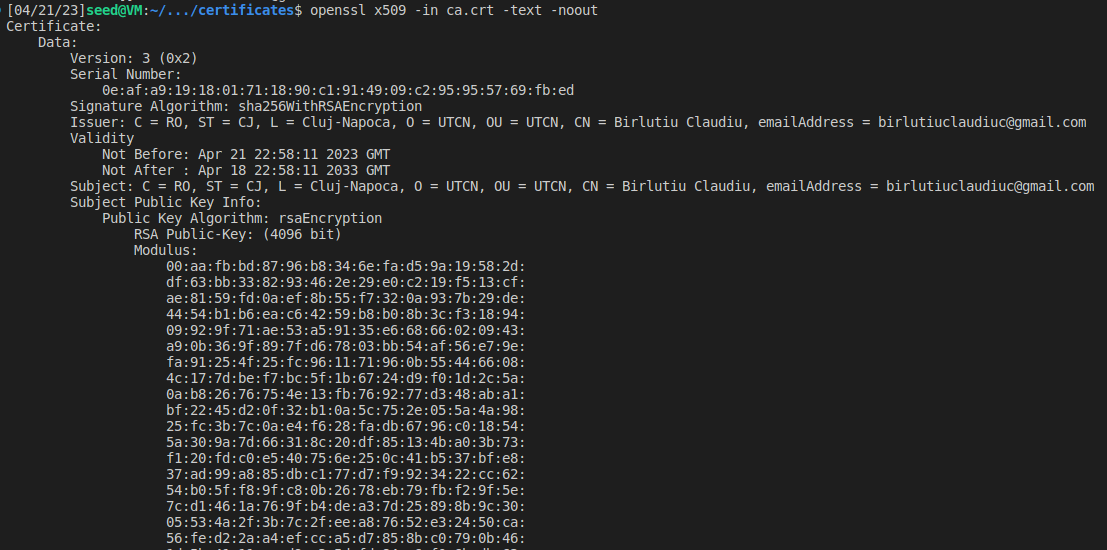
## În prima faza am creat containerul și am pornit acest serviciu



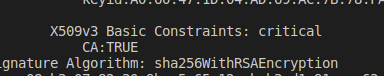
* Am adaugat în /etc/hosts cele 2 intrari



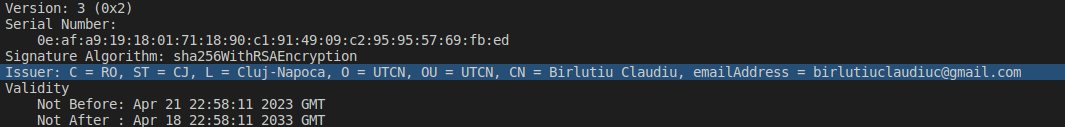
* am creat cele 2 fisiere mentionate
* am generat certificatul X.509 auto-semnat astfel:
* generăm o pereche de chei RSA cu o lungime de 4096 biți prin optiunea -newkey rsa:4096
* -sha256: specifică faptul că dorim să folosim algoritmul de hash SHA-256 pentru semnarea certificatului.
* Certificatul va fi valid 3650 de zile (10 ani aprox)
* am vizualizat continutul decodat al fisierelor



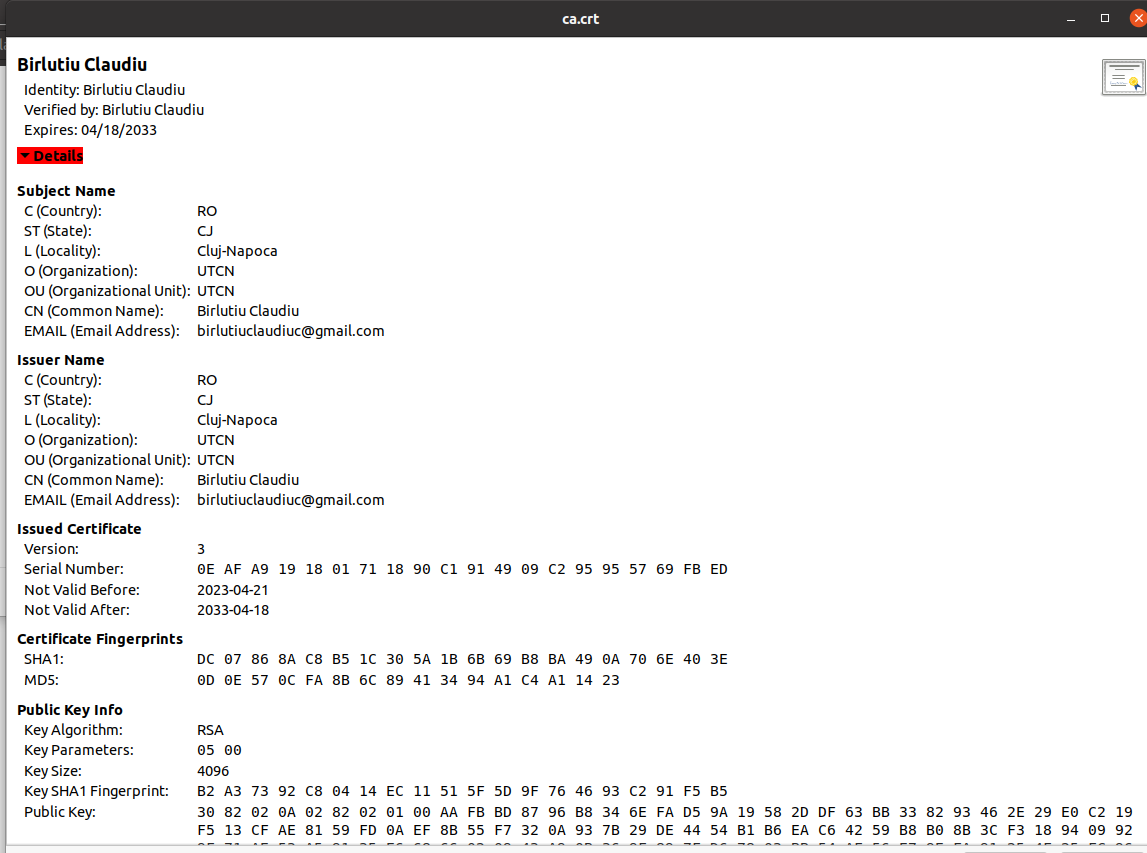
* Ce parte a certificatului indica ca acesta este un certificat CA?
  + există o extensie numită "Basic Constraints" care indică dacă certificatul este un certificat de autoritate de certificare (CA) sau nu.



* Ce parte a certificatului indica ca acesta este un certificat semnat de sine?
  + există un câmp numit "Issuer" care indică entitatea care a emis certificatul.

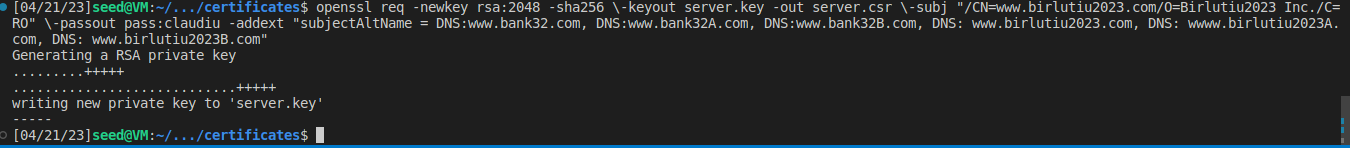


* In algoritmul RSA, avem un exponent public **e,** un exponent privat **d** , un modul **n** si doua secrete, numerele **p** și **q,** astfel ıncat **n=pq.** Va rugam sa identificati valorile pentru aceste elemente in certificatul dvs. si fisierele cheie
  + Exponent: 65537 (0x10001)
  + publicExponent: 65537 (0x10001)
  + celelalte valori se pot observa din : **openssl rsa -in ca.key -text -noout**
* certificatul

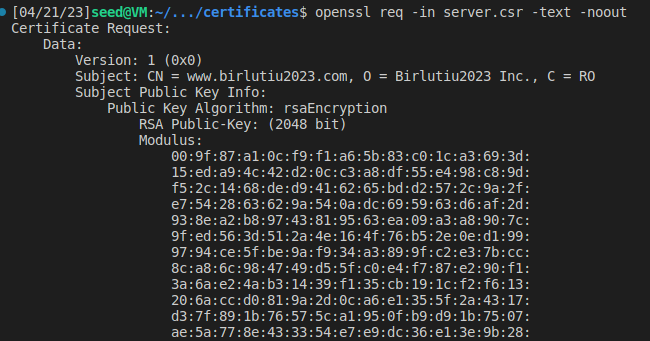


## Sarcina 2: Generarea unei cereri de certificat pentru serverul dvs.

* Am generat o cerere de certificat pentru serverul nostru folsoind comanda:
  + openssl req -newkey rsa:2048 -sha256 \-keyout server.key -out server.csr \-subj "/CN=www.birlutiu2023.com/O=Birlutiu2023 Inc./C=RO" \-passout pass:claudiu -addext "subjectAltName = DNS:www.bank32.com, DNS:www.bank32A.com, DNS:www.bank32B.com, DNS: www.birlutiu2023.com, DNS: wwww.birlutiu2023A.com, DNS: www.birlutiu2023B.com"

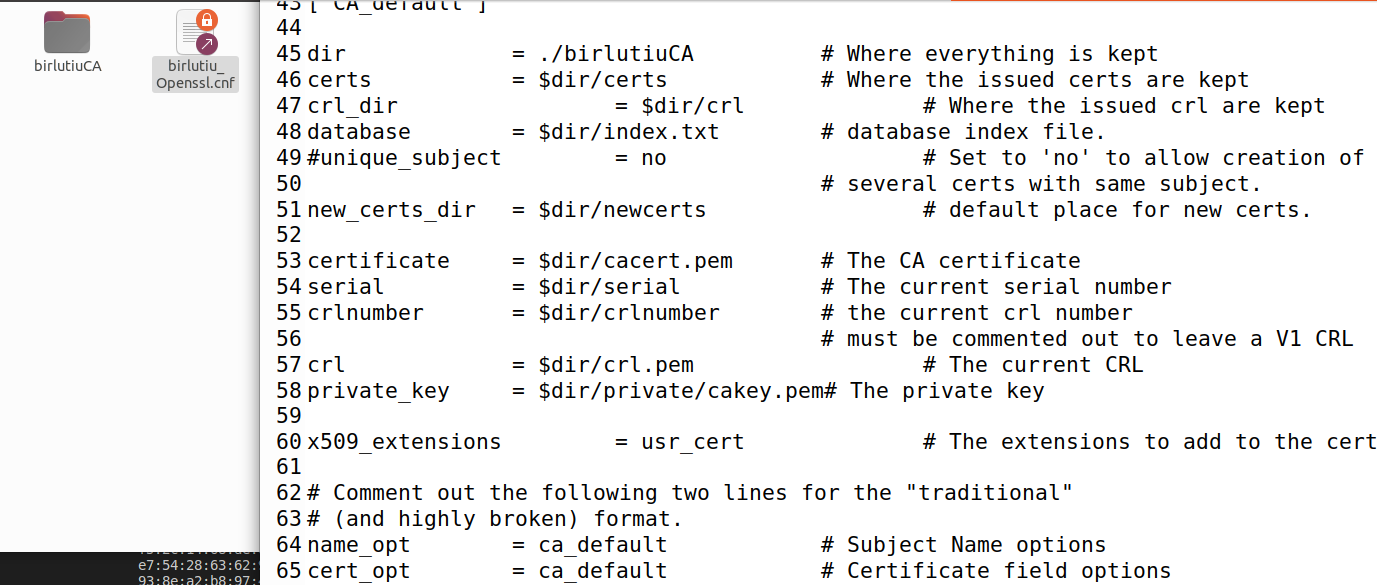


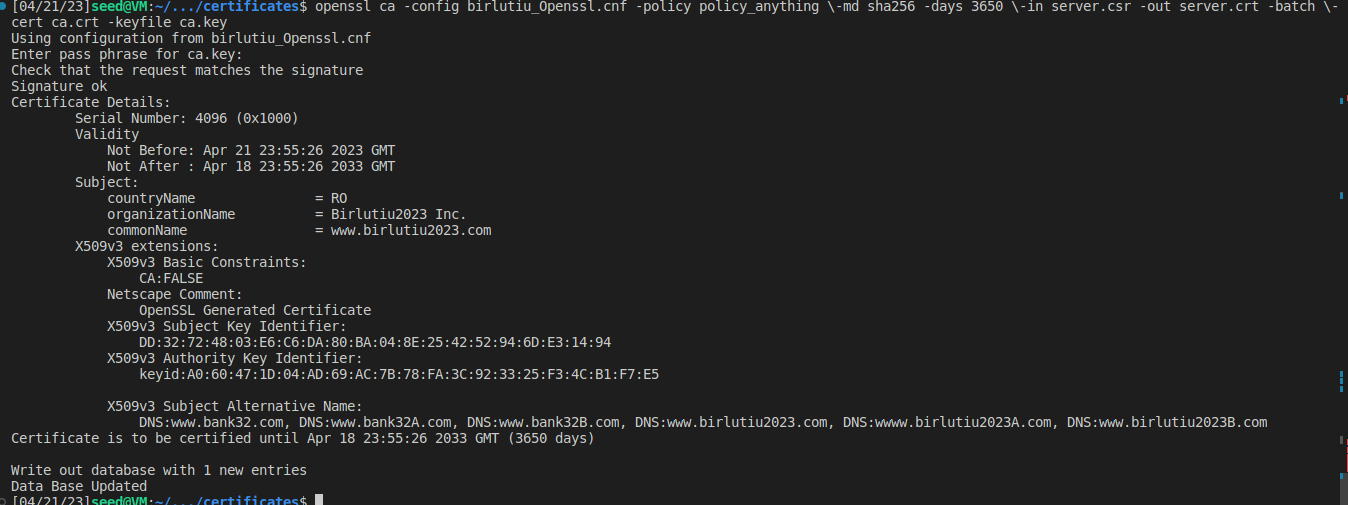
* am urmărit continutul decoda al fisierului .csr

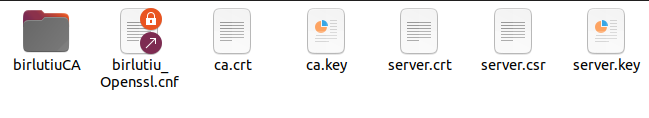


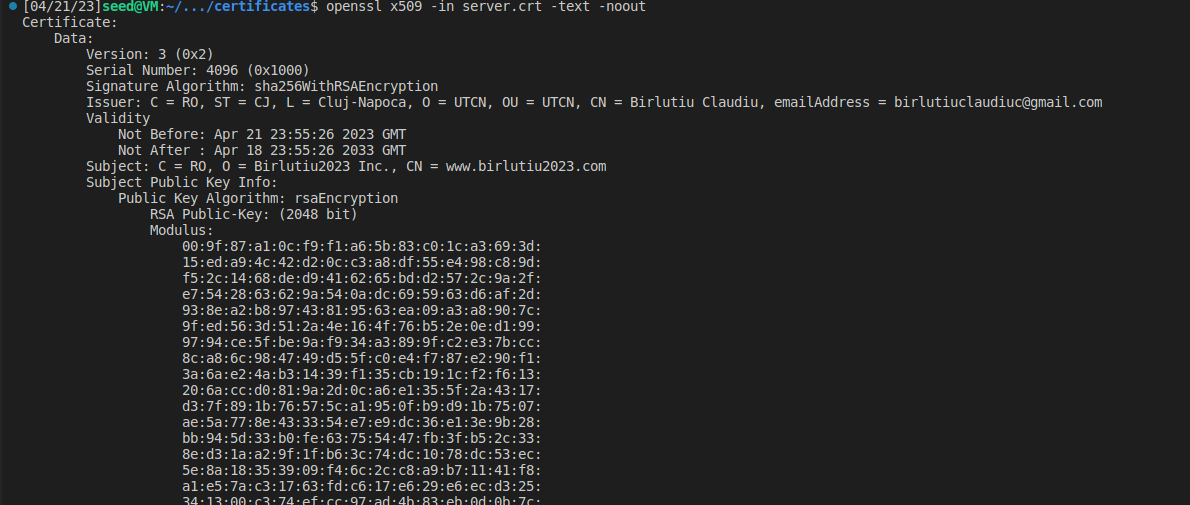
## Sarcina 3: Generarea unui certificat pentru serverul dvs.

* Fișierul **server.csr** trebuie sa aibă **semnatura CA** pentru a forma un certificat.
* Am copiat fișierul openssl.cnf într-un folder din BirlutiuClaudiu\_Cod lângă directorul **birlutiuCA** unde se afla toate certificatele pe care le-am generat



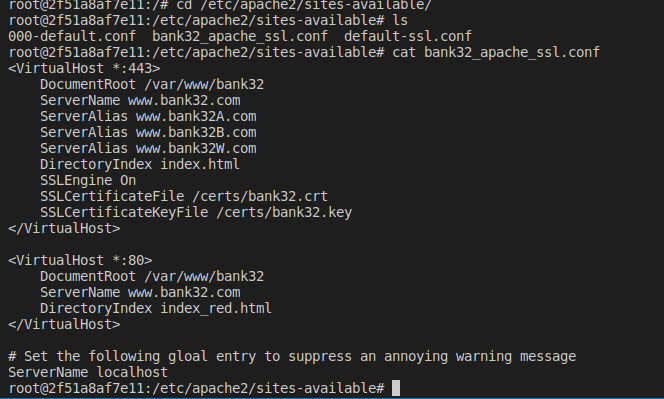
* Astfel, pentru sermnarea certificatului **server.csr** într-un certificat x509 (**server.crt**), folosind **CA.crt** și **ca.key** vom rula următoarea comanda:
  + - openssl ca -config **birlutiu\_Openssl.cnf** -policy **policy\_anything** \-md sha256 -days 3650 \-in **server.csr** -out **server.crt** -batch \-cert **ca.crt** -keyfile **ca.key**
* rezultatul obținut



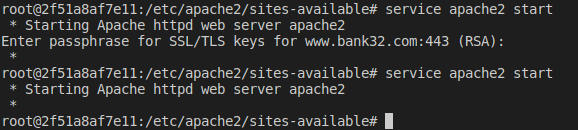


## Sarcina 4: Plasarea unui certificat într-un sit de web HTTPS bazat pe Apache

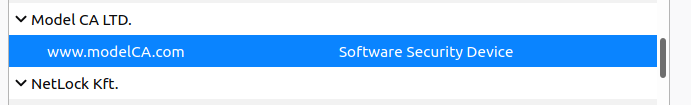
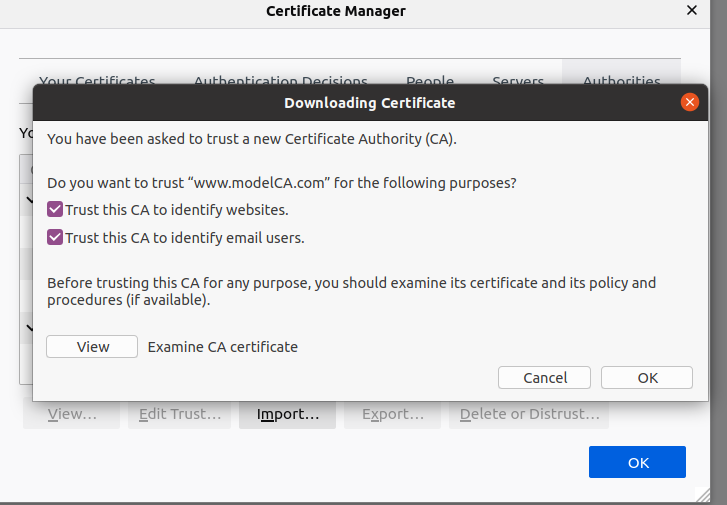
* Vizualizare în cobatianrul nousru a fisierului **bank32\_apache\_ssl.conf**

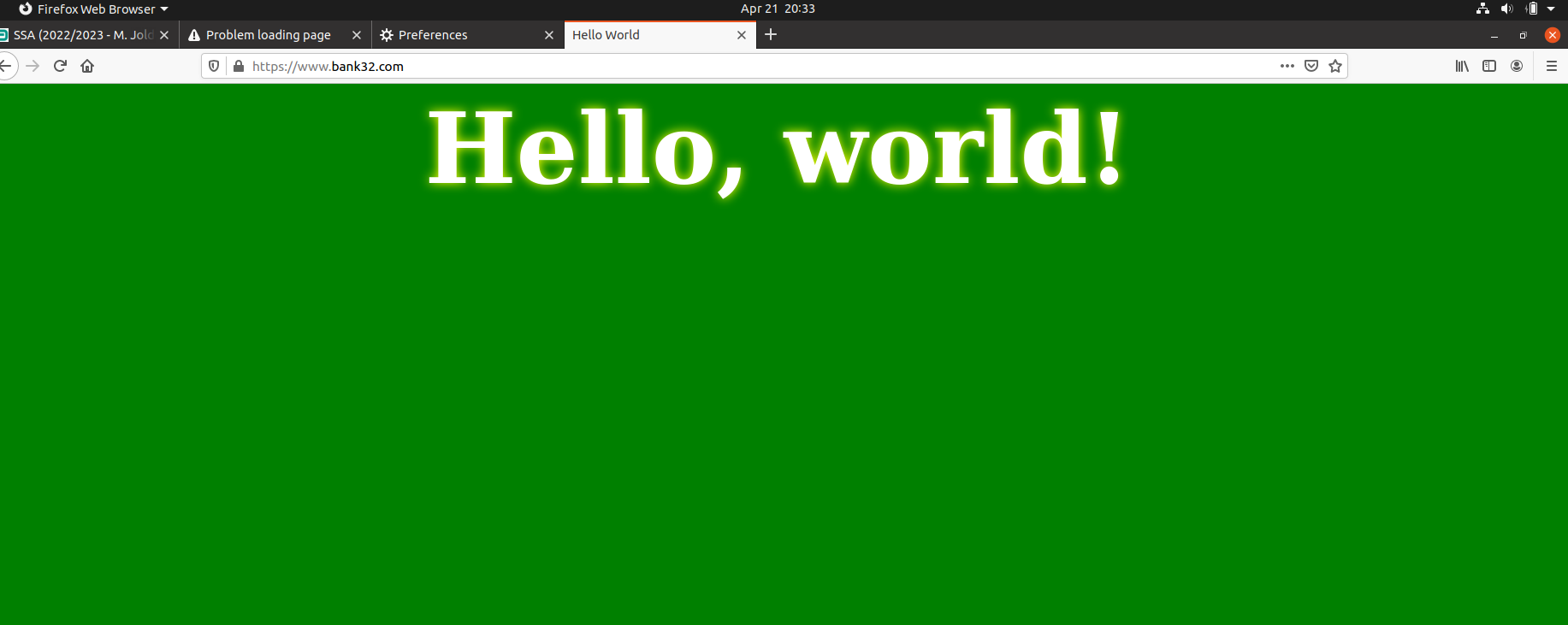


* DocumentRoot – unde sunt stocate fisierele pentru site-ul web
* **activam apache2** din container:



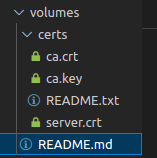
* la o prima încercare a accesarii din browser a site-ului [www.bank.com](http://www.bank.com/) observam ca nu se randeaza niciun site deaorece nu avem încărcata autoritatea pentru validarea certif pentru acest site; vom încarca în FireFox certificatul **modelCA.crt** din folderul **certs;** aceasta e autoritatea care va certifica site-ul bank32 cu certificatul bank32.crt



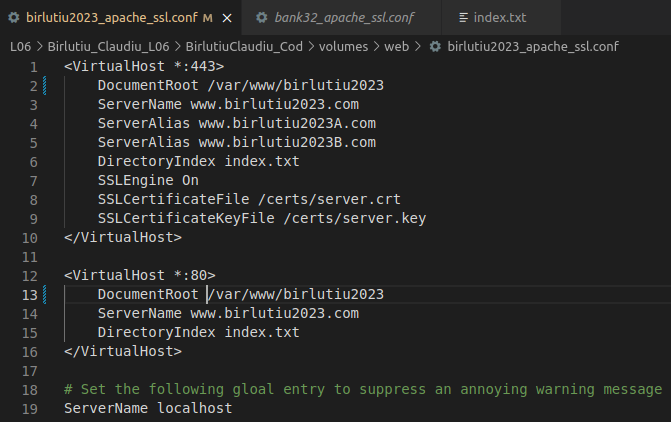


În continuare ne vom ocupa de configurarea propriului nostru site pentru care am obținut certificatul.

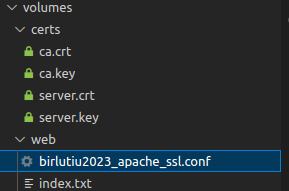
* pentru început vom crea în volumes un fisier denumiit certs unde se vor afla fiserele **ca.crt, ca.key, server.crt si server key** fisiere pe care le-am obținut anterior în



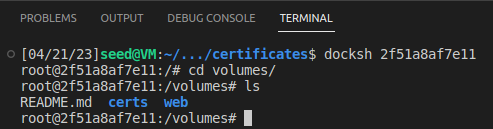
* cream un folder nou în volumes numit web și adaugam fișierul **index.txt**
* cream fișierul de configurare pentru apache2



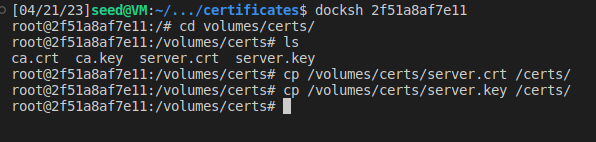
* structura fisierului Volumes este aceasta:



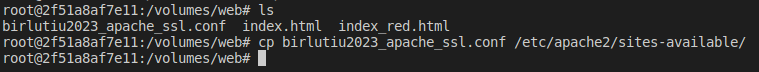
* repornim containerul și observam ca s-au copiat cele 2 foldere



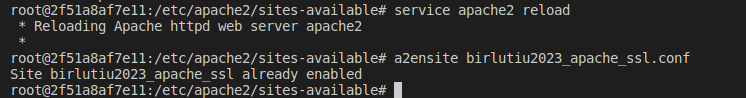
* în continuare vom copia în **certs**  *server.crt* și *server.key* din /volumes/certs



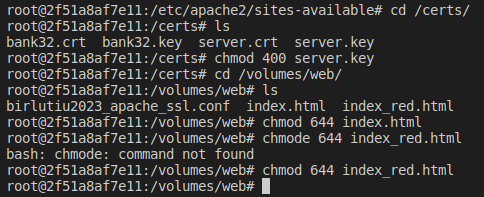
* vom copia și **birlutiu2023\_apache\_ssl.conf** în /etc/apache2/sites-available



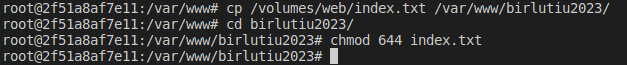
* activam acest site



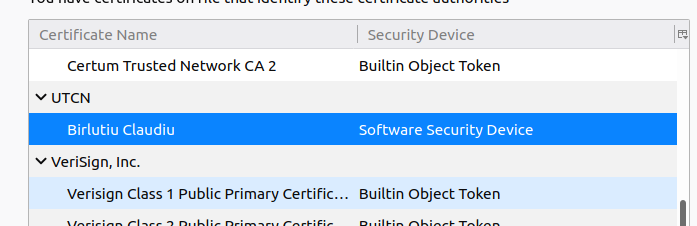
* schimbam tipurile de acces pentru fiesiere în felul următor:



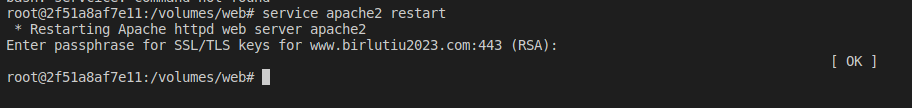
* copiere index.txt în ***var/*www/birlutiu2023** și adaugare chmod **644**



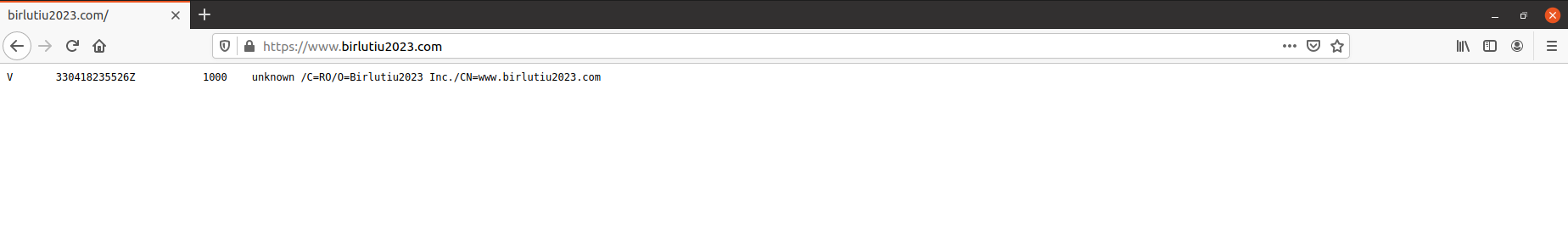
* încarcam certificatul **ca.crt** în Firefox



* pornim serviciul apache2

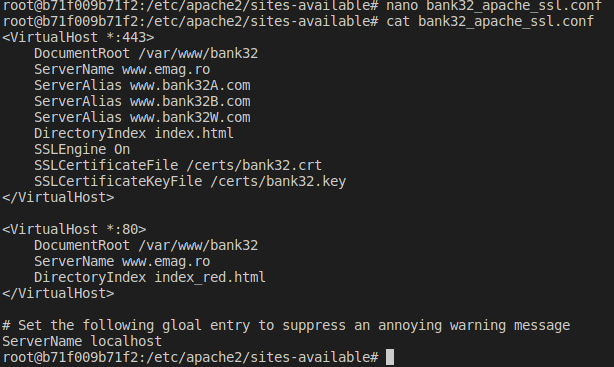


REZULATUL OBTINUT!!!!



## Sarcina 5: Lansarea unui atac de tipul om-la-mijloc

* În continure vom lansa un atac de tipul MITM având ca site luat ca studiu fiind <https://www.emag.ro/> , un site de cumpărături cunoscut
* m-am cobectat la container și am modificat server name de la bank32 conf apache i [www.emag.](http://www.emag.com/)ro



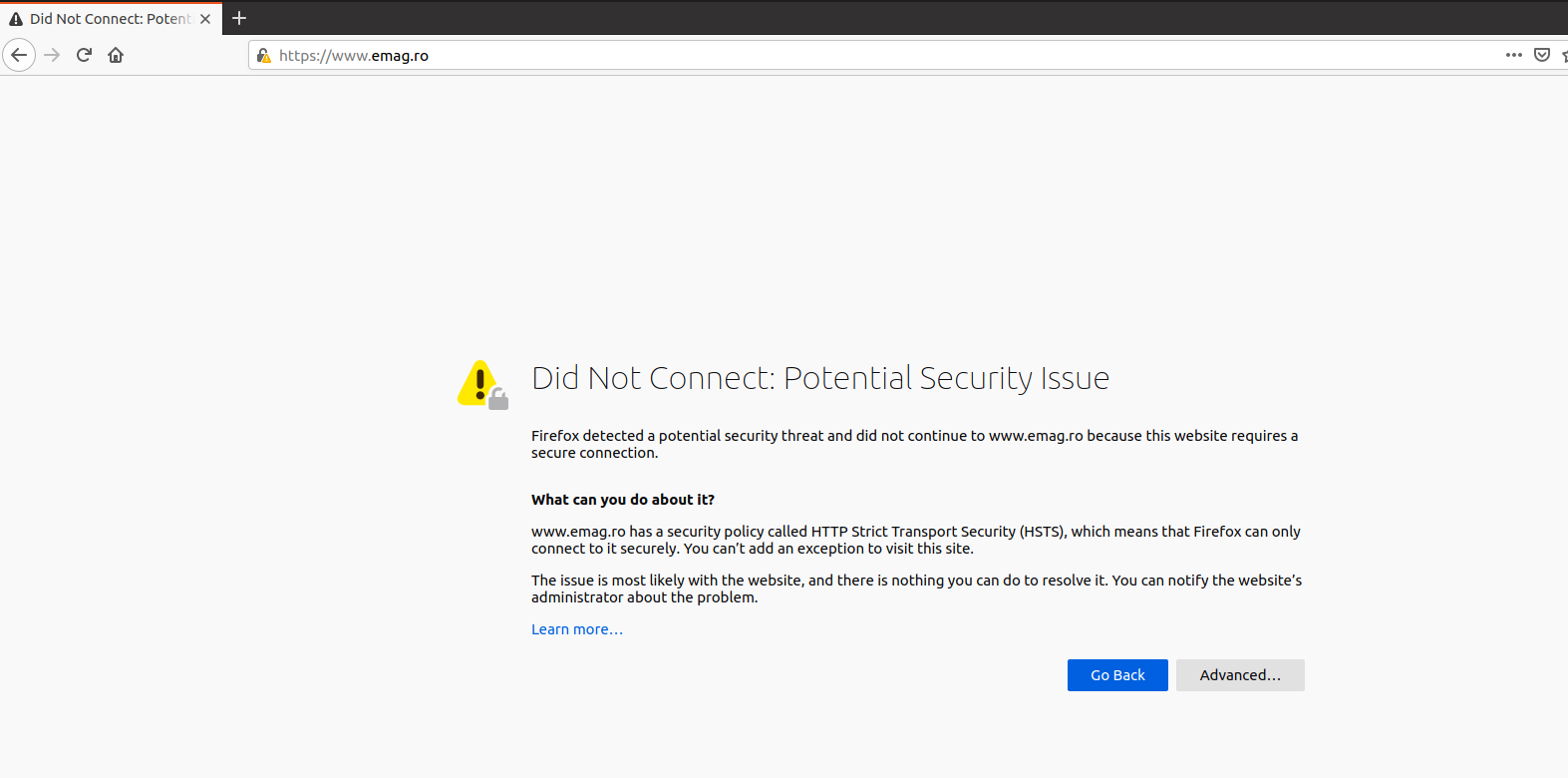
* dam un restart serverul de apache2



* adaugam în etc/hosts al mașinii virtuale intrearea 10.9.0.80 [www.emag.ro](http://www.emag.ro/)pentru a simula atacarea DNS a site-ului emag.ro



* obținem următoarea eroare când accesam site-ul www.emag.ro



* site-ul **emag.ro ut**ilizeaza polita de securitate HTTP Strict Transport Security (HSTS) ce impune ca browserul să se conecteze doar în mod securizat la site-ul web, utilizând protocolul HTTPS.
* chiar dacă am adăuga o intrare în fișierul "etc/hosts" pentru a redirecționa traficul către site-ul web, browserul continua să aplice politica HSTS și blocheaza accesul la site prin HTTP.
* atunci când un site web utilizează HSTS, browserul, în cazul nostru Firefox, va fi instruit să se conecteze numai prin HTTPS la acel site pentru o anumita perioada de timp

## Sarcina 7: Lansarea unui atac de tipul om-la-mijloc cu o CA compromisă