📒 JWT学习笔记

背景

了解用户身份认证的几种方式

一、session验证

1、客户端使用用户名和密码请求登录

2、服务端接收请求，验证信息，成功后，在当前对话（session）里面保存相关数据，比如用户角色、登录时间等等

3、服务器向用户返回一个 session_id，写入客户端的 Cookie

4、往后客户端每次向服务端请求资源时，都会通过 cookie，将 session_id 传回服务器。

5、服务端收到 session_id，找到前期保存的数据，由此得知用户的身份，如果验证成功，正常返回客户端响应数据

二、token验证

1、客户端使用用户名和密码请求登录

2、服务端接收请求，验证信息，成功后，服务端返回一个token

3、客户端收到token后可以将其存储起来，例如cookie或localStorage

4、往后客户端每次向服务端请求资源时都需要携带这个token，可以通过cookie或者header携带

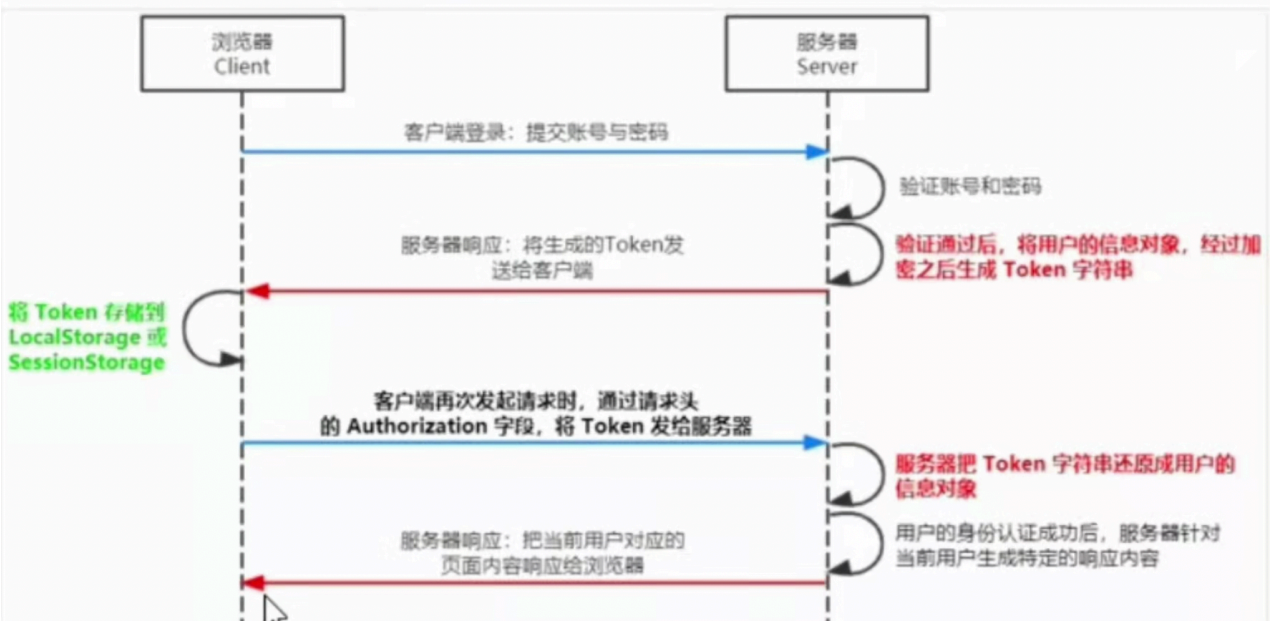
5、服务端接收请求，验证token，如果验证成功，正常返回客户端响应数据

session	**数据存放在服务器端**	**服务器端资源开销大** **存储时间短，需要服务器设置；** **服务器如果挂掉了需要重新登录**
方式	特点	缺点
token	节约服务器资源	CPU消耗大

JWT就是token验证方式的一种具体实现方式

工作原理

JWT：JSON Web Token，当前最流行的跨域认证解决方案



总结：用户的信息通过Token字符串的形式，保存在客户端浏览器中，服务器通过还原Token字符串的形式来认证用户的身份

组成部分

通常由三部分组成：Header（头部）、Payload（有效荷载）、Signature（签名）

格式：Header.Payload.Signature

其中payload： 真正的用户信息，加密后的字符串；header和signature：安全相关的部分，保证token的安全性

使用方式

客户端收到服务器返回的JWT后，通常会将它存储在localstorage或sessionstorage中

此后，客户端每次与服务端通信，都要带上这个JWT的字符串，从而进行身份认证，推荐的做法是把JWT放在HTTP请求头的Authorization字段中，格式如下：

Authorization： Bearer

Portal项目中的实现

portal项目中的实现借鉴了JWT的原理，在具体实现上做了调整

第一步：portal端跳转外部页面之前总是会先以当前用户信息作为参数调用“更新token”的接口，后端返回一个携带token的url地址供前端跳转

```
const linkClick = () => {
  const { loginCustNo = '', token = '' } = getUserInfo() || {};
  if (opacity === 1) {
    setOpacity(0.5);
    props.url &&
      props.freshJWT &&      lihui2018, 3 months ago • typescript
      props
        .freshJWT({
          params: {
            loginCustNo,
            userAgent: navigator.userAgent,
            token,
            route: props.url.substr(0, props.url.indexOf('?')),
          },
          apiName: 'freshJWT',
        })
        .then(() => {
          setOpacity(1);
        });
    window.open(props.ssoUrl || props.url, '_blank');
  }
};
```

第二步：alpha端从url中拿到token值，调用token校验接口判断是否认证通过，如果通过则更新userInfo客户信息

```
const { lvc: jwtToken } = parse(window.location.search) || {};
if (jwtToken) {
  const result = await request(`${ssoHost}/pmserver/portal/jwt/checkJWT`, {
    method: 'POST',
    data: {
      userAgent: navigator.userAgent,
      route: window.location.origin + window.location.pathname,
      jwt: jwtToken,
    },
  });
  console.log('Portal JWT Result:', result);
  if (result?.data?.data?.ssoToken) {      ludejun, 4 months ago • portal跳转
    // Store 设置
    props.dispatch({
      type: 'login/changeState',
      payload: {
        userInfo: {
          ...props.userInfo,
          ...result.data.data,
          custno: result.data.data.custNo,
        },
      },
    });
    setUserInfo({
      ...props.userInfo,
      ...result.data.data,
      custno: result.data.data.custNo,
    });
    updateMenuList(); // portal跳转过来重新加载menuList
  }
```

JWT的优势

JWT的优势
1、JWT token数据量小，传输速度快
2、不需要在服务端保存信息，不依赖于cookie和session，易于应用的扩展
3、单点登录友好（cookie无法跨域难以实现单点登录）