

Mini-projet: Fortinet

Alexandre Em

May 22, 2022

1 Introduction

Pour ce mini projet nous implementerons cette topologie qui contiendra une machine **appsecres** servant une application web php avec apache2. Cette application utilisera des données d'une db SQL se situant sur une autre machine **dbsecres**. On ajoutera aussi une machine **certsecres** qui permettra de générer des certificats ssl, et servira donc a générer un certificat pour l'application web pour ensuite être disponible par les utilisateurs en http et en https.

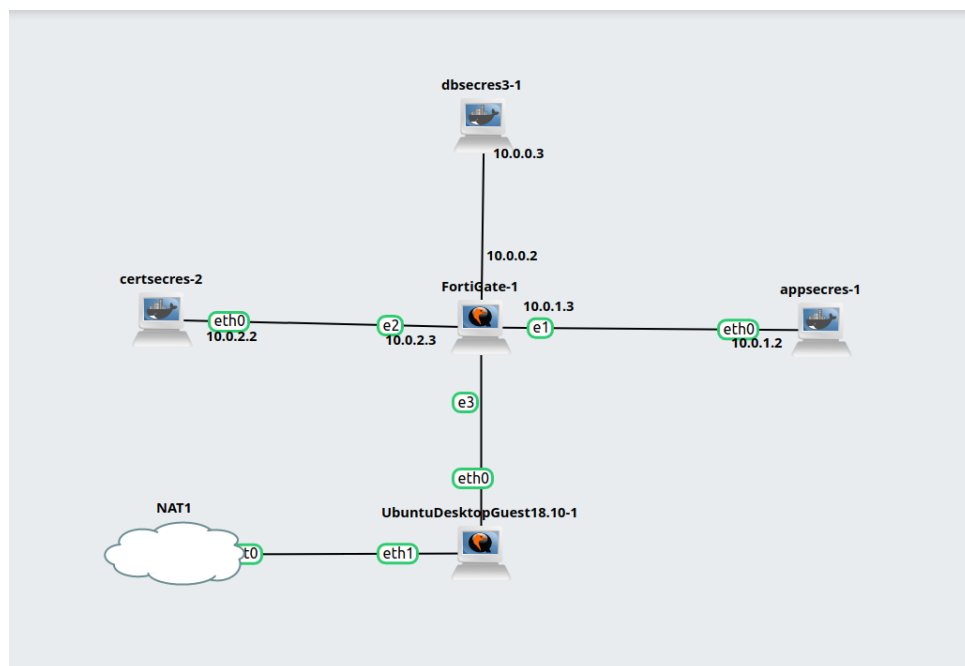


Figure 1: Topology

2 Machines Docker

Nous avons utilisé ici Docker pour simuler les différentes machines (hors machine cliente qui requière une interface graphique) grâce à des images debian, ou nous avons installer les packages nécessaire a chaque machine.

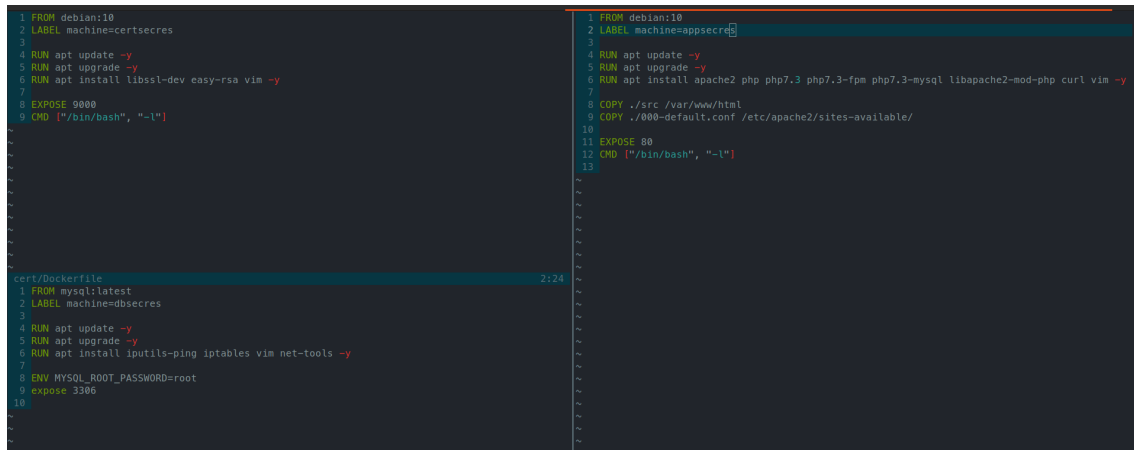


Figure 2: Server's Dockerfile content

Pour chaque image, nous avons ensuite exécuter les commandes suivantes, qui permettront de build, de lancer l'image si on a besoin de lancer dans un container et vérifier que la machine fonctionne avant de l'intégrer dans l'archi et sauvegarder les changements au cas où on en aurait effectué lors de l'exécution.

```
docker build -t <image_name> <Dockerfile_path>
# Optionnal
docker run -it <image_name> /bin/bash
docker ps -a
docker commit <container_id> <image_tag>
```

Une fois toutes les machines build et fonctionnelles, on peut les ajouter sur GNS3 depuis le menu Preference → Docker → Add.

3 Machine Database SQL

3.1 Network configuration

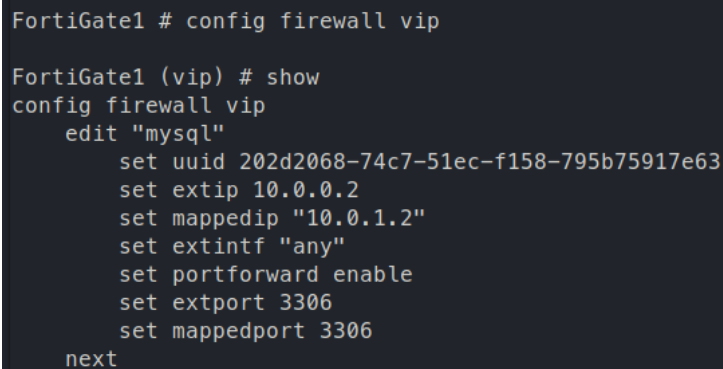
Nous avons connecté sur GNS3 la machine `dbsecres` à Fortinet aux ports Ethernet. Nous allons donc configurer leurs interfaces afin qu'elles soient connectées avec les bonnes autorisations. Pour configurer la machine linux, on effectue un clic droit sur la machine depuis GNS3 qui correspond au fichier `/etc/network/interfaces`

```
# Static config for eth0
auto eth0
iface eth0 inet static
    address 10.0.0.4
    netmask 255.255.255.0
    gateway 10.0.0.2
    up echo nameserver 10.0.0.2 > /etc/resolv.conf
```

Puis on configure l'interface du coté de FortiGate

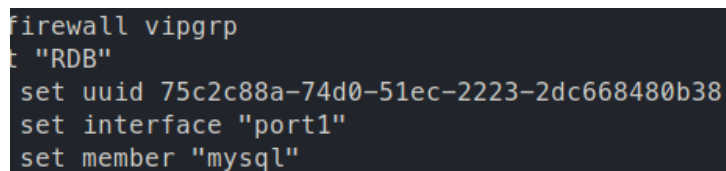
```
config system interface
edit port1
set mode static
set ip 10.0.0.2 255.255.255.0
set allowaccess ping ssh
set role dmz
end
```

Pour pouvoir avoir accès à la MySQL qui sert depuis le port 3306, on doit donc faire un port forward. Pour cela on va créer et configurer des Virtual IP



```
FortiGate1 # config firewall vip
FortiGate1 (vip) # show
config firewall vip
edit "mysql"
set uuid 202d2068-74c7-51ec-f158-795b75917e63
set extip 10.0.0.2
set mappedip "10.0.1.2"
set extintf "any"
set portforward enable
set extport 3306
set mappedport 3306
next
```

Figure 3: Virtual IP configuration for dbsecres machine



```
firewall vipgrp
t "RDB"
set uuid 75c2c88a-74d0-51ec-2223-2dc668480b38
set interface "port1"
set member "mysql"
t
```

Figure 4: Adding VIP on a VIP group related to dbsecres interface

Puis on ajoute aussi une policy pour autoriser les communications entre appsecres et dbsecres. Puis on ajoute une autre policy "retour" donc permettant les communications entre dbsecres et appsecres.

```

config firewall policy
  edit 1
    set name "mysqlpolicy"
    set uuid 0badc4d0-74d1-51ec-1a44-2f7b60f5dcaa
    set srcintf "port1"
    set dstintf "port2"
    set srcaddr "all"
    set dstaddr "RDB"
    set action accept
    set schedule "always"
    set service "ALL"
  next
end

```

Figure 5: Add a policy for dbsecres → appsecres

```

edit 2
  set name "phpapp"
  set uuid 696ff95a-74de-51ec-c2db-587b300e1288
  set srcintf "port2"
  set dstintf "port1"
  set srcaddr "all"
  set dstaddr "all"
  set action accept
  set schedule "always"
  set service "ALL"
next

```

Figure 6: Add a policy for appsecres → dbsecres

3.2 Add User with permission

Pour pouvoir se connecter à MySQL, il faut ajouter un utilisateur qui pourra se connecter depuis la machine contenant l'application et donner les permissions nécessaire pour accéder à la base de donnée. On peut assigner l'adresse ip de **secresapp** mais ici par soucis de simplicité, on autorisera la connexion depuis toutes les machines.

```

CREATE USER 'alexandre'@'%' IDENTIFIED BY 'password';
GRANT ALL ON employeesDB.* TO 'alexandre'@'%' ;

```

4 Machine Apache

Dans le Dockerfile, on déplace le code de l'application Php dans le container au chemin `/var/www/html` et grâce au plugin `php7.3` et `php-mysql` on va servir l'application au port 80 et 443, qui sont configurer dans le fichier `/etc/apache2/sites-available/000-default.conf`.

```

<FilesMatch ".php$"
  SetHandler "proxy:unix:/var/run/php/php7.3-fpm.sock|fcgi://localhost/"
</FilesMatch>

```

Figure 7: Add all php files run with php fpm and serve the app on localhost:80

```
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

<VirtualHost *:80 *:443>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    ServerAdmin webmaster@localhost

    SSLEngine On
    SSLCertificateFile /root/csr/10.0.1.2.crt
    SSLCertificateKeyFile /root/csr/10.0.1.2.key
    SSLCertificateChainFile /root/csr/ca.crt

    DocumentRoot /var/www/html

    <FilesMatch ".php$">
        SetHandler "proxy:unix:/var/run/php/php7.3-fpm.sock|fcgi://localhost/"
    </FilesMatch>

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
# /etc/apache2/sites-available/000-default.conf
"/etc/apache2/sites-available/000-default.conf" 73L, 3013C written
```

Figure 8: Add the app and its public and private certificates and serve the app on localhost:80/443

Puis executez cette commande:

```
a2enmod ssl
```

4.1 Network configuration

```
# Static config for eth0
auto eth0
iface eth0 inet static
    address 10.0.1.2
    netmask 255.255.255.0
    gateway 10.0.1.3
    up echo nameserver 10.0.1.3 > /etc/resolv.conf
```

Puis on configure l'interface du coté de FortiGate

```
config system interface
edit port2
set mode static
set ip 10.0.1.3 255.255.255.0
set allowaccess ping ssh
set role dmz
end
```

5 Certificate Authority

De même l'autorité de certificat sera installé sur une autre machine Debian. On a donc choisi easy-rsa.

5.1 Initialize easy-rsa

```
apt install libssl-dev easy-rsa openssh-server -y
cd ~
mkdir easy-rsa
ln -s /usr/share/easy-rsa/* ~/easy-rsa/ #Create a symbolic link on root
chmod 700 ~/easy-rsa
cd easy-rsa
./easy-rsa init-pki
```

5.2 Creation of our CA

Ecrire et remplir sur le fichier `/root/easy-rsa/vars`

```
set_var EASYRSA_REQ_COUNTRY "...to fill..."
set_var EASYRSA_REQ_PROVINCE "...to fill..."
set_var EASYRSA_REQ_CITY "...to fill..."
set_var EASYRSA_REQ_ORG "...to fill..."
set_var EASYRSA_REQ_EMAIL "...to fill..."
set_var EASYRSA_REQ_OU "...to fill..."
set_var EASYRSA_ALGO "ec"
set_var EASYRSA_DIGEST "sha512"
```

```
./easyrsa build-ca # Generate public & private CA keys
# outputs: ~/easy-rsa/pki/ca.crt and ~/easy-rsa/pki/private/ca.key
```

On obtiendra après avoir entré le mot de passe un certificat public (`ca.crt`) et prive (`ca.key`) de l'autorité de certificat. Sachant que l'on va uniquement distribuer le certificat public aux machines du réseau.

5.3 Distribute our CA's public certificate

Une fois le certificat public de notre autorité de certificat généré, nous allons le transférer dans nos différentes machines donc appsecres, que l'on utilisera pour générer les certificats public et prive de notre application, et de la machine cliente que l'on ajoutera dans la liste des autorités reconnu du navigateur.

5.4 SSH configuration

Pour effectuer les transferts du certificats public par ssh, il faut d'abord spécifier le port et autoriser le root a s'identifier.

```
# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
```

Figure 9: Specify port 22 and authorize to log on root account because there is no account except root on Docker debian image

5.5 Transfert certificate to Webapp server

```
scp pki/ca.crt root@10.0.1.2:/tmp
```

5.6 Generate Web app certificate

Depuis la machine contenant l'application web, on installe tout d'abord openssl pour ensuite generer une cle privée. Dans les screens ci dessous, il faut remplacer 10.0.1.2 par le nom de domaine souhaité donc ici secrese.m (je m'étais trompé sur le nom et les screens avec le nom de domaine correct n'ont pas été sauvegardé donc je mets les ancien)


```

certsec-1:~/.ssh# openssl genrsa -out 10.0.1.2.key
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
e is 65537 (0x010001)
root@certsec-1:~/.ssh# openssl req -new -key 10.0.1.2.key -out 10.0.1.2.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) []:Paris
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Monk ai
Organizational Unit Name (eg, section) []:Organization
Common Name (e.g. server FQDN or YOUR name) []:10.0.1.2
Email Address []:alexandre.em@pm.me

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:alexandre
An optional company name []:monk
root@certsec-1:~/.ssh# scp 10.0.1.2.req root@10.0.5.2:/tmp/
root@10.0.5.2's password:
10.0.1.2.req
100% 1131 1.0MB/s 00:00
root@certsec-1:~/.ssh#

```

Figure 10: Private key and req file generation

Ce fichier .req va nous permettre de faire une requete au CA afin qu'il puisse nous générer par la suite une cle public que l'on couplera avec la cle privé généré précédement. Pour cela on transfere ce fichier .req a la machine certsecres par ssh et on switch donc sur la machine certsecres.

Une fois le certificat .crt généré il ne reste plus qu'à le transférer.

```

root@certsecres-2:~/easy-rsa# ./easyrsa import-req /tmp/10.0.1.2.req 10.0.1.2
Note: using Easy-RSA configuration from: ./vars
Using SSL: openssl OpenSSL 1.1.1d 10 Sep 2019
The request has been successfully imported with a short name of: 10.0.1.2
You may now use this name to perform signing operations on this request.

root@certsecres-2:~/easy-rsa# ./easyrsa sign-req server 10.0.1.2
Note: using Easy-RSA configuration from: ./vars
Using SSL: openssl OpenSSL 1.1.1d 10 Sep 2019

You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 1080 days:

subject=
  countryName           = FR
  stateOrProvinceName    = Ile de France
  localityName           = Paris
  organizationName       = Monk ai
  organizationalUnitName = Organization
  commonName             = 10.0.1.2
  emailAddress           = alexandre.em@pm.me

Type the word 'yes' to continue, or any other input to abort.
Confirm request details: yes
Using configuration from /root/easy-rsa/pki/safessl-easyrsa.cnf
Enter pass phrase for /root/easy-rsa/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName             :PRINTABLE:'FR'
stateOrProvinceName     :ASN.1 12:'Ile de France'
localityName            :ASN.1 12:'Paris'
organizationName        :ASN.1 12:'Monk ai'
organizationalUnitName  :ASN.1 12:'Organization'
commonName               :ASN.1 12:'10.0.1.2'

```

Figure 11: Public key generation

```
scp pki/issued/secres.em.crt root@10.0.1.2:/tmp
scp pki/ca.crt root@10.0.1.2:/tmp
cp /tmp/secres.em /root/csr
cp /tmp/ca.crt /root/csr
```

6 Fortigate

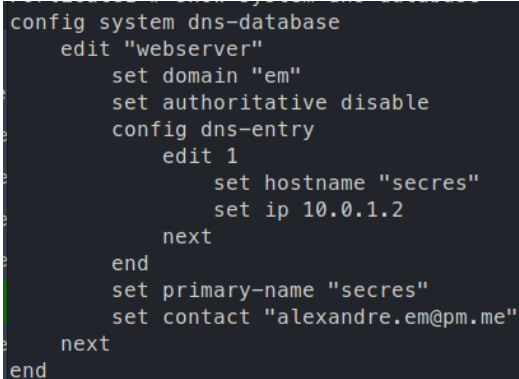
6.1 Sniffer for packets

Afin de surveiller le trafic d'une interface à une autre, on utilise la commande suivante:

```
diagnose sniffer packet any "tcp port 3306 or udp port 3306" 4 a
```

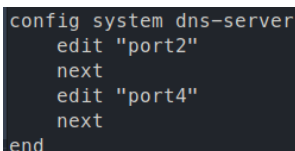
6.2 appsecres DNS

Une fois le certificat généré avec comme nom *secres.em*, l'accès vers l'application web doit s'effectuer vers ce domaine afin que le certificat soit reconnu par notre autorité de certificat. Pour cela nous allons définir un dns à partir de Fortigate lié à l'interface de l'application web.



```
config system dns-database
  edit "webserver"
    set domain "em"
    set authoritative disable
    config dns-entry
      edit 1
        set hostname "secres"
        set ip 10.0.1.2
      next
    end
    set primary-name "secres"
    set contact "alexandre.em@pm.me"
  next
end
```

Figure 12: Configuration du DNS depuis Fortigate



```
config system dns-server
  edit "port2"
  next
  edit "port4"
  next
end
```

Figure 13: Configuration du DHCP

```

config system dhcp server
  edit 1
    set dns-service default
    set default-gateway 10.0.1.2
    set netmask 255.255.255.0
    set interface "port4"
    config ip-range
      edit 1
        set start-ip 10.0.1.1
        set end-ip 10.0.1.254
      next
    end
  next
end

```

Figure 14: Public key generation

On vérifie ensuite que le ping avec le DNS qui vient d'être créer, depuis Fortigate et depuis notre machine User fonctionne.

```

FortiGate1 # execute ping secresem
PING secresem (10.0.1.2): 56 data bytes
64 bytes from 10.0.1.2: icmp_seq=0 ttl=64 time=2.8 ms
64 bytes from 10.0.1.2: icmp_seq=1 ttl=64 time=2.8 ms
64 bytes from 10.0.1.2: icmp_seq=2 ttl=64 time=1.8 ms

```

Figure 15: Ping secresem depuis Fortigate

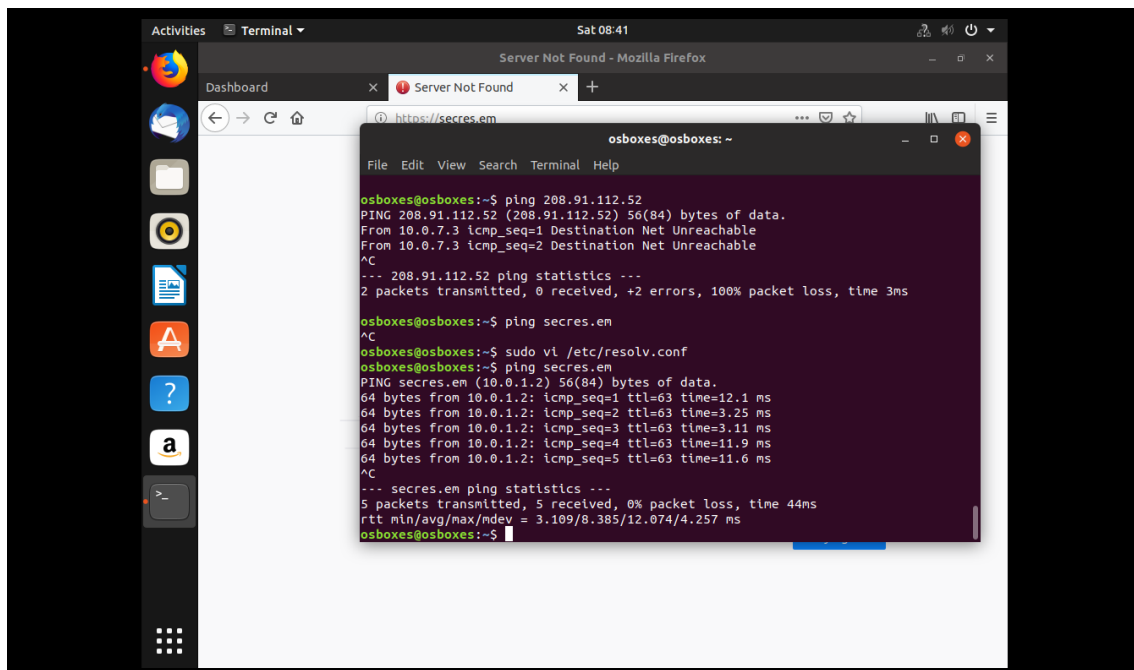


Figure 16: Ping secresem depuis la machine User

6.3 User machine

Depuis la machine client, après avoir configuré les communications http et https depuis Fortigate des machines appsecres et user, on peut accéder depuis le navigateur à l'application depuis l'adresse 10.0.1.2,

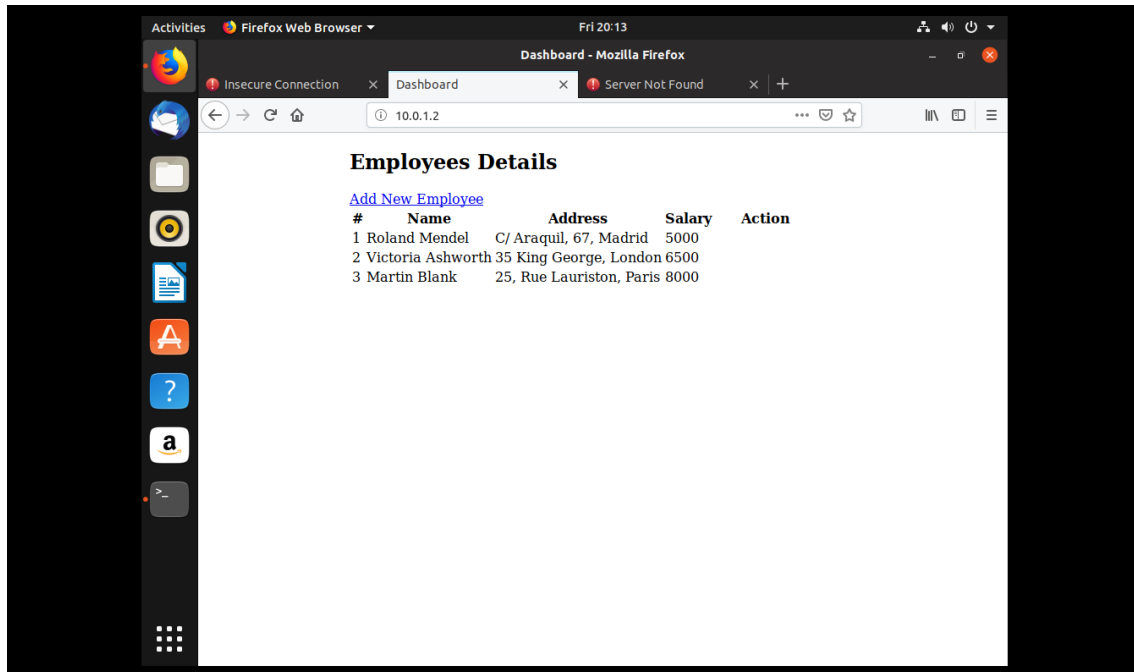


Figure 17: Accès à l'application web en http avec l'adresse ip du serveur web

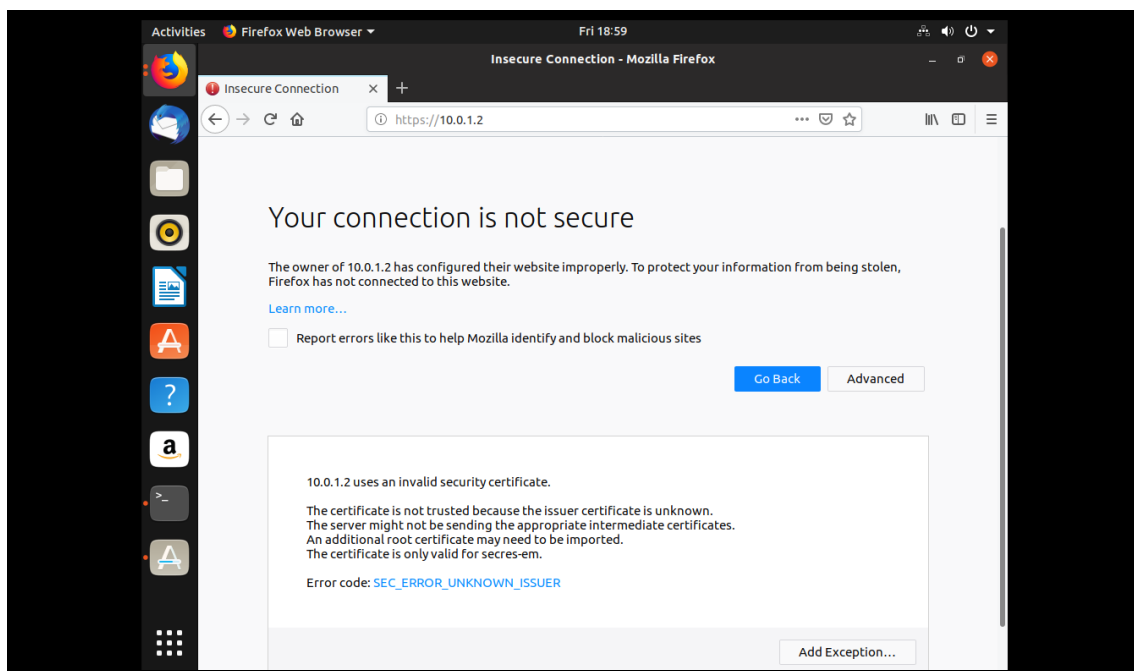


Figure 18: Accès à l'application web en https avec l'adresse ip du serveur web

mais lorsque l'on essaie d'y accéder en https l'autorité de certificat n'est pas reconnu. Il faut donc que l'on ajoute la cle public de notre CA a la liste des CA autorisés.

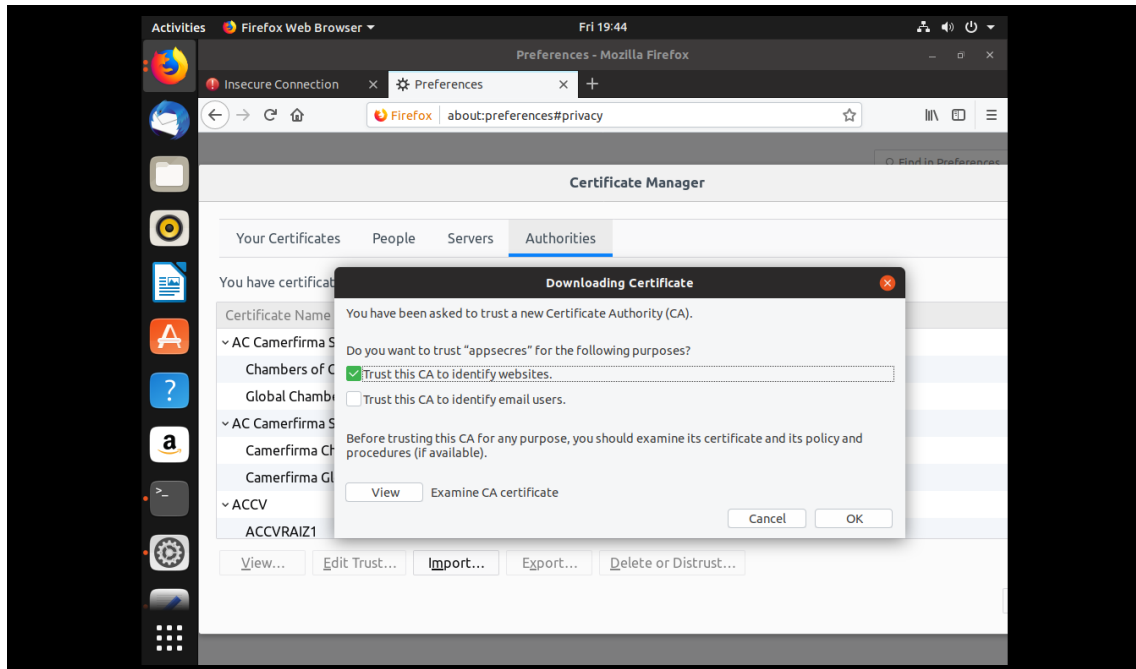


Figure 19: Reconnaissance de notre autorité de certificat par le navigateur web

Maintenant on a une erreur de validation du certificat de l'application cette fois ci, d'où l'intérêt d'avoir configuré le DNS sur fortigate

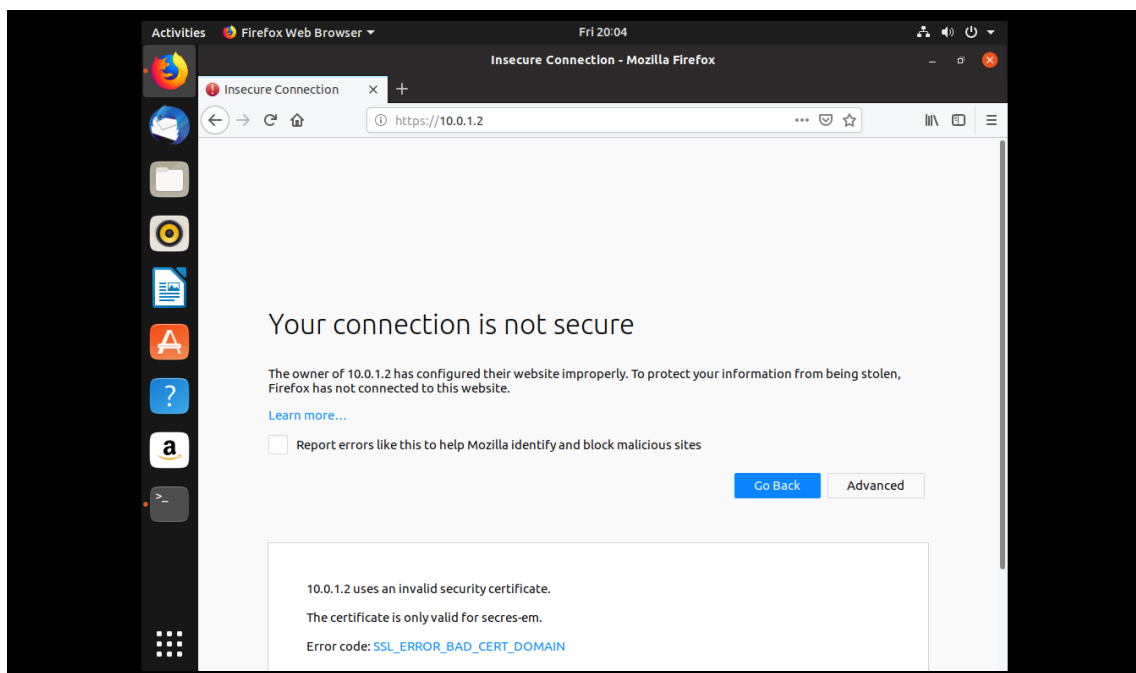


Figure 20: Erreur de nom de domaine lié au certificat utilisé par l'application web

et donc on y accede depuis secres.em

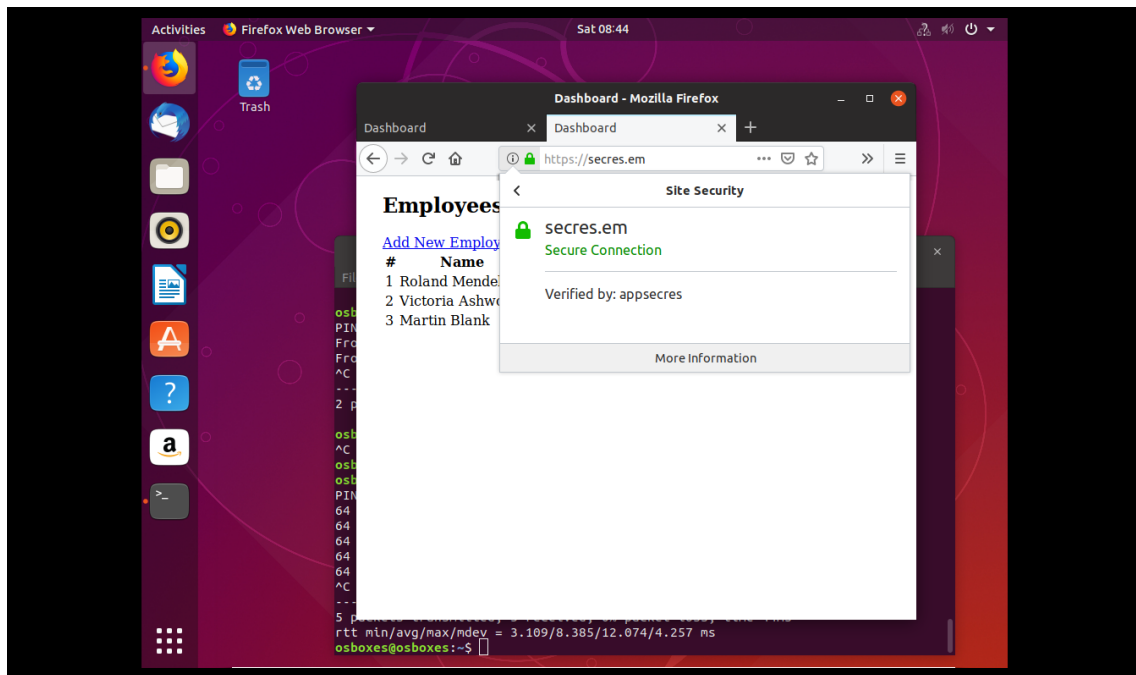


Figure 21: Accès du server web avec le nom de domaine secres.em