前端权限认证逻辑梳理

一．参考项目

1. Vue-element-admin:经典权限管理项目，有文档，项目结构清晰，但年代稍久，存在着一些已经弃用的函数和很久的包。

2. Vue-admin-better:功能完善，比较新，但项目内广告太多，且使用了一个因安全原因被git禁用的包。

3.OHRP:公司项目，参考了Vue-admin-better项目。

二．路由权限认证

1.总体思路：见draw.io附件

2.核心要点：

（1）登录时用户名和密码加密传输

（2）提交用户名和密码后服务器返回token

（3）token本地存储：session（OHRP），cookie（Vue-element-admin），localstorage（Vue-admin-better）

（4）用户进行页面跳转前通过全局前置路由守卫上传token获取权限，之后通过权限生成动态路由

（5）动态路由列表的生成有两种方法，可在前端生成也可在后端生成后再传输给前端

3.实现思路

（1）动态路由（包括前端和后端生成两种方式）：现阶段流行的方式，但较为复杂

1. 静态路由：简单，不太容易出bug，但比较笨重

4.存在的问题：

（1）按照参考项目的实现方式，当后端更改用户权限后，用户需要进行一次刷新或者退出后再重新登录，页面才会根据相应的权限作出改变。Mqtt。

（2）由于无法操作范例项目的后端所以无法测试权限改变时页面的变化。

（3）前端权限管理页面的实现未知。

三．组件权限认证

1.不完善的总体思路：通过自定义指令或者v-if实现

2.核心要点：

（1）部分element-ui组件只能通过v-if来控制是否显示

3.存在的问题：

（1）相较于路由权限认证，范例项目只包含了最简单基本组件权限认证，无法满足项目需求。还需要参考其他项目或者组内自己讨论出一套解决方案。

四．具体实现

在实际开发中，要实现上述的路由及组件权限认证大致有三种方式。

1. copy已有的项目（个人推荐Vue-element-admin），之后删除掉所有不需要的页面，只保留项目最核心最通用的骨架，并结合具体的需求稍作修改。
2. 按照范例项目的思路从头开始搭建项目，包括路由、Vuex等。
3. 从头开始搭建项目，但不采用动态路由。