**北京科技大学实验报告**

学院：计通学院 专业：信息安全 班级：信安182

姓名： 梁逸飞 学号：41824141 实验日期： 2020年 12月 18日

**实验名称：基于JWT的安全认证实验**

**实验目的：**通过JWT安全认证实验深入了解安全认证的方法与思想。

**实验原理：**Json web token (JWT), 是为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准。JWT的声明一般被用来在身份提供者和服务提供者间传递被认证的用户身份信息，以便于从资源服务器获取资源，也可以增加一些额外的其它业务逻辑所必须的声明信息，该token也可直接被用于认证，也可被加密。

基于token的鉴权机制类似于http协议也是无状态的，它不需要在服务端去保留用户的认证信息或者会话信息。这就意味着基于token认证机制的应用不需要去考虑用户在哪一台服务器登录了，这就为应用的扩展提供了便利。

**实验环境：**pycharm、python3.8

**实验内容与步骤：**

一、了解JWT安全认证机制

此部分单独制作了说明文档，详请请见“JWT说明文档.docx”说明文档。

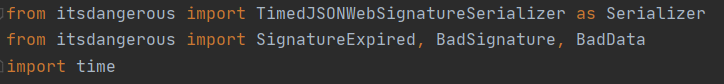
二、基于python实现群签名

将通过一个python程序来实现JWT安全认证的过程，将定义两个函数用于模拟服务器对JWT的签发和认证，定义主函数来模拟客户端向服务器发出请求。

1、导入库信息

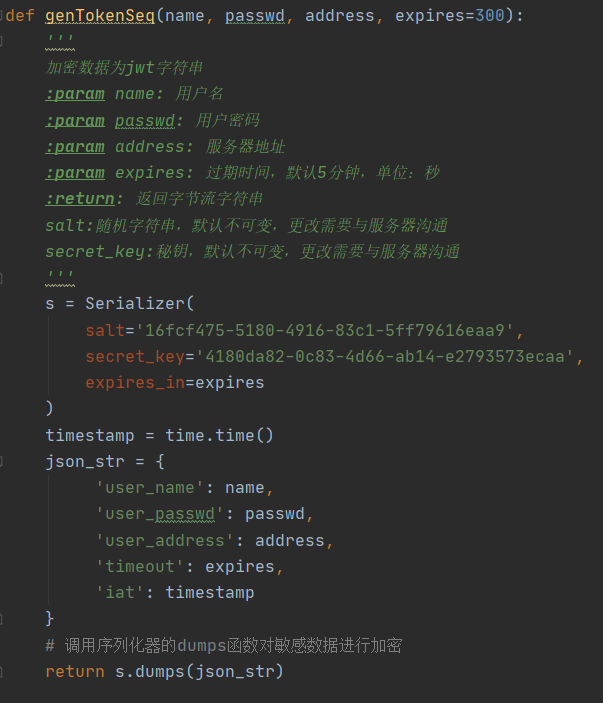
time库是用于同步时间和规定时间戳的工具库。

itsdangerous是python中jwt库的升级版，将利用其库函数实现JWT的安全认证。



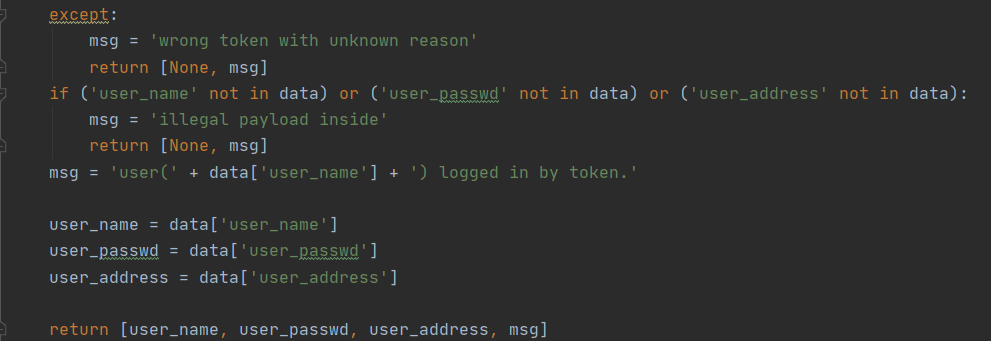
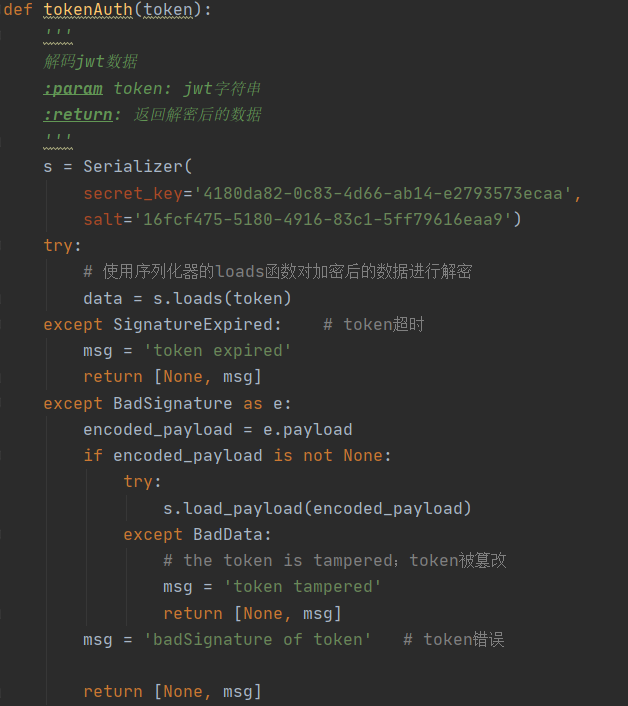
2、签发函数

定义genTokenSeq()函数用于服务器端签发生成Token序列。在该函数里，利用itsdangerous库里的函数Serializer生成一个序列化器，内部设置第三部分签名所需要用的盐和密钥，其中盐和密钥可以自行设置，但是不可公开。通过此设置，已经内置了JWT的头部。然后是自行设计定义payload部分，在此设计为用户名、用户名密码、服务器地址、过期时间以及时间戳，通过传入的参数进行初始化。最后调用dump()函数，将设置的payload传入进去，序列化器会将payload结合内置的头、以及签名部分分别进行base64加密再连接，最终返回一个Token序列。代码实现如下。



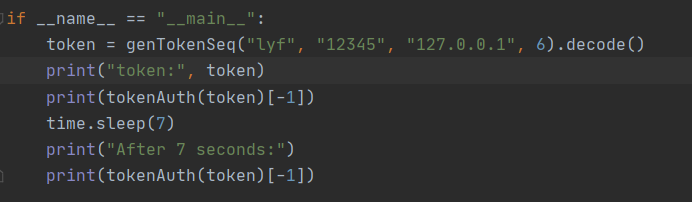
3、认证函数

定义tokenAuth()函数用于服务器端对token序列的验证，传入的参数即为token。同时也要设置序列化器，其盐和密钥的设置与签发函数中的设置一样，否则就会导致验证不成功。通过loads函数对序列token进行解密获取其中的payload信息，然后对信息进行判断验证，返回判断的信息。实现代码如下。



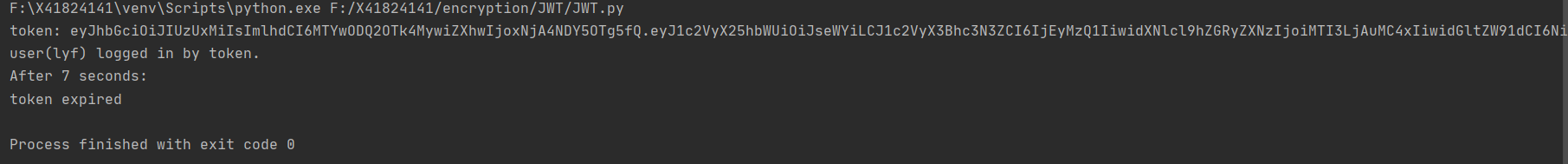
4、主函数

主函数模拟客户端，在主函数调用genTokenSeq函数模拟客户端向服务器请求签发token序列，设置的过期时间为6s。然后再通过调用tokenAuth函数用于模拟客户端向服务器请求数据时服务器对客户端token序列的认证。延时7s再进行一次调用模拟token过期后的认证结果。实现代码如下。



5、测试运行

运行结果如下图所示。



说明JWT认证功能已经成功实现。

三、docker部署与镜像封装

此部分单独制作了说明文档，详情请见“镜像生成说明文档.docx”说明文档

**实验结果与分析：**

上述代码通过python，模拟客户端与服务端的核心功能，即token序列的签发和认证，能够实现基于JWT的安全认证。通过此实验深刻了解了JWT的认证流程以及其优点，认识了安全认证的另一先进技术。

其完整代码见JWT.py。