# python实现基于OpenSSL的SSL通信

1. 实验原理

openssl 是目前最流行的 [SSL](https://so.csdn.net/so/search?q=SSL&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/gengxiaoming7/article/details/_blank) 密码库工具，其提供了一个通用、健壮、功能完备的工具套件，用以支持SSL/TLS 协议的实现。openssl在SSL的基础上，另外包含了公钥私钥的生成、摘要生成等各种工具。

例如有些时候我们浏览网站的时候会有一些广告，这些广告什么的不一定是原网站挂上去的，也有可能是中间的运营商在中间篡改了内容导致的，可以使用https技术（一般是基于openssl）来对数据进行加密，保证数据不被篡改。

SSL协议通信的握手步骤如下：

第1步，SSL客户机连接至SSL服务器，并要求服务器验证它自身的身份；

第2步，服务器通过发送它的数字证书证明其身份。这个交换还可以包括整个证书链，直到某个根证书颁发机构（CA）。通过检查有效日期并确认证书包含可信任CA的数字签名来验证证书的有效性。

第3步，服务器发出一个请求，对客户端的证书进行验证，但是由于缺乏公钥体系结构，当今的大多数服务器不进行客户端认证。

第4步，协商用于加密的消息加密算法和用于完整性检查的哈希函数，通常由客户端提供它支持的所有算法列表，然后由服务器选择最强大的加密算法。

第5步，客户机和服务器通过以下步骤生成会话密钥：

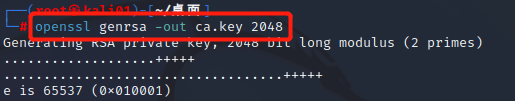
客户机生成一个随机数，并使用服务器的公钥（从服务器证书中获取）对它加密，以送到服务器上。

服务器用更加随机的数据（客户机的密钥可用时则使用客户机密钥，否则以明文方式发送数据）响应。

1. 实验操作
2. **在kali环境下配置openssl证书内容**

先创建 CA 私钥，生成ca.crt；密钥长度为2048位

e默认是65537



生成自签名CA证书：

req X.509证书签发请求(CSR)管理

-new 新的请求

-x509 输出一个X509格式的证书

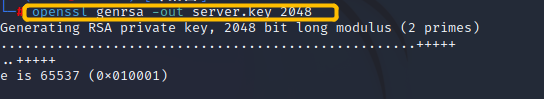
-days X509证书的有效时间

-key 用于签名待生成的请求证书的私钥文件

-subj 参数指定证书信息，避免在终端逐个输入



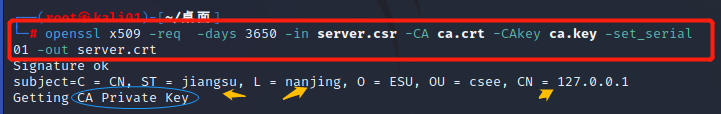
生成服务器私钥



生成要颁发证书的证书签名请求



用第2步创建的 CA 证书给第4步生成的签名请求进行签名，表明该证书请求已被CA信任，得到一个被CA签名过的证书。

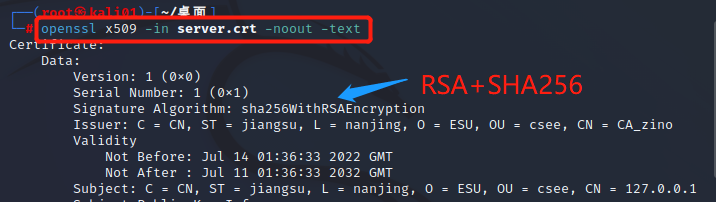


客户端用CA证书来对服务端被CA签名过的证书来进行认证



结果是签名验证成功

查看证书具体信息



Issuer:

证书颁发机构CA

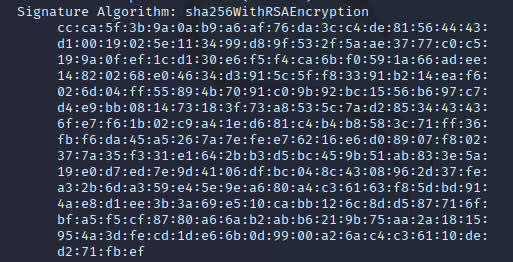
Subject:

拥有证书的网站信息

查看证书里的公钥信息



查看数字签名



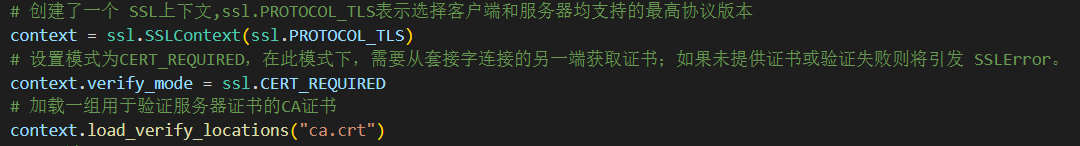
1. 编写代码实现SSL通信

对连接的另一方的证书进行验证，客户端必须提供一个 “CA 证书” 文件，即CA机构的公钥，用这个公钥去验证被CA机构的颁发的目标网站的证书，这个证书相当于被CA机构的私钥签名过。

Server

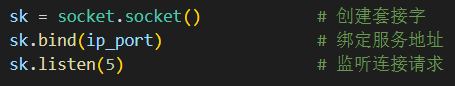


Client



所以如果通过验证，则表明该网站被CA信任，那么客户端就可以相信它。

创建套接字及绑定地址进行监听



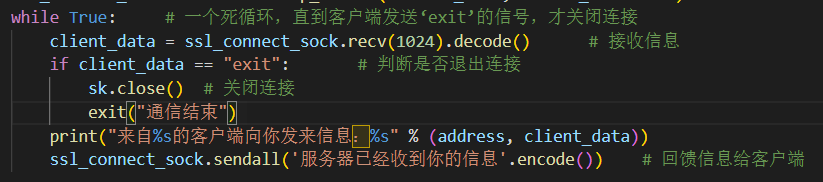
客户端返回的 SSL 套接字会绑定上下文、设置以及证书



服务端包装一个现有的 Python socket,并返回一个ssl socket,server\_side为true表示为服务器行为，默认为false则表示客户端



服务端接收消息



客户端连接服务器



先打印证书信息

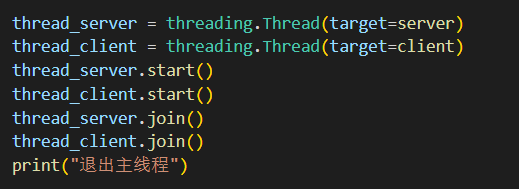


连接上服务器后进行消息传输



Main.py

负责整合服务端和客户端的内容，进行封装



实现在main.py调用server和client的所有功能，进行一键通信

#把所有编写好的py文件和证书放在同一文件夹下

1. 实验结果

运行server.py首先打开服务端



打开客户端输入发送消息

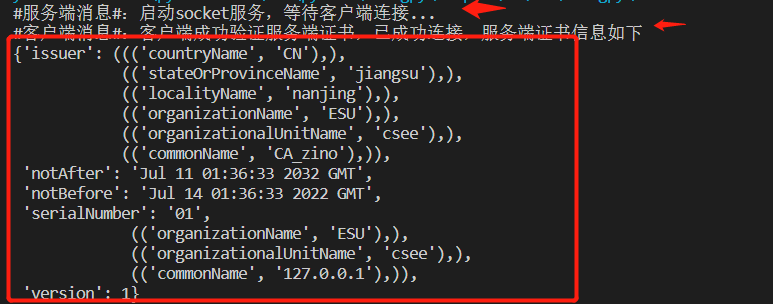


服务端接收完成



运行main.py

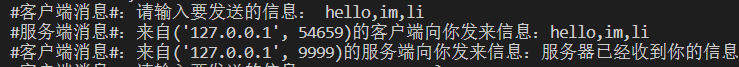
打印出来的证书信息

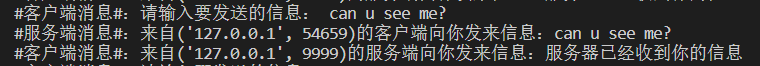


客户端发送：hello,im,li

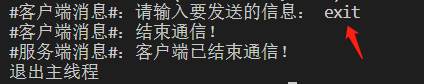
服务端接受来自端口54659客户端发来的消息

客户端接收服务端确认收到消息的通知，握手完成





Exit退出会话



1. 结语

Openssl在kali linux环境下完成配置，学习到了许多openssl命令，同时深入理解了各个实验的原理及实现。

实验中逐步实现从颁发证书并通过对指定端口实现监听完成SSL通信方面的安全让我更加确信实践才是检验知识的唯一标准，本次实验提高了动手实践能力，激发了学习安全技术的兴趣，在以后的学习生活中要多加练习和实践，将理论转化为应用。