**Task 3 - TLS**

**环境配置**

进入实验目录/home/seed/Desktop/Labs\_20.04/Cryptography/Transport Layer Security (TLS) Lab/Labsetup/，使用dcup命令创建容器镜像和启动容器，新开启终端，输入命令dockps查看docker ID，分别为74、00、17开头，另外开启3个终端，输入docksh连接容器。

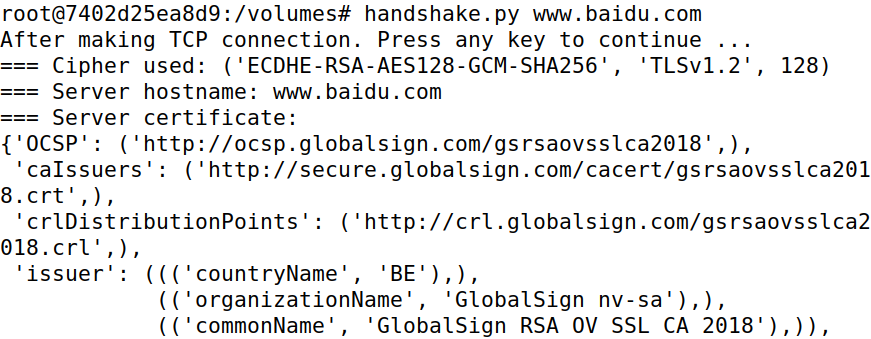


1. **TLS Client**

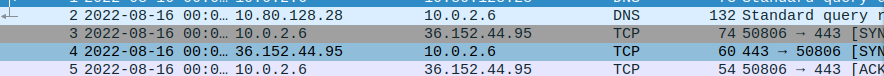
**1A TLS handshake**

启动wireshark，监听网卡enp0s3。

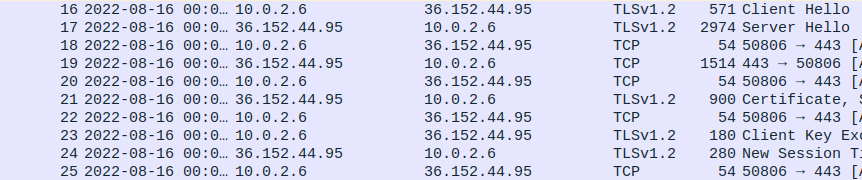
进入/volumes路径下，输入handshake.py [www.baidu.com](http://www.baidu.com)，可以看到服务器与客户端所使用的加密算法还有服务器的CA证书：



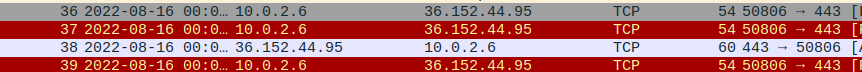
可以在wireshark中看到，运行后首先建立TCP连接：



按任意键后开始建立TLS连接：

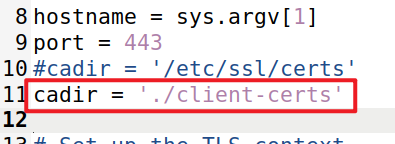


再次按下任意键后释放连接：

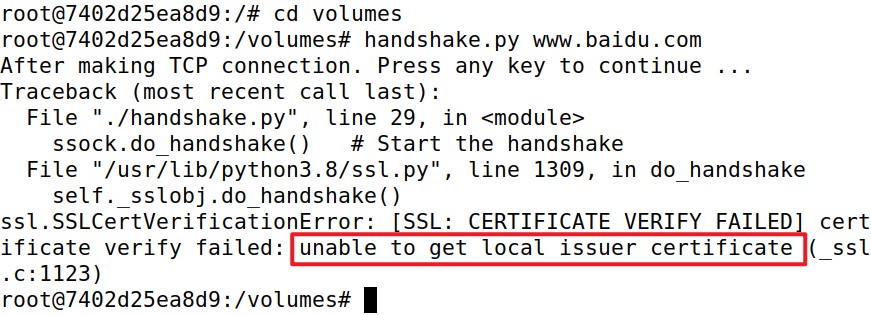


**1B CA’s Certificate**

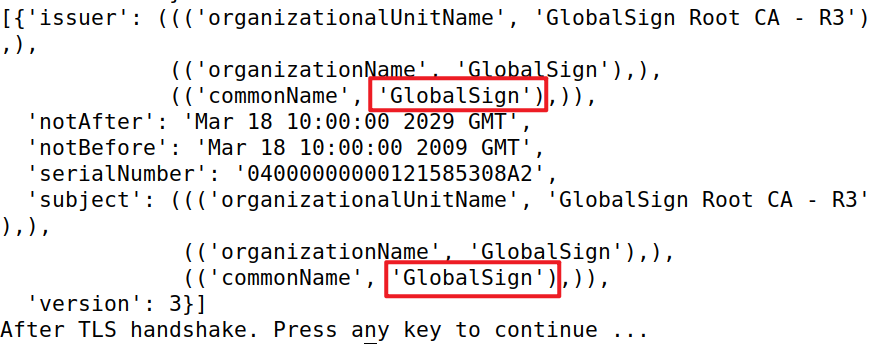
修改volumes路径下的handshake.py中的cadir，即证书路径：



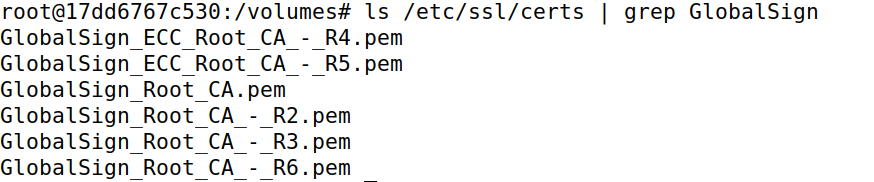
终端中输入handshake.py [www.baidu.com](http://www.baidu.com)，运行报错，显示原因为在建立TLS连接时出错，原因为无法获取本地证书：



接下来把需要的CA证书文件放入所设置的client-certs文件夹下，具体需要哪些CA证书可以查看最开始执行handshake.py时的输出信息中的issuer项和subject项，其中的GlobalSign就是所需要的CA证书：

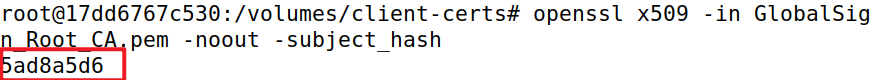


进入路径/etc/ssl/certs中寻找所需要的CA证书，使用grep GlobalSign来过滤显示结果：

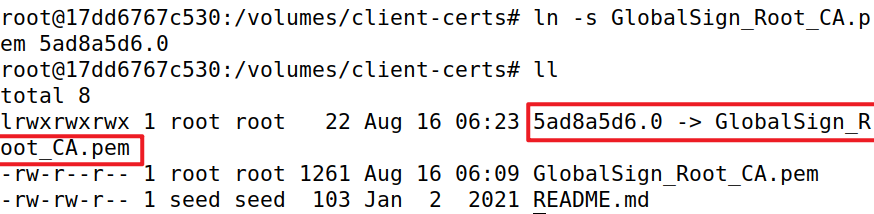


输入cp /etc/ssl/certs/GlobalSign\_Root\_CA.pem /volumes/client-certs/复制CA证书到client-certs路径下。

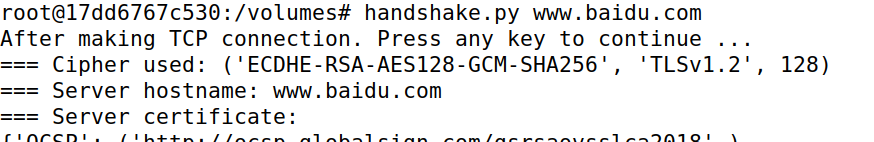
进入/volumes/client-certs路径下，输入openssl x509 -in GlobalSign\_Root\_CA.pem -noout -subject\_hash获取一个哈希值，为5ad8a5d6:



输入ln -s GlobalSign\_Root\_CA.pem 5ad8a5d6.0创建一个名为5ad8a5d6的软连接，连接到该CA证书，完成后输入ll查看：



回到volumes文件夹，输入handshake.py www.baidu.com，发现可以访问成功，结果如下：



按照前面类似的方法，使用另外两个网站实验，这里选用[www.seu.edu.cn](http://www.seu.edu.cn)和[www.zhihu.com](http://www.zhihu.com)。

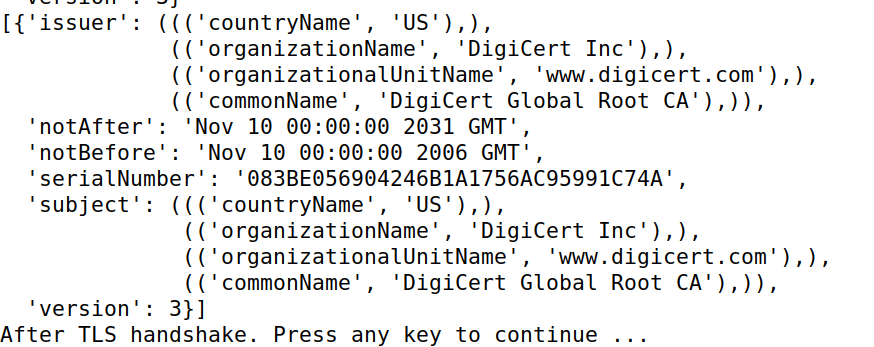
修改handshake.py中的证书路径为初始值。

在终端中输入handshake.py [www.seu.edu.com](http://www.seu.edu.com)，结果如下：



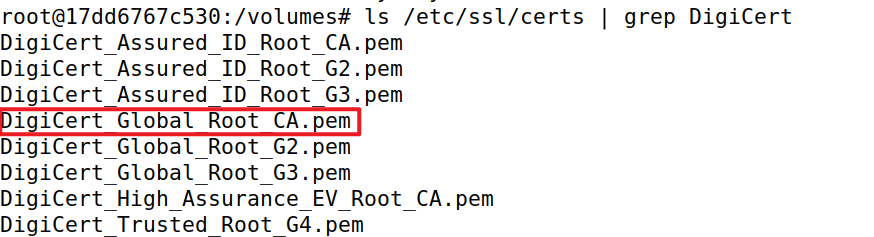
记录下所需要的CA证书为DigiCert。

输入handshake.py [www.zhihu.com](http://www.zhihu.com)，结果如下：



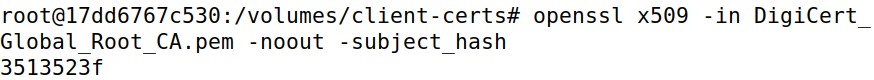
结果与前面的[www.seu.edu.cn](http://www.seu.edu.cn)相同。

进入路径/etc/ssl/certs中寻找所需要的CA证书，使用grep DigiCert来过滤显示结果：

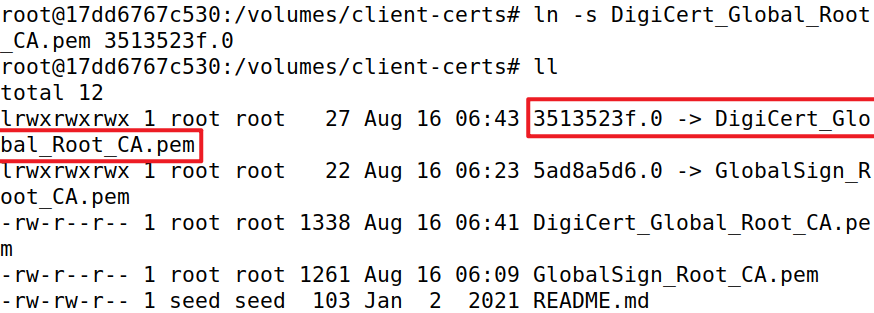


输入cp /etc/ssl/certs/DigiCert\_Global\_Root\_CA.pem /volumes/client-certs/复制CA证书到client-certs路径下。

进入/volumes/client-certs路径下，输入openssl x509 -in DigiCert\_Global\_Root\_CA.pem -noout -subject\_hash获取一个哈希值，为3513523f:

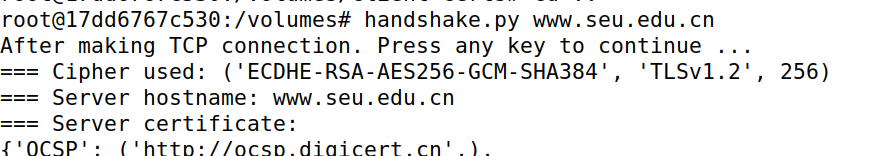


输入ln -s DigiCert\_Global\_Root\_CA.pem 3513523f.0创建一个名为3513523f的软连接，连接到该CA证书，完成后输入ll查看：

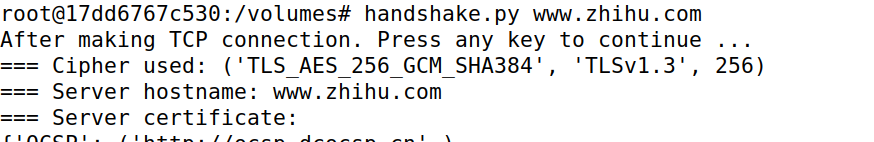


回到volumes文件夹，修改handshake.py中的证书路径。

输入handshake.py [www.seu.edu.cn](http://www.seu.edu.cn)，可以访问成功，结果如下：

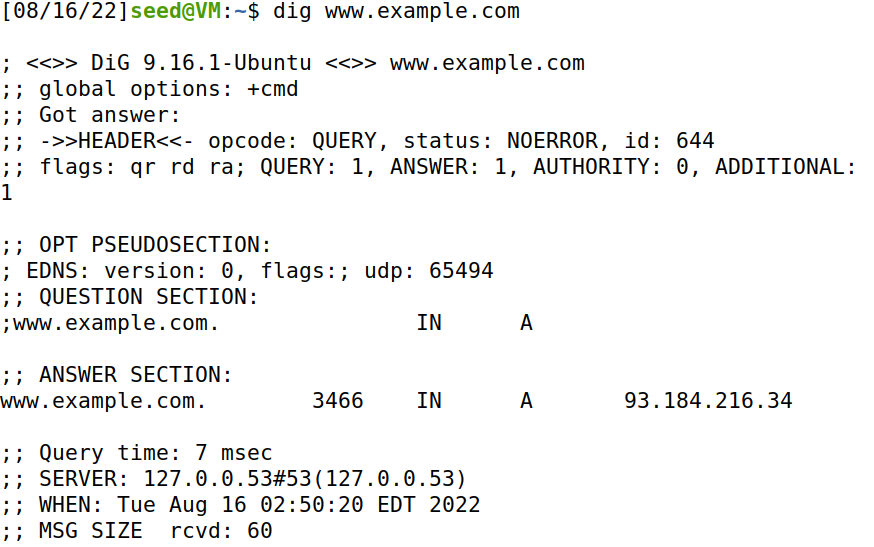


输入handshake.py [www.zhihu.com](http://www.zhihu.com)，可以访问成功，结果如下：



**1C Experiment with the hostname check**

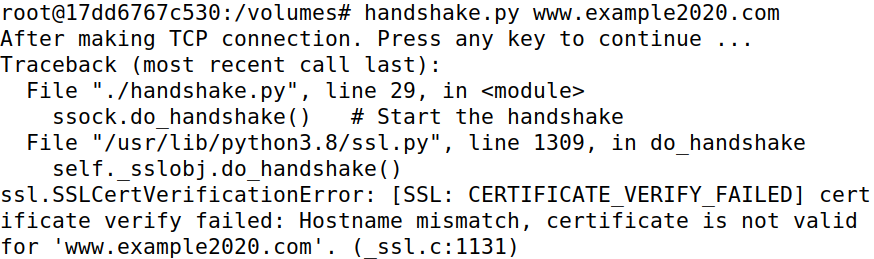
在主机终端中输入dig [www.example.com](http://www.example.com)，来获取ip地址：



将获取到的ip地址与另外一个域名www.example2020.com的对应关系添加到容器的hosts文件中：

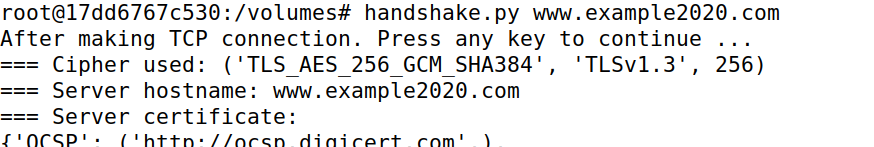


执行handshake.py [www.example2020.com](http://www.example2020.com)，结果如下：



修改handshake.py中的context.check\_hostname为False（初始状态为True），再次执行handshake.py [www.example2020.com](http://www.example2020.com)，结果如下：

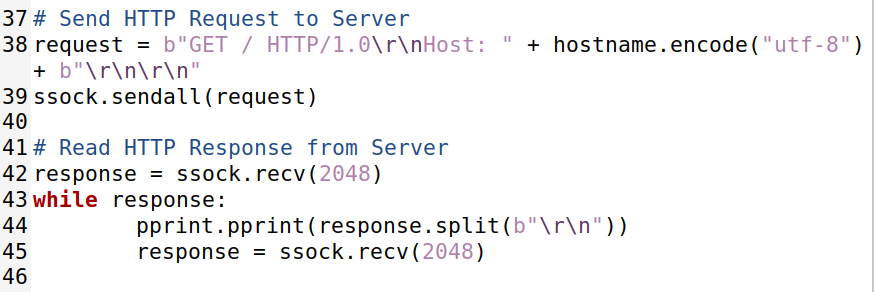




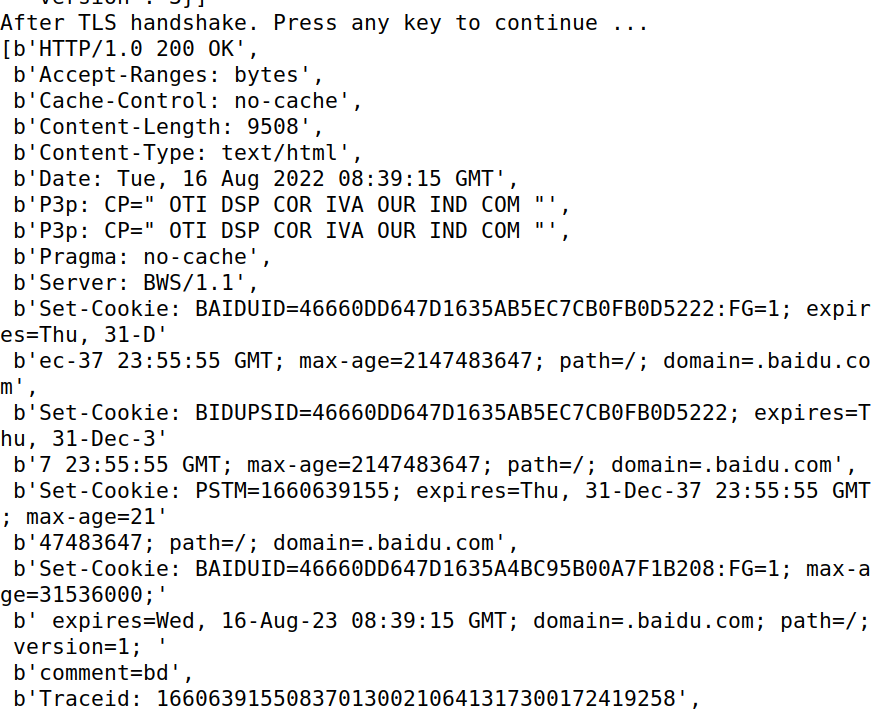
与实验手册上的结果相反，但通过查询context.check\_hostname的用法，认为在False时能访问通是正确的。

**1D Sending and getting Data**

修改handshake.py文件，添加如下内容：



在终端中输入handshake.py [www.baidu.com](http://www.baidu.com)，可以看到与前面相比，新增了以下内容，获取到的是[www.baidu.com](http://www.baidu.com)主页中的内容：



1. **TLS Server**

**2A Implement a simple TLS server**

所开启的服务器容器的ip地址是10.9.0.43，在客户端容器的hosts文件中新增如下dns记录，[www.seu2022.com](http://www.seu2022.com)是为服务器所设置的域名：

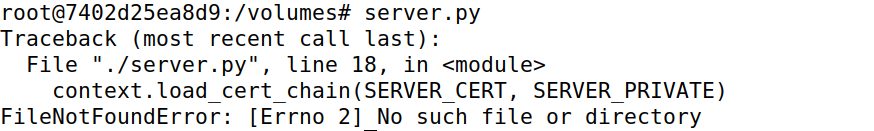


按照实验二中的步骤，生成证书签名，为避免重复在生成ca.crt、ca.key时将颁发机构设为Model CA1，另外生成server.crt、server.key时将域名修改为服务器容器所设域名，将生成的server.key和server.crt复制到/volumes/server-certs文件夹中。

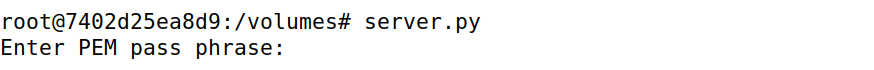
修改server.py中的路径与名称，修改如下：



在服务器容器中输入server.py启动服务：



提示报错，查找原因发现，错把证书与密钥文件放入了client-certs路径下，再移动到server-certs路径下，重新执行命令：

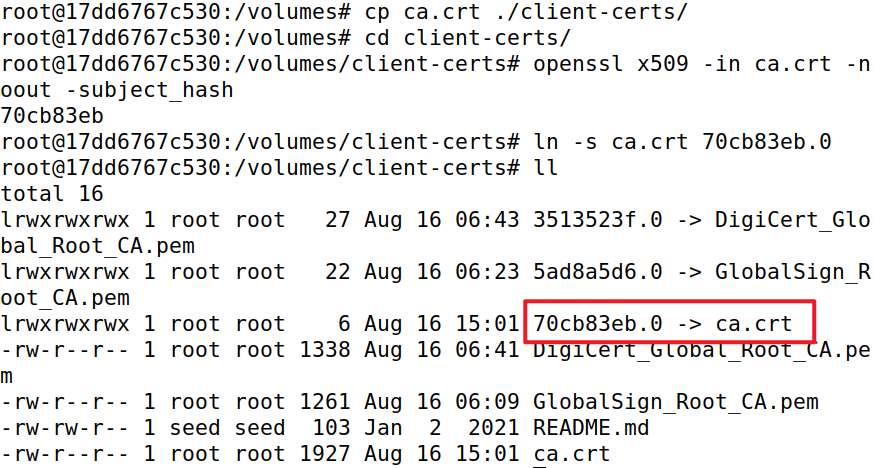


口令输入dees。

输入cp ca.crt ./client-certs复制ca.crt到client-certs路径下。

进入client-certs文件夹，输入openssl x509 -in ca.crt -noout -subject\_hash，获取文件的哈希值，得到的值为70cb83eb。

输入ln -s ca.crt 70cb83eb.0创建软连接，输入ll查看连接情况：

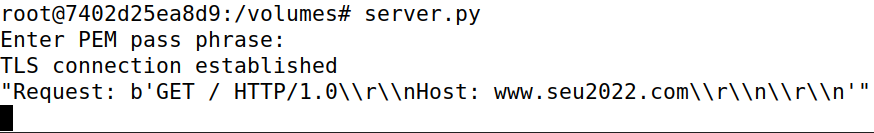


回到volumes文件夹下，修改handshake.py中的cadir为'./client-certs'。

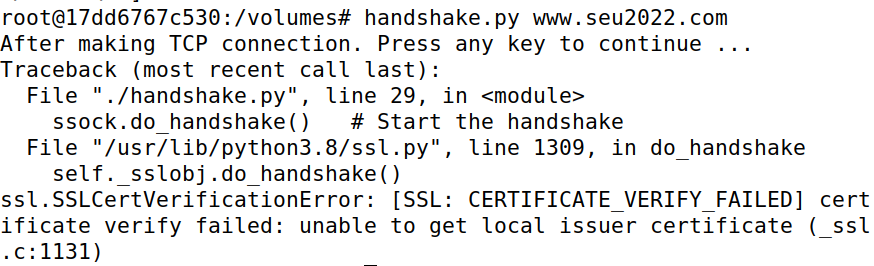
输入handshake.py [www.seu2022.com](http://www.seu2022.com)，结果如下：



可以看到成功建立了连接，证书机构等也正确，并且在服务器容器中也有了相应的输出：



接下来将handshake.py中的证书路径修改回/etc/ssl/certs，再次运行handshake.py，结果如下：

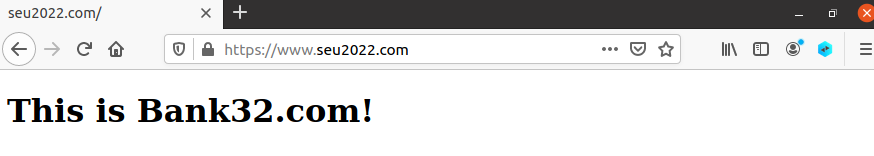


能够建立起TCP连接，但无法定位到证书，所以TLS连接建立失败，在服务器容器中可以看到TLS建立失败：



**2B Testing the server program using browsers**

在firefox浏览器中添加新的ca.crt证书，之后访问[www.seu2022.com](http://www.seu2022.com)，失败，无法连接，原因是忘记在虚拟机的hosts文件中添加[www.seu2022.com](http://www.seu2022.com的ip)的ip地址，添加后再次访问：



成功访问。

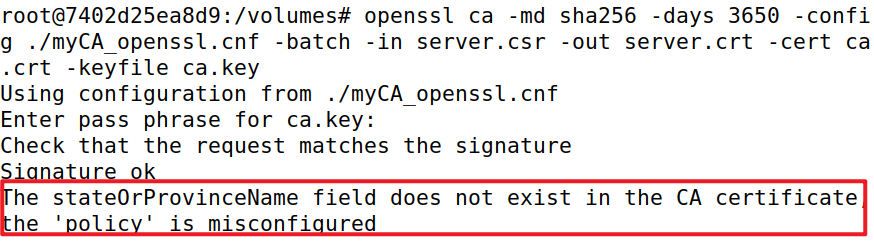
**2c Certificate with multiple names**

在volumes文件夹下新建配置文件server\_openssl.cnf，填写如下配置信息，保存文件：

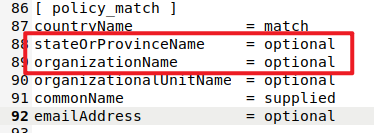


输入openssl req -newkey rsa:2048 -config ./server\_openssl.cnf -batch -sha256 -keyout server.key -out server.csr，生成csr文件。

输入openssl ca -md sha256 -days 3650 -config ./myCA\_openssl.cnf -batch -in server.csr -out server.crt -cert ca.crt -keyfile ca.key，这里注意要将实验手册中中的myopenssl.cnf替换为本机上的名称myCA\_openssl.cnf，执行结果如下：

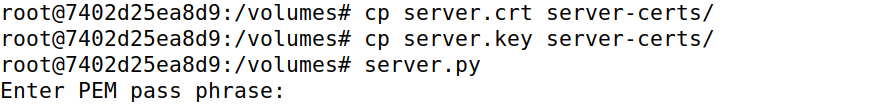


出现报错信息，查找原因，需要myCA\_openssl.cnf中的部分参数，来调整匹配规则，修改如下：

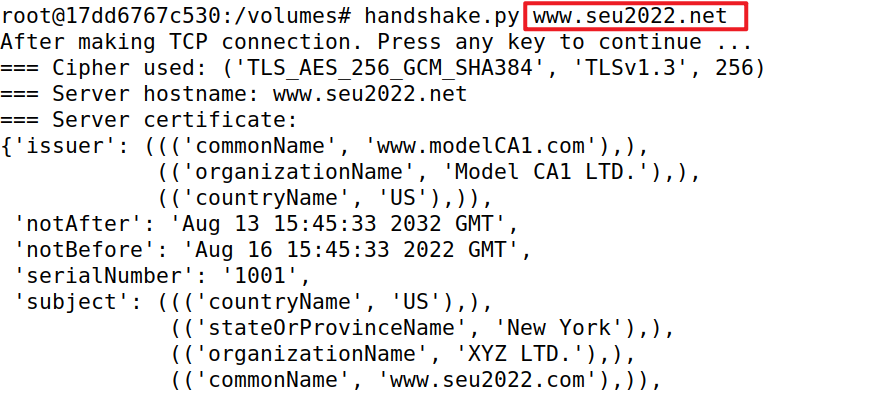


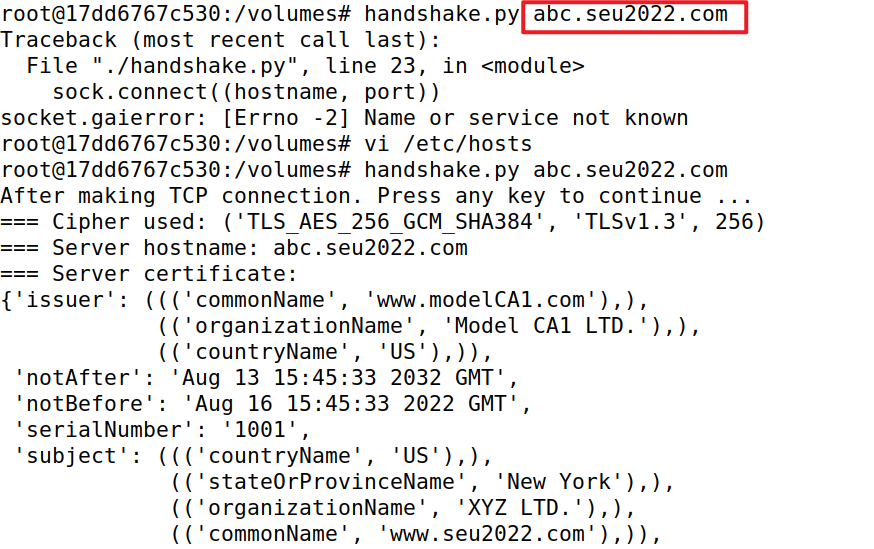
重新执行以上两条指令，成功。

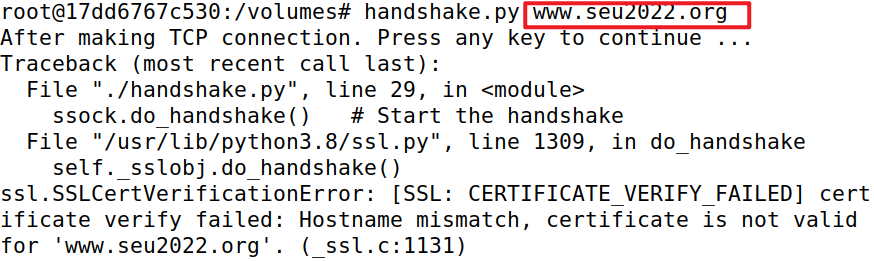
使用新生成的证书开启server服务，需要将新服务器证书与密钥复制到server-certs路径下，执行server.py，成功开启：



依次通过handshake.py访问www.seu2022.net、abc.seu2022.com、www.seu2022.org，其中前两个是被添加进配置文件的别名，而第三个不是，但都在hosts中有记录，访问结果如下：

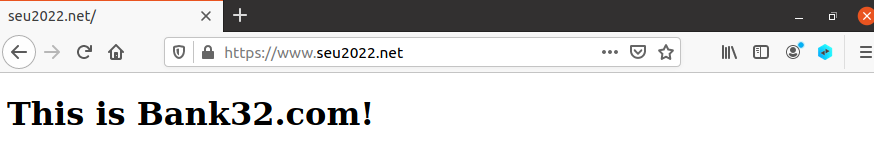


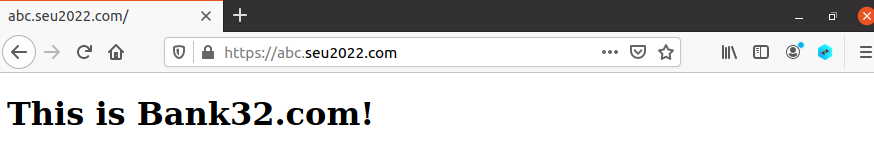




可以看到前两个访问成功，而第三个不成功。

在浏览器中的访问结果如下：





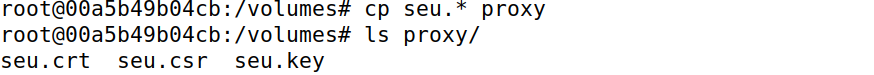
1. **A Simple HTTPS Proxy**

选择网站[www.seu.edu.cn](http://www.seu.edu.cn)为例，在代理容器终端中输入openssl req -newkey rsa:2048 -sha256 -keyout seu.key -out seu.csr -subj "/CN=www.seu.edu.cn/O=seu Inc./C=CN" -passout pass:dees，生成seu.csr和seu.key两个文件。

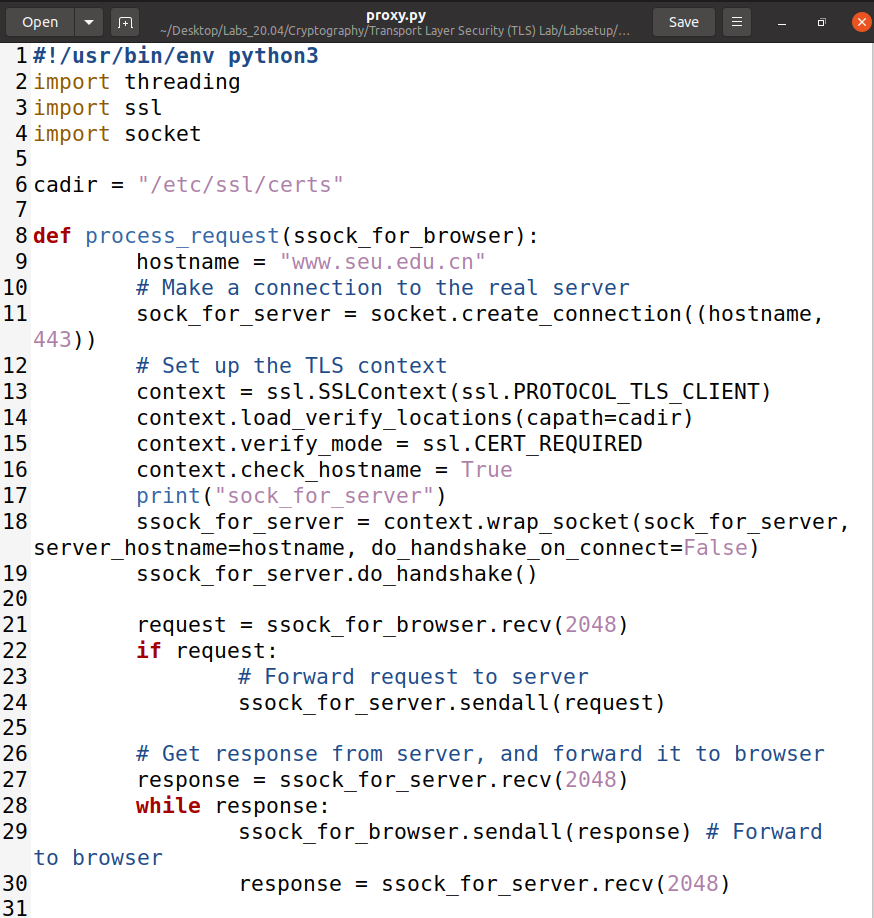
输入openssl ca -config myCA\_openssl.cnf -policy policy\_anything -md sha256 -days 3650 -in seu.csr -out seu.crt -batch -cert ca.crt -keyfile ca.key，生成seu.crt文件。

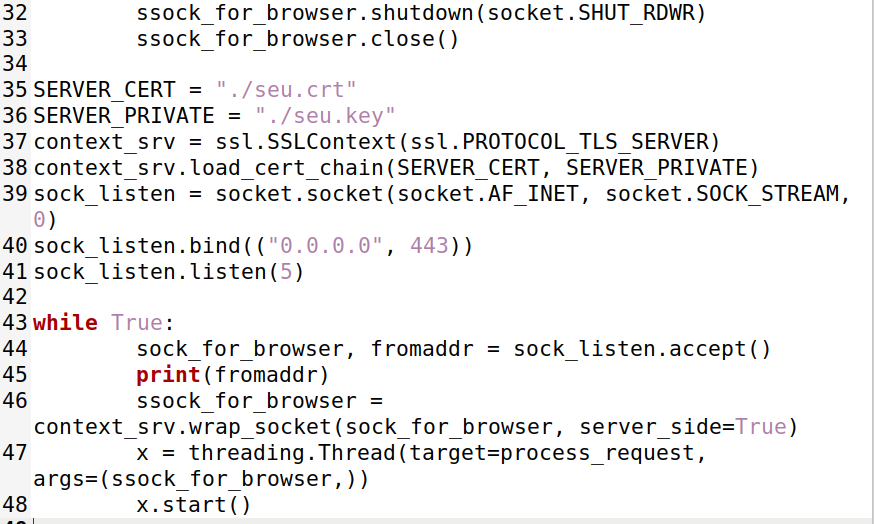
输入mkdir proxy，在volums路径下新建文件夹proxy/。

输入cp seu.\* proxy，将生成的seu.\*三个文件复制到该文件夹下：



输入touch proxy/proxy.py，编写以下程序：





注意修改示例程序中的[www.zhihu.com](http://www.zhihu.com)为[www.seu.edu.cn](http://www.seu.edu.cn)。

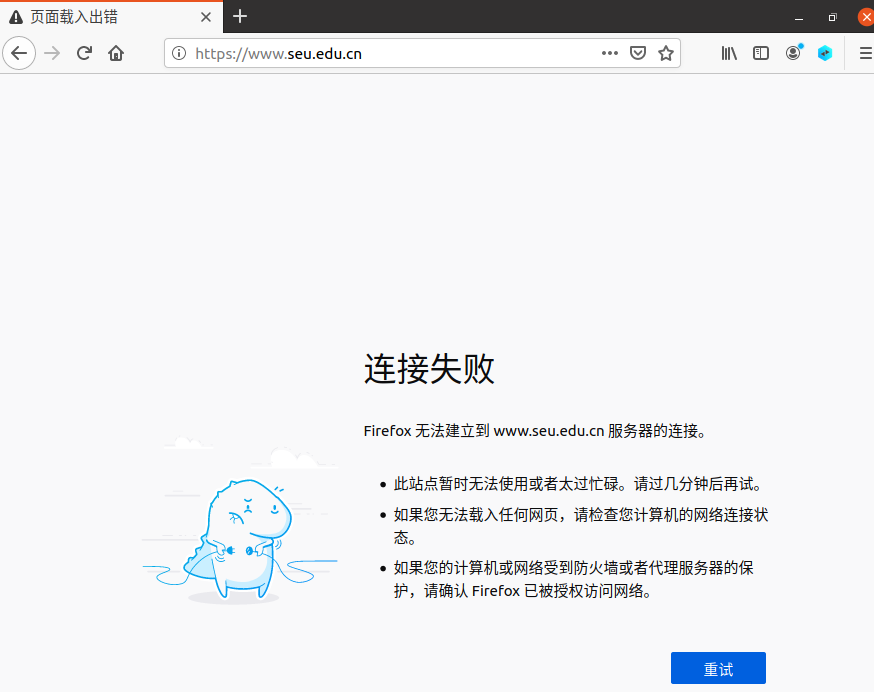
修改虚拟机hosts文件，将[www.seu.edu.cn的ip](http://www.seu.edu.cn的ip)地址映射到10.9.0.143（代理容器）：



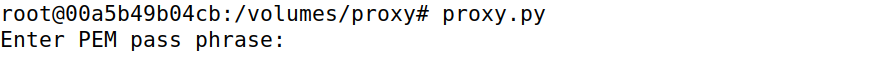
在代理容器中修改/etc/resolv.conf文件，nameserver的值修改为8.8.8.8，保存：



在firefox浏览器中访问[www.seu.edu.cn](http://www.seu.edu.cn)，连接失败：



在代理容器中启动proxy.py：



再次在浏览器中访问[www.seu.edu.cn](http://www.seu.edu.cn)：



访问成功，同时在代理容器中看到输出：

