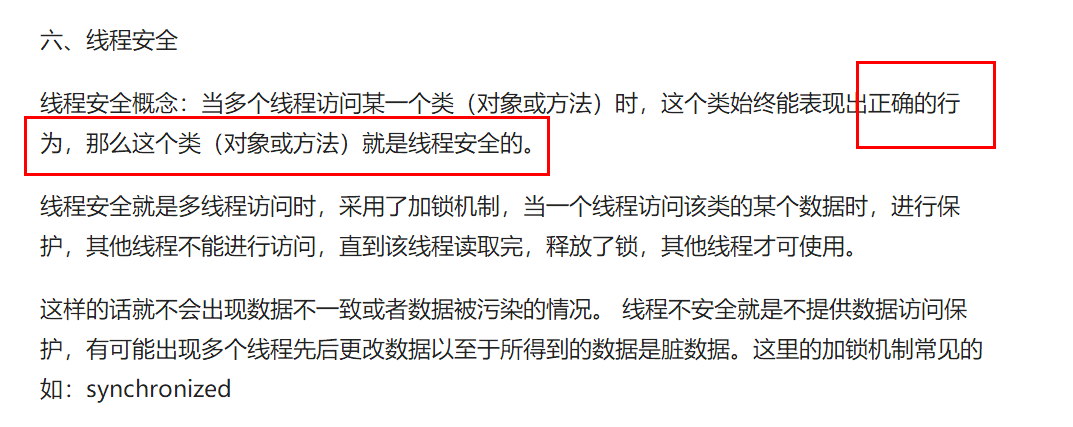
# details

##### 报错：mybatis传入List后<if>标签判断是否为空要加上长度的判断



##### 线程安全和线程不安全



##### 待定

# Springboot

# springCloud

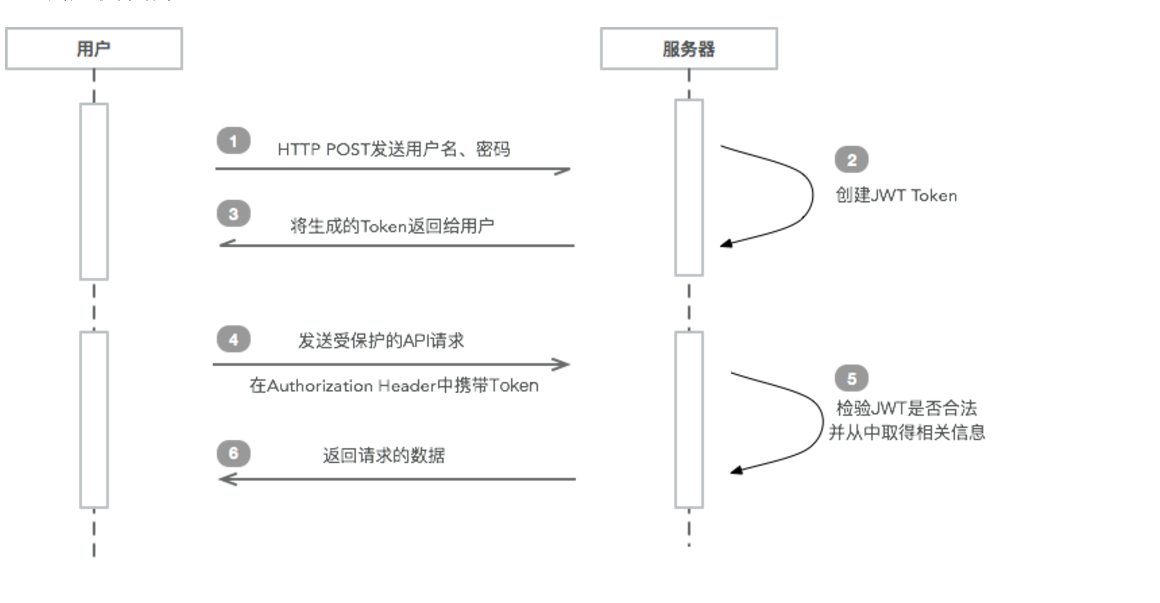
# Mybatis

# Java

# 鉴权

https://github.com/wpcfan/spring-boot-tut/blob/master/README.md

#### 1.基本概念：JWT是 Json Web Token的缩写



#### 2. 也就是说JWT是由三段组成的，按官方的叫法分别是header（头）、payload（负载）和signature（签名）

一个JWT实际上就是一个字符串，它由三部分组成，头部、载荷与签名。  
载荷（Payload）

{ "iss": "Online JWT Builder",

"iat": 1416797419,

"exp": 1448333419,

"aud": "www.example.com",

"sub": "jrocket@example.com",

"GivenName": "Johnny",

"Surname": "Rocket",

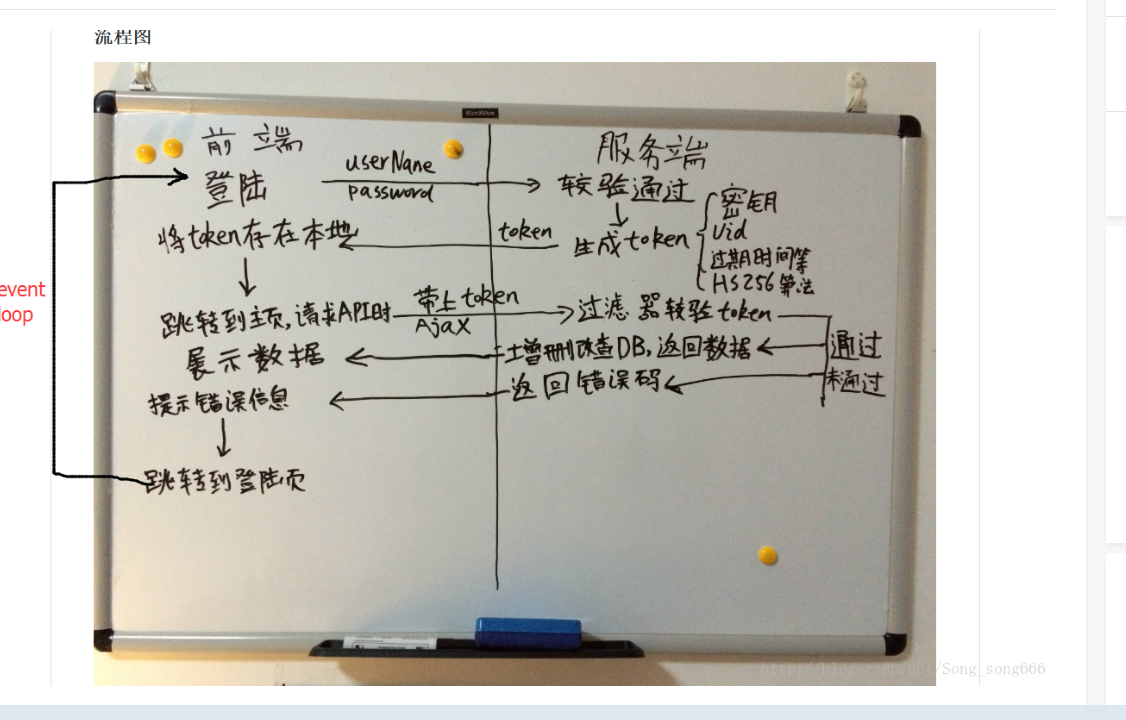
"Email": "jrocket@example.com",

"Role": [ "Manager", "Project Administrator" ]

}

* iss: 该JWT的签发者，是否使用是可选的；
* sub: 该JWT所面向的用户，是否使用是可选的；
* aud: 接收该JWT的一方，是否使用是可选的；
* exp(expires): 什么时候过期，这里是一个Unix时间戳，是否使用是可选的；
* iat(issued at): 在什么时候签发的(UNIX时间)，是否使用是可选的；  
  其他还有：
* nbf (Not Before)：如果当前时间在nbf里的时间之前，则Token不被接受；一般都会留一些余地，比如几分钟；，是否使用是可选的；

#### 3.登录流程图



1.将荷载payload，以及Header信息进行Base64加密，形成密文payload密文，header密文。

2.将形成的密文用句号链接起来，用服务端秘钥进行HS256加密，生成签名.

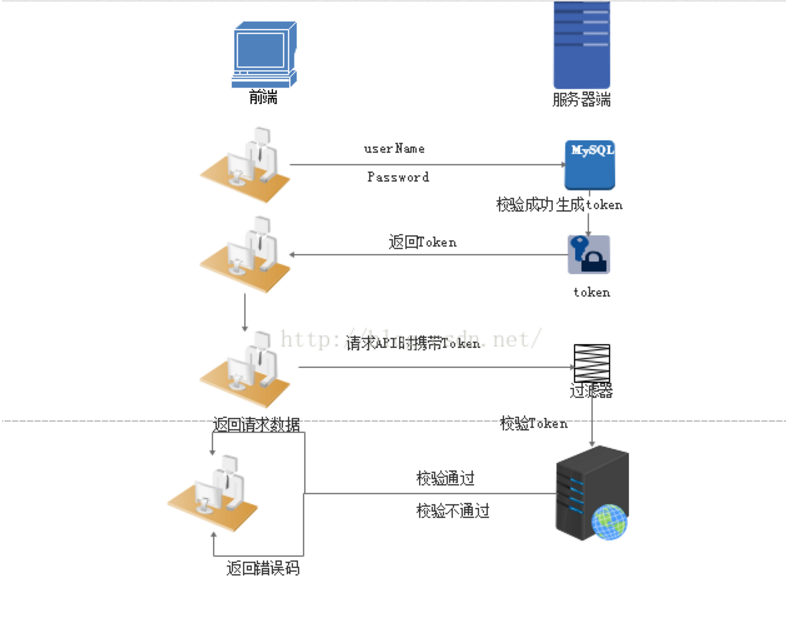
3.将前面的两个密文后面用句号链接签名形成最终的token返回给服务端

注：

（1）用户请求时携带此token（分为三部分，header密文，payload密文，签名）到服务端，服务端解析第一部分（header密文），用Base64解密，可以知道用了什么算法进行签名，此处解析发现是HS256。

（2）服务端使用原来的秘钥与密文(header密文+"."+payload密文)同样进行HS256运算，然后用生成的签名与token携带的签名进行对比，若一致说明token合法，不一致说明原文被修改。

 （3）判断是否过期，客户端通过用Base64解密第二部分（payload密文），可以知道荷载中授权时间，以及有效期。通过这个与当前时间对比发现token是否过期。



#### 4.