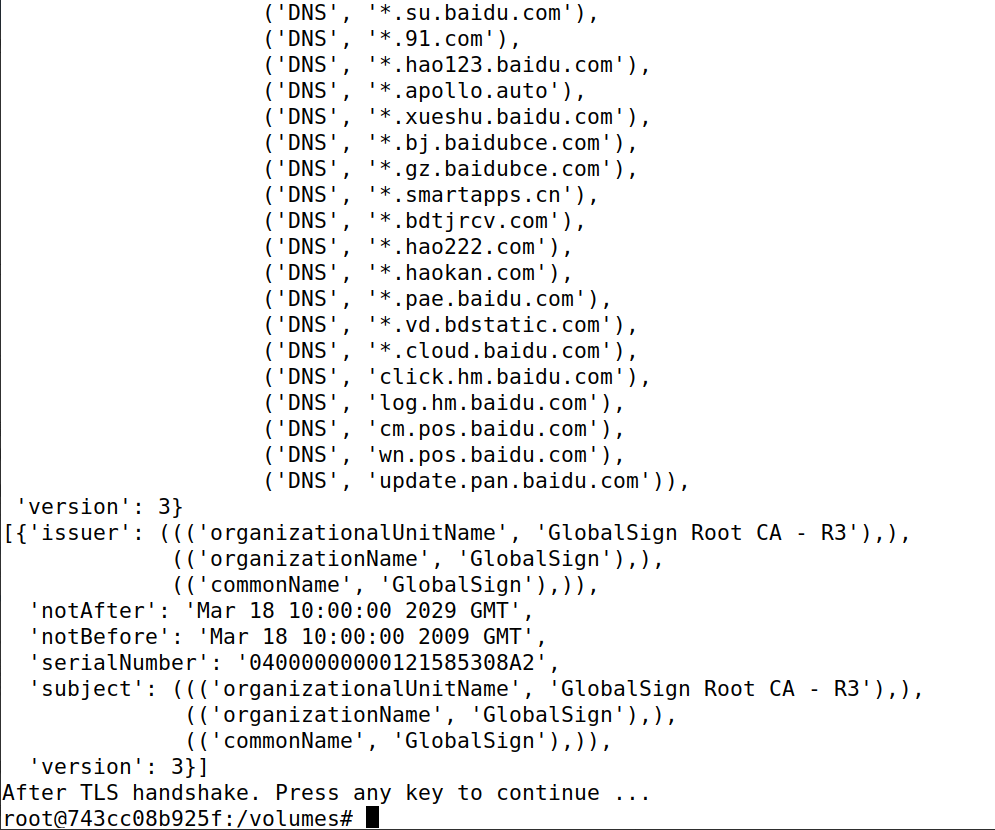
|  |
| --- |
| 哈尔滨工业大学(深圳) |
| **《网络与系统安全》 实验报告** |
|  |
| 实验五  TLS 实验  学 院: 计算机科学与技术   |  |  | | --- | --- | | 姓 名: | 宗晴 | | 学 号: | 200110513 | | 专 业: | 计算机 | | 日 期: | 2023年5月 | |

1.在客户端容器中执行如下命令 ./handshake.py www.baidu.com根据执行结果回答下面三个问题。







（1）客户端和服务器端使用的加密算法有哪些，分别起什么作用？

客户端和服务器端使用的加密算法有：ECDHE-RSA-AES128-GCM-SHA256。

RSA是一种公钥密码算法，是第一个能同时用于加密和[数字签名](http://baike.baidu.com/view/7626.htm)的算法，也易于理解和操作。

ECDHE算法是在 DHE 算法的基础上利用了 ECC 椭圆曲线特性，可以用更少的计算量计算出公钥，以及最终的会话密钥。具有前向安全，被广泛使用。

AES128是一个分组密码，属于对称密码范畴，AES算法的模块在对称密码领域特别是分组密码领域常有使用。AES128分组长度为128比特，密钥长度也为128比特。算法涉及4种操作：字节替代、行移位、列混淆和轮密钥加。

GCM ( Galois/Counter Mode) 指的是该对称加密采用Counter模式，并带有GMAC消息认证码。二者分别保证了加密算法的保密性、完整性。

SHA-256作为SHA-2(Secure Hash Algorithm 2，安全哈希算法2) 的一部分，目前已经是最流行的哈希算法之一。安全加密算法通过将输入文本拆分成独立的片段，并通过这些独立的片段生成最终的结果——加密算法哈希值。这些加密算法哈希值几乎是唯一的字符串，因而它们往往被用作数据块的摘要"digest"，指纹"figerprint"或签名"signature"。SHA-256算法往往被用来生成256位的签名。

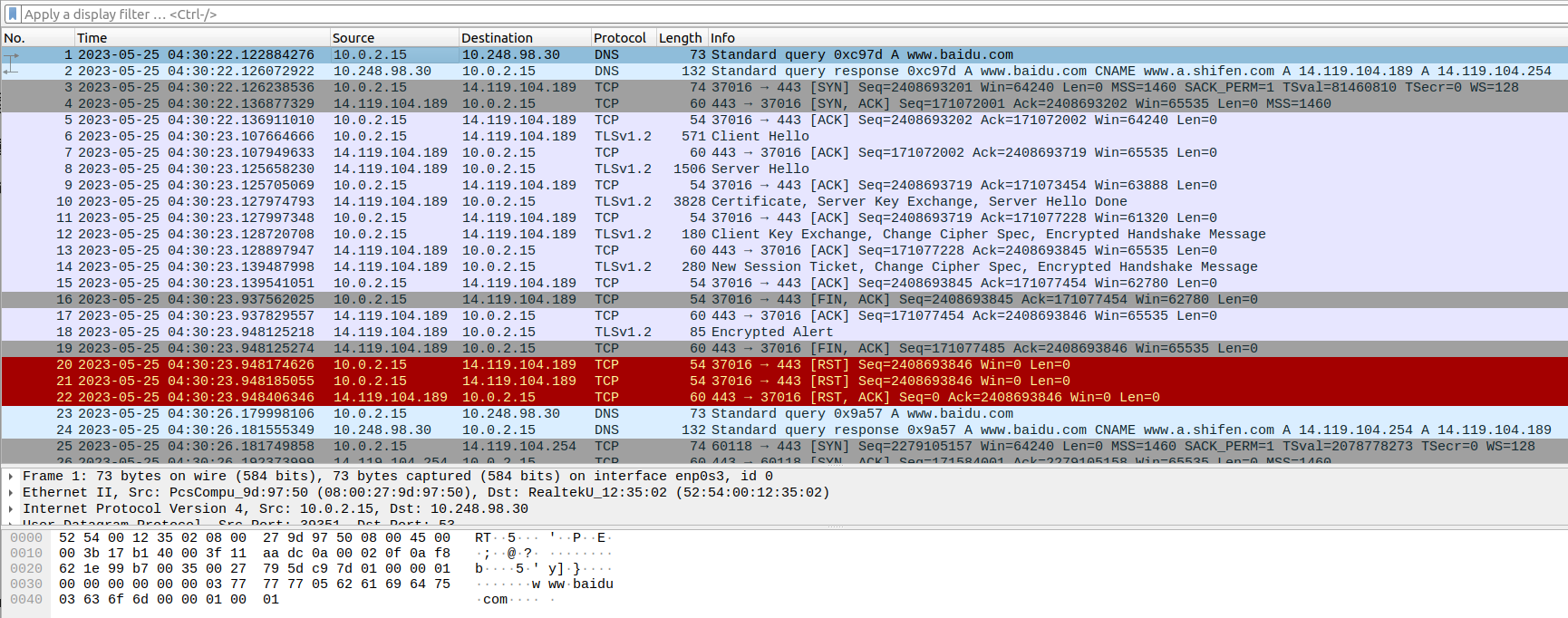
（2）分析打印出来的服务器端证书

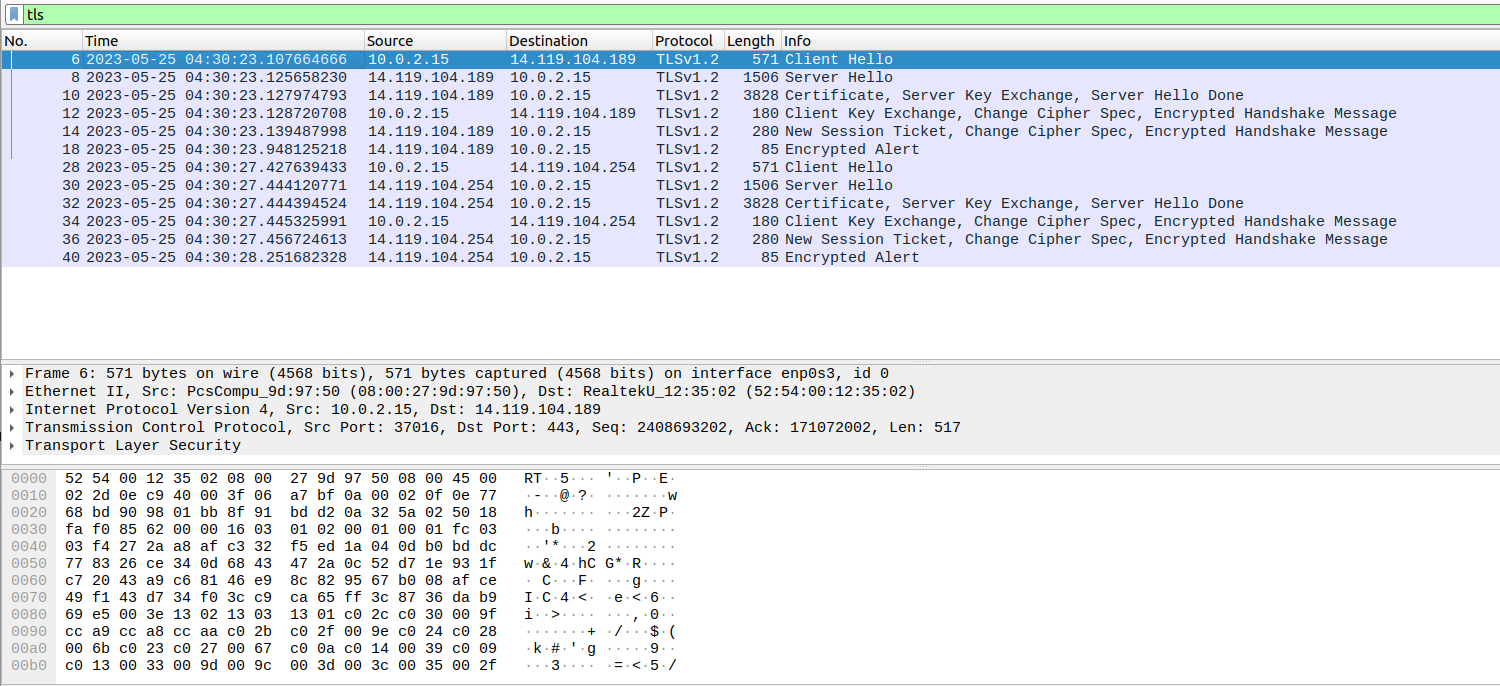
服务器端证书如上图所示，为“===Server certificate: ”字段后的内容。

在打印出来的服务器端证书前首先说明了签名算法、主机名（www.baidu.com）。然后证书中说明了所用的OCSP即在线证书状态协议、caIssuers即证书的签发机构等信息，以及notAfter和notBefore即证书的有效时间、serialNumber即证书的独特序列号、subject即证书的拥有者信息、subjectAltName即证书拥有者的别名、version即版本号为3，以及其他相关信息。

（3）抓包分析TLS握手协议

抓包结果如下：





如上图所示，为进行了两次TLS握手的抓包过程（已经筛选出了所有的TLS数据包）。可以看出TLS握手过程如下：

（1）客户端：发送客户端问候消息（Client Hello），表明它自己支持哪些密码套件和客户端的一次性随机数（Client\_random）。

（2）服务器：发送服务器问候消息（Server Hello），根据客户端发来的问候消息，选择确认一个客户端和服务器都支持的密码套件，并提供服务器的一次性随机数（Server\_random）。

（3）服务器：发送公钥证书（certificate，server key exchange）给客户端。

（4）服务器：发送握手完成的消息，表明已完成握手协商。（server hello done）

（5）客户端：发送客户端密钥交换消息（client key exchange）。客户端随机生成一个预主密钥，然后用服务器的公钥对其进行加密，并将加密后的密钥发送给服务器。客户端和服务器首先使用预主密钥生成主密钥，然后再使用主密钥生成会话密钥。

（6）客户端和服务器：互相发送更改密码规范消息。

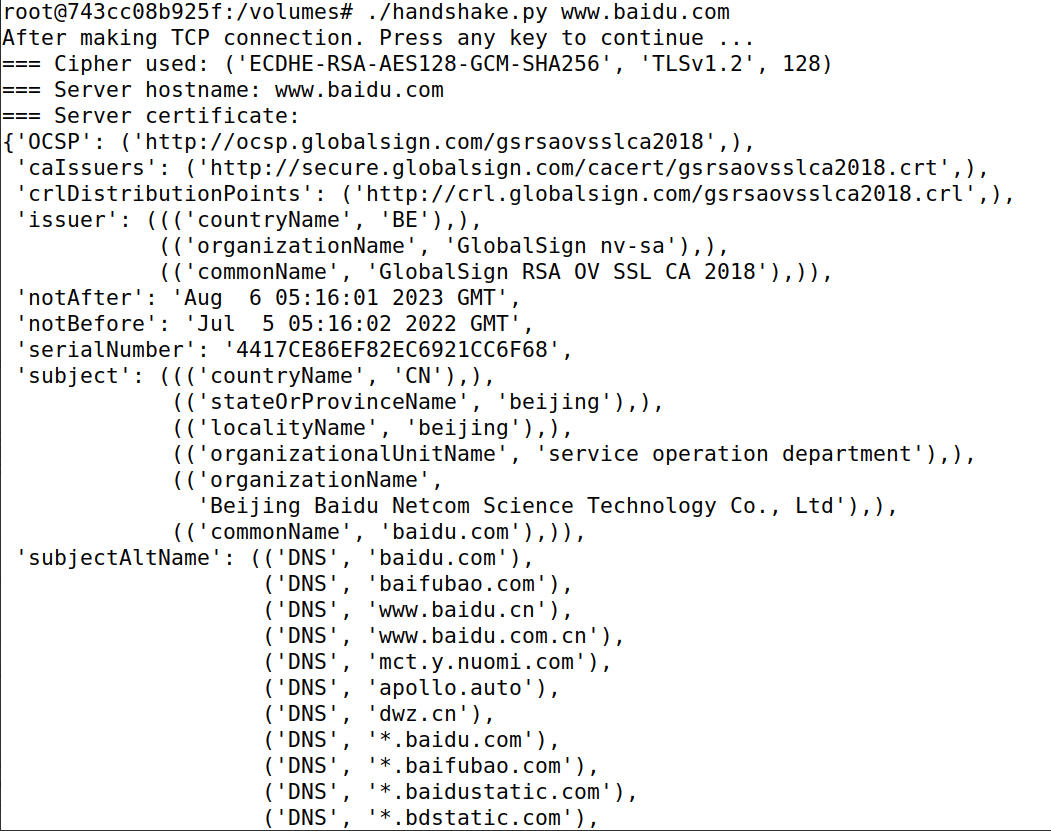
（7）客户端和服务器：互相发送一个加密完成的消息。

2.更改证书文件路径，请同学们将www.baidu.com网站的测试过程截图保存（如果不将证书拷贝过来应该有报错信息，拷贝过来之后应该正常），也可选用其他网站做测试。

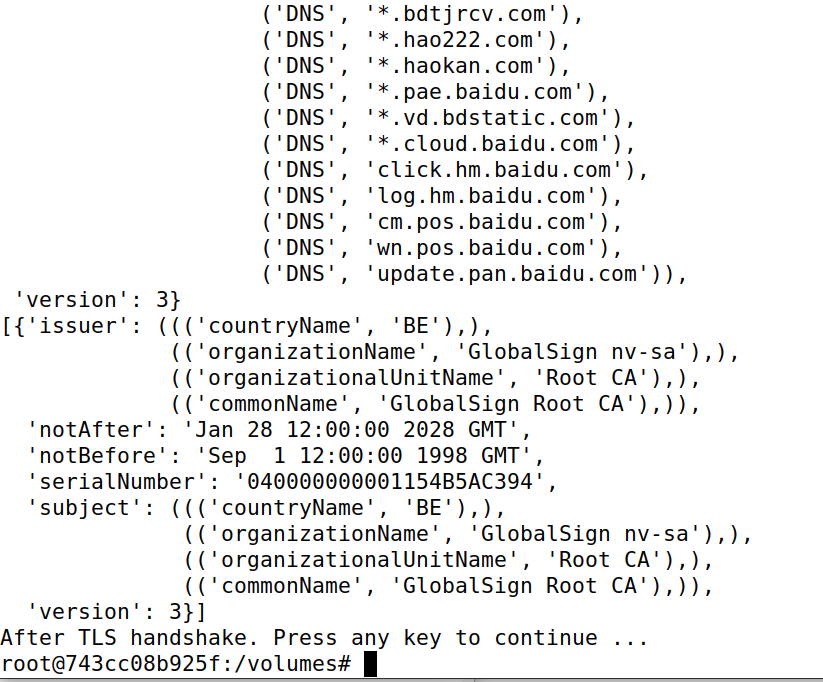
如果不将证书拷贝过来，执行./handshake.py www.baidu.com会有报错信息，如下图所示：



如果将证书拷贝过来之后，显示正常，如下图所示：

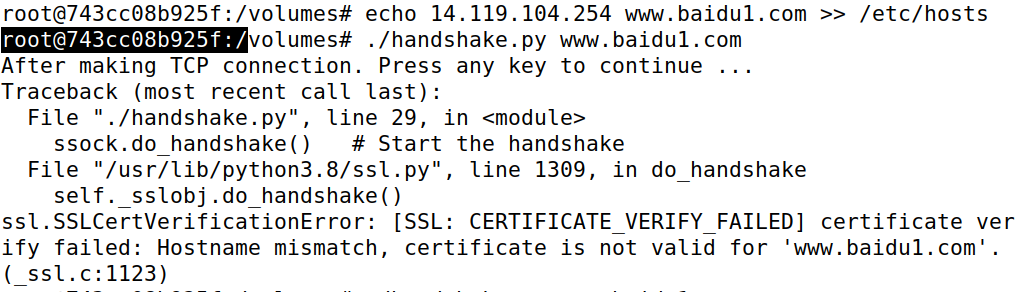




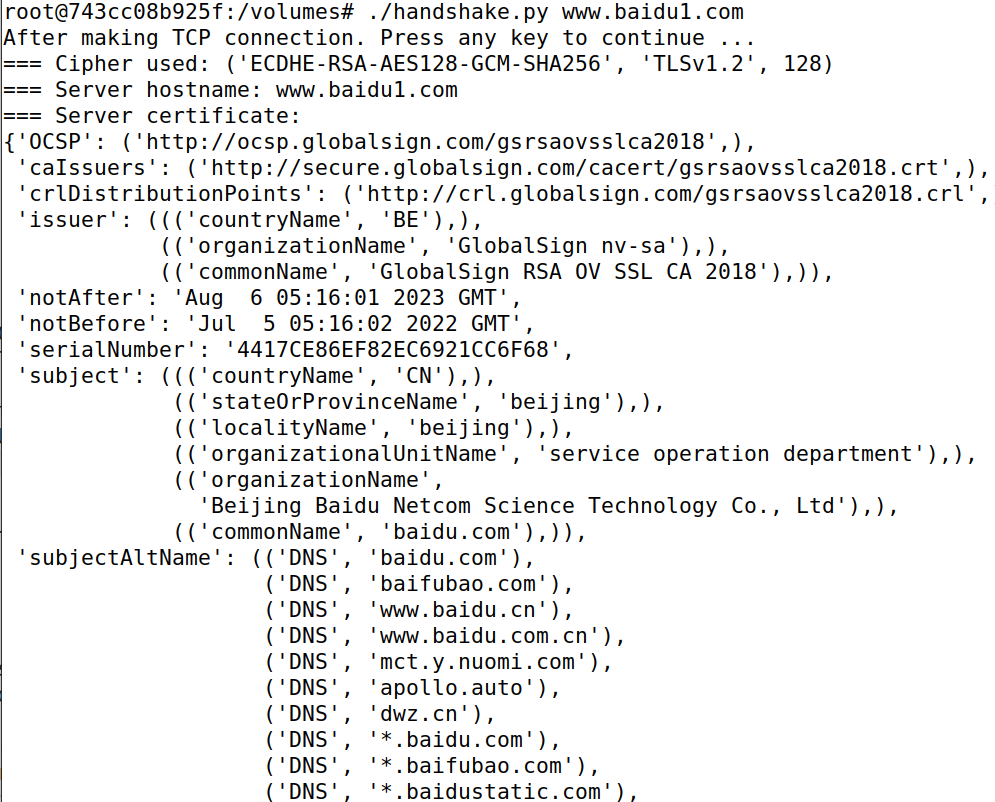


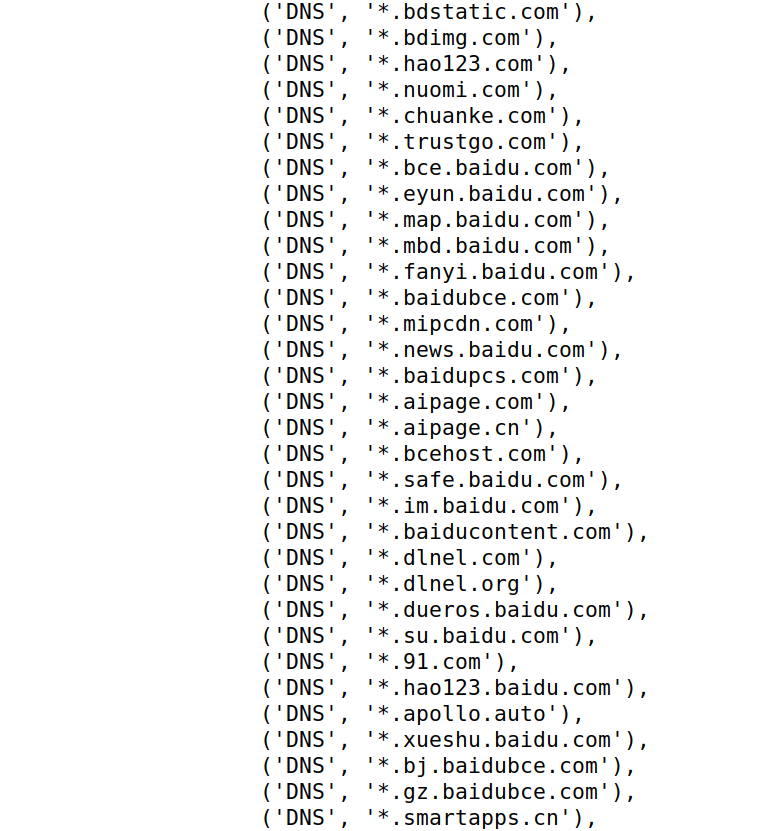
3. 请同学们将修改 www.baidu.com 网站主机名的测试过程截图保存在报告里并分析执行的结果，也可选用其他网站做测试。

当context.check\_hostname = True时，执行./handshake.py www.baidu.com会有报错信息，显示、Hostname不匹配，如下图所示：



当context.check\_hostname = False时，执行./handshake.py www.baidu.com显示正常，如下图所示：



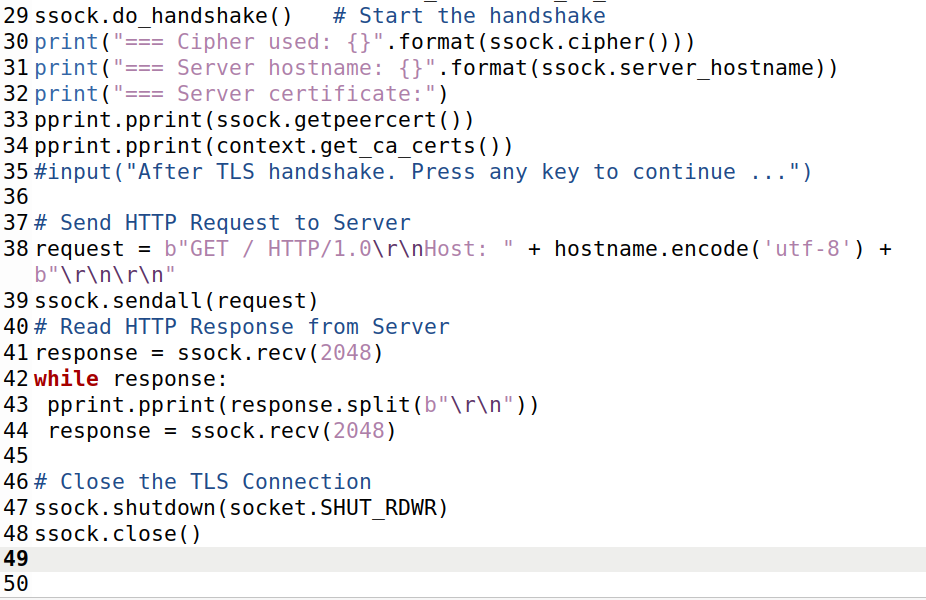




这是因为，如果context.check\_hostname = False，当服务器返回"Server Hello"消息时，客户端将不会对服务器的证书中的hostname进行校验。在此示例中，服务器证书主机名应该是 www.baidu.com，但由于客户端不进行主机名校验，所以可以接受证书中的主机名为 www.baidu1.com，并与服务器建立连接。

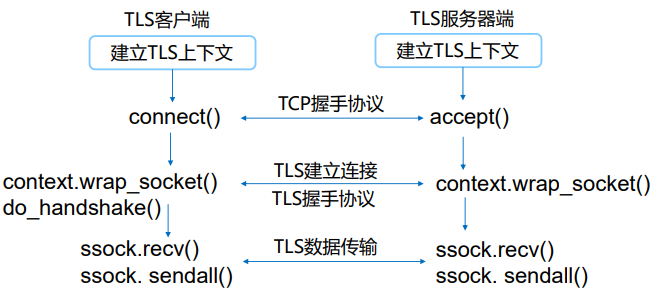
4.请分析TLS客户端编程和server.py的代码，说明客户端和服务器程序的关键步骤。

利用handshake.p修改client.py，并增加相应代码：



分析代码可知，客户端和服务器程序的关键步骤为，双方首先利用connect()函数和accept()函数确定TCP握手协议，然后利用context.wrap\_socket()函数和do\_handshake()函数进行TLS建立连接与TLS握手协议，最后利用ssock.recv()函数和ssock.sendall()函数进行TLS数据传输。

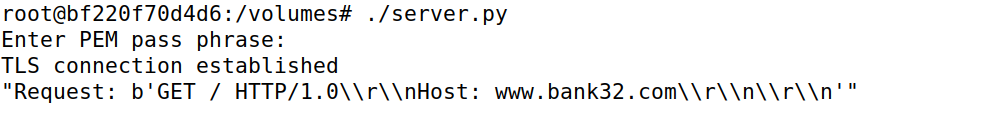
过程如下图所示：



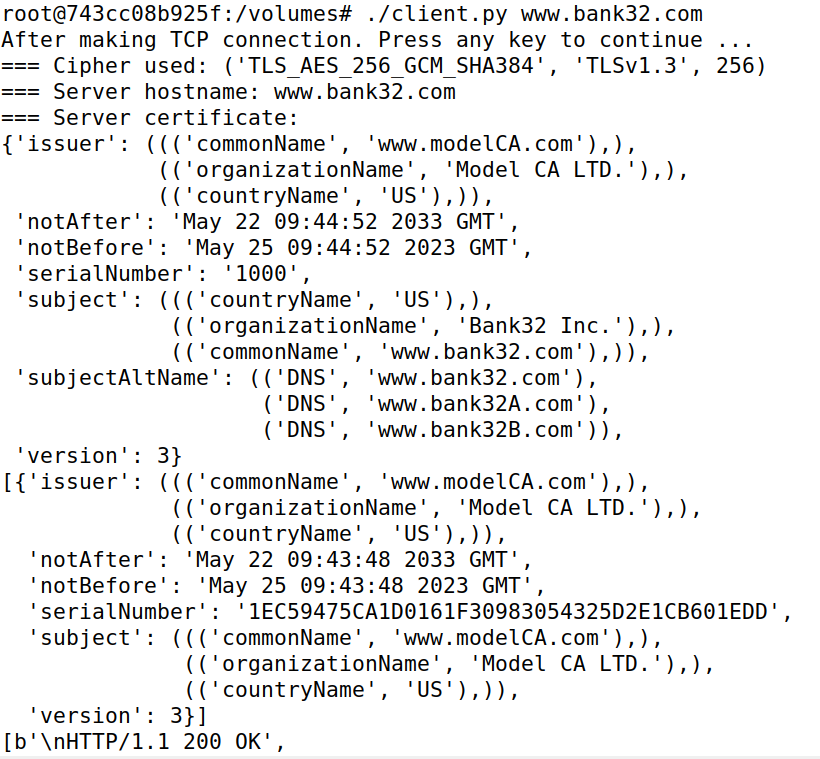
5.请分别用client.py和浏览器两种方式访问服务器，并记录你观察的结果（截图）

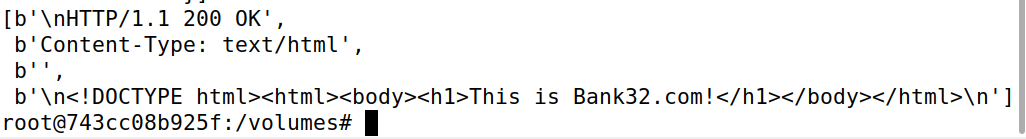
用client.py访问服务器的结果如下：

客户端处的结果：



服务器处的结果：





用浏览器访问服务器的结果如下：

