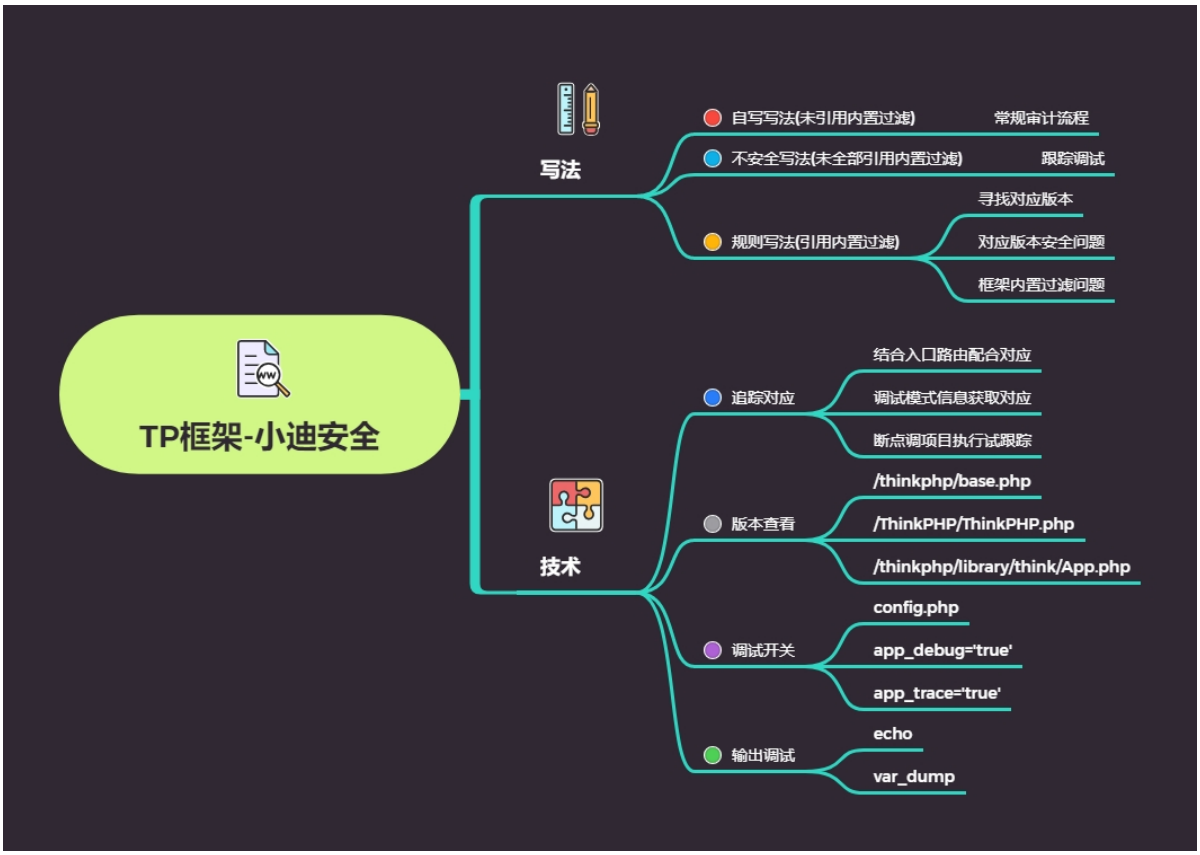
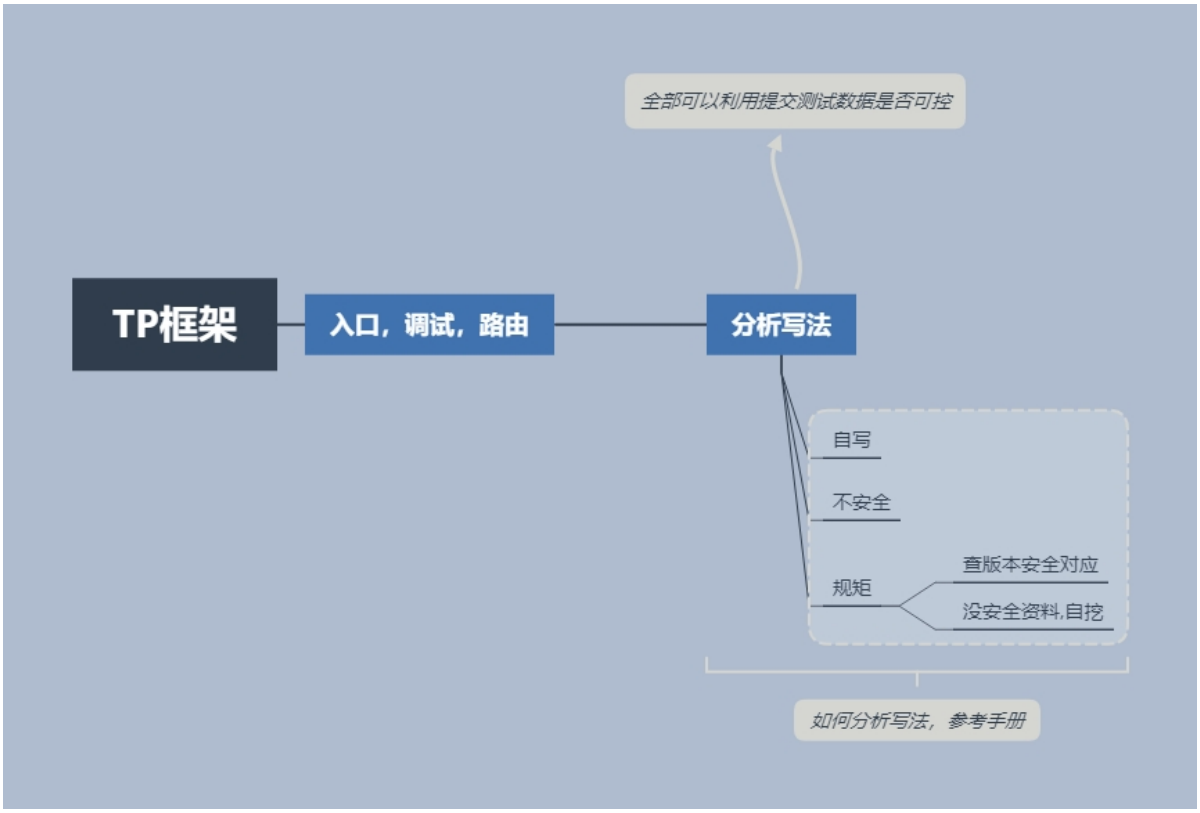
Day54　代码审计–TP5 框架审计写法分析及代码追踪





```
#知识点1：调试，访问，路由，配置，版本等等

#知识点2：自写写法，不安全写法，规则写法

#知识点3：
---调试模式信息获取对应
---结合入口路由配合对应
---项目断点调试执行跟踪
```

54.1 demo 代码段自写和规则写分析

```php
<?php
    namespace app\index\controller;
    use think\Controller;
    use think\Db;


class Test extends Controller
{
    public function x()
    {
        echo 'x test';
    }

    public function testsqlin()
        {
            //自写数据库查询，存在注入
            $id=$_GET['x'];

 $conn=mysql_connect("127.0.0.1","root","root");
```

```
19              $sql="select * from injection.users
    where id=$id";
20              echo $sql;
21              $result=mysql_query($sql,$conn);
22          }
23
24      public function testsqlin1()
25          {
26              //table('users')->where('id',1)-
    >select();
27              $id=$_GET['x']; db('users')-
    >where('id',$id)->select();
28          }
29
30      public function index()
31          {
32              $username = request()->get('id/a');
33              db('users')->insert(['id' =>
    $username]);
34              return 'Update success';
35          }
36  }
37  />
```

对于自写写法，可以采用常规注入方法进行注入

对于半自写，半官方的写法，打开config配置文件开启调试模式，观察数据库的操作语句，如果还是没有观察到mysql语句，可以使用专门的mysql的检测工具进行监控（MySQL-Monitor）

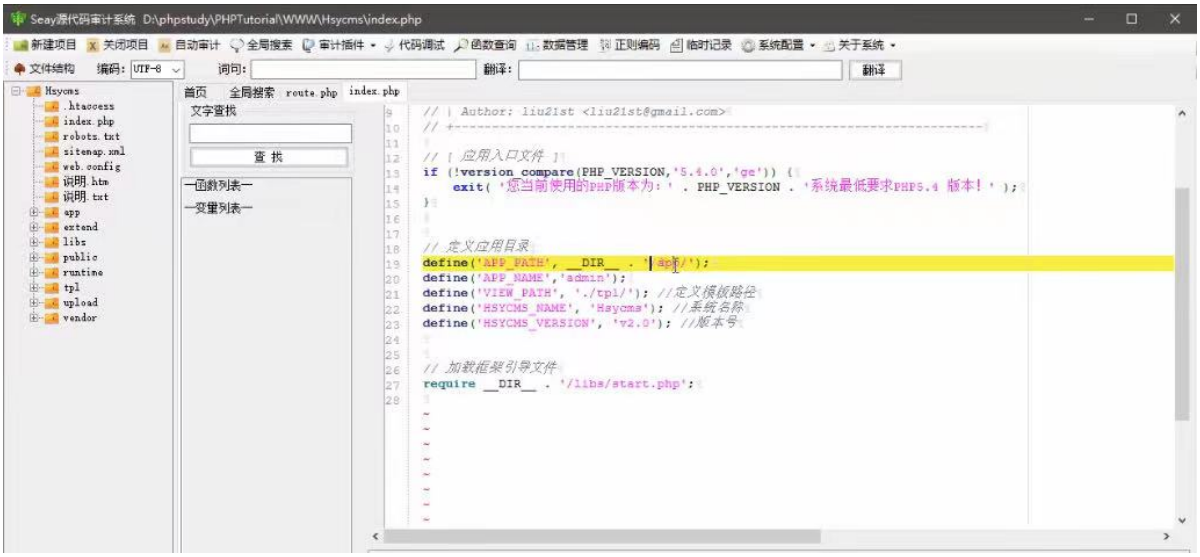
对于规则写法，可以查看该thinkphp的版本，查看对应漏洞，如果没有对应漏洞，则需要自己进行审计

```
https://github.com/Mochazz/ThinkPHP-Vuln
https://github.com/top-think/framework
```

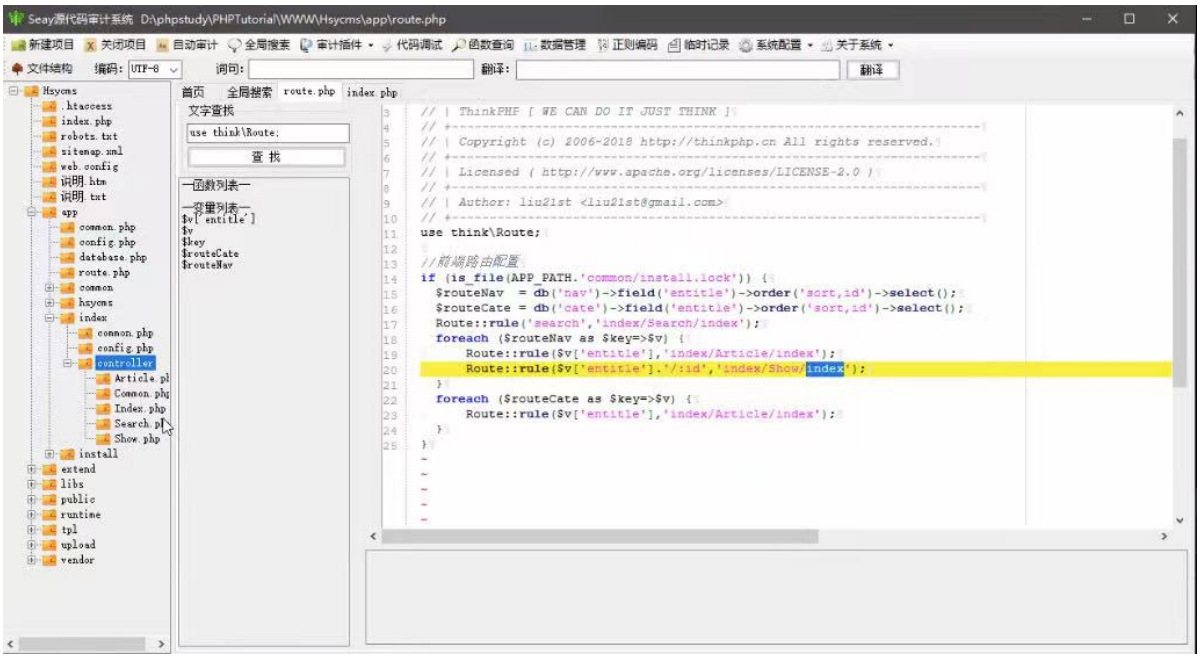
54.2 hsycms-TP 框架-不安全写法-未过滤

进入靶场：

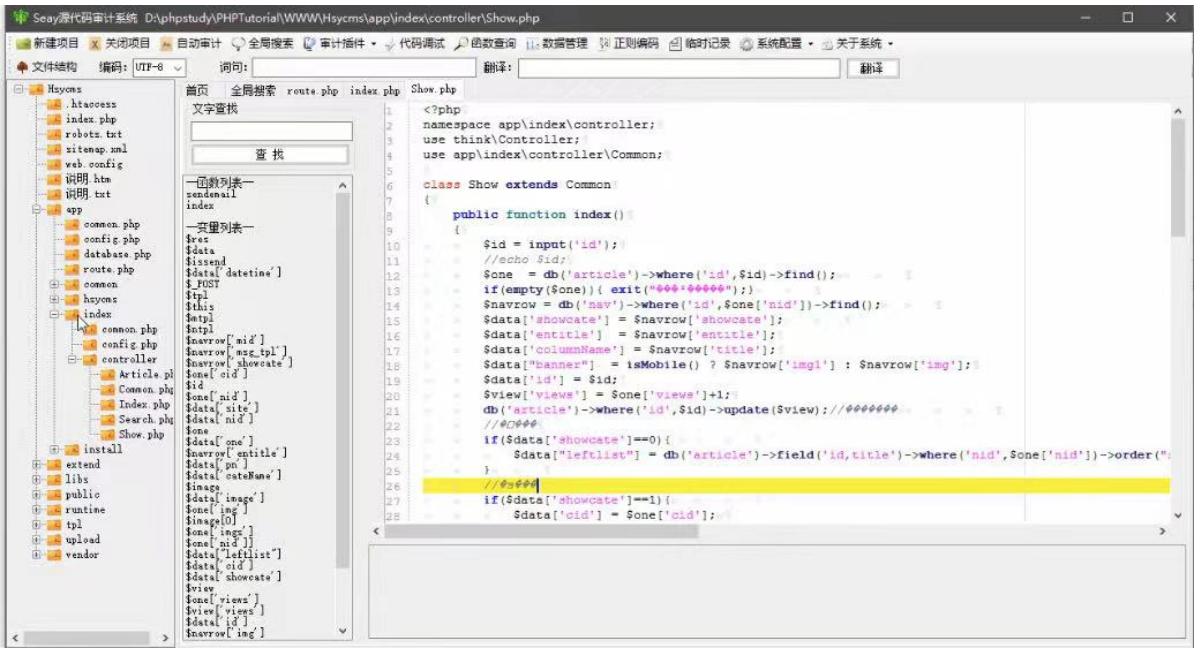
发现url地址和官方提供的地址不同，因此进行了路由配置，首先查看入口文件：



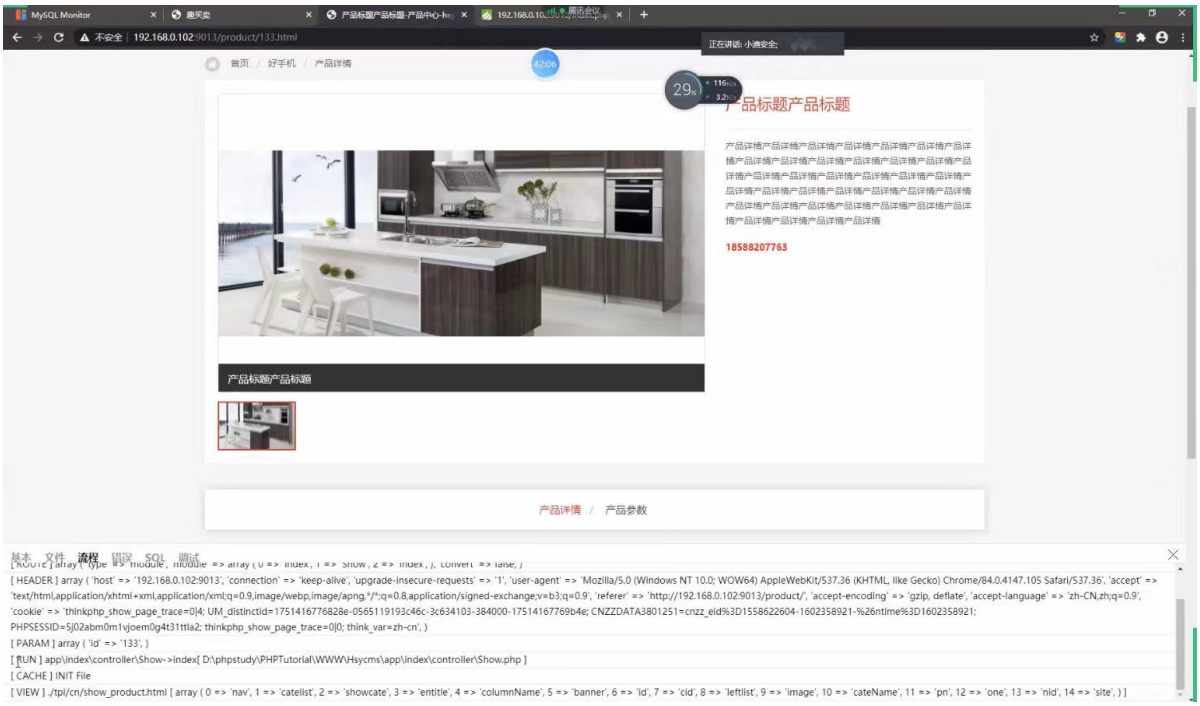
其次，由于进行了路由配置，因此还需要查看路由配置文件：



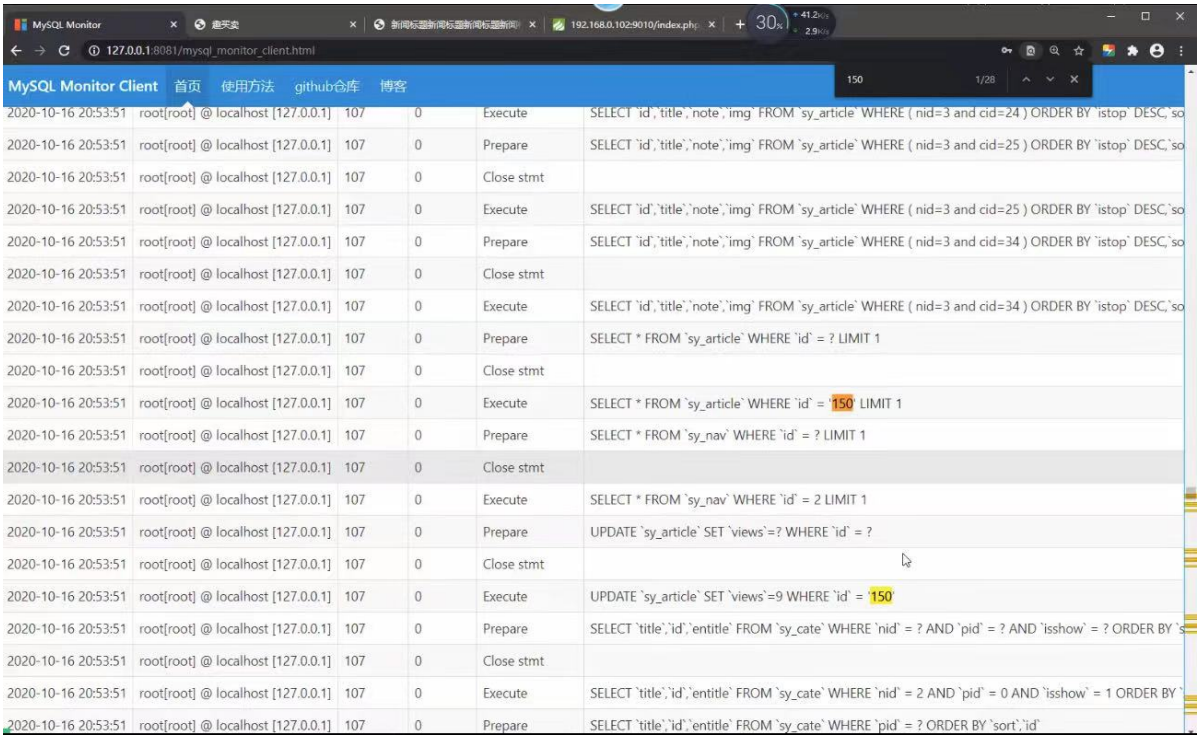
查看对应的路由配置，找到真正的文件地址：



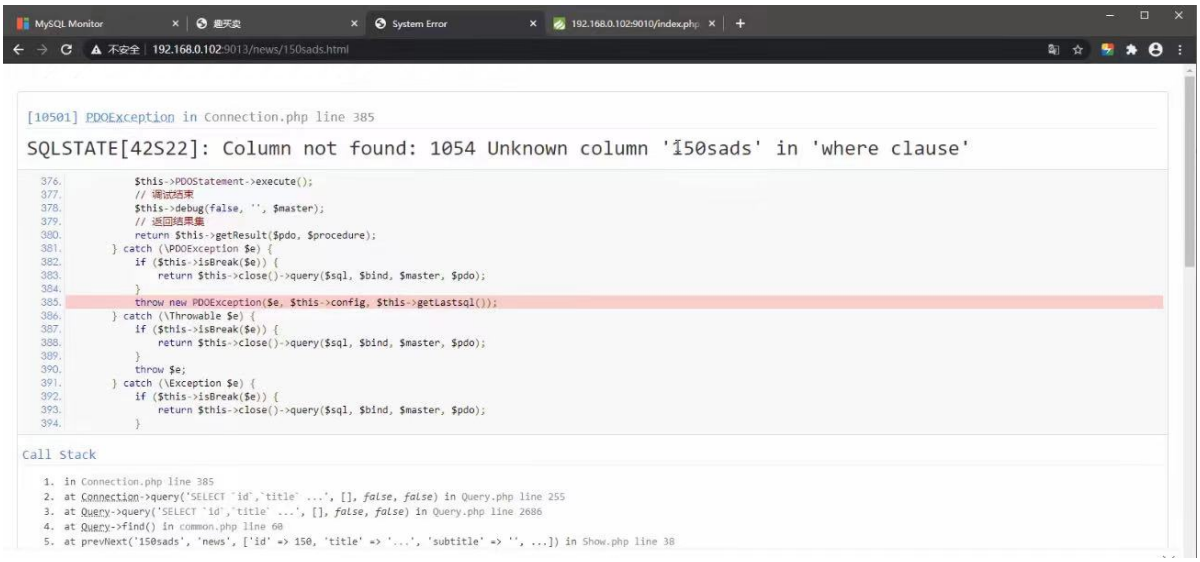
还可以通过调试的方法，在流程中观察文件执行的顺序，找到run模块，找到url的真实路径，但是不能找到该文件下具体执行的函数：



通过路由配置文件可知133为id，在接受该参数发现与官方文档不一致怀疑可能存在注入，点击页面不同的模块发现该参数会变化，但是在sql中并没有发现sql执行的语句，此时需要MySQL-Monitor监控，在该页面发现了sql执行的语句：

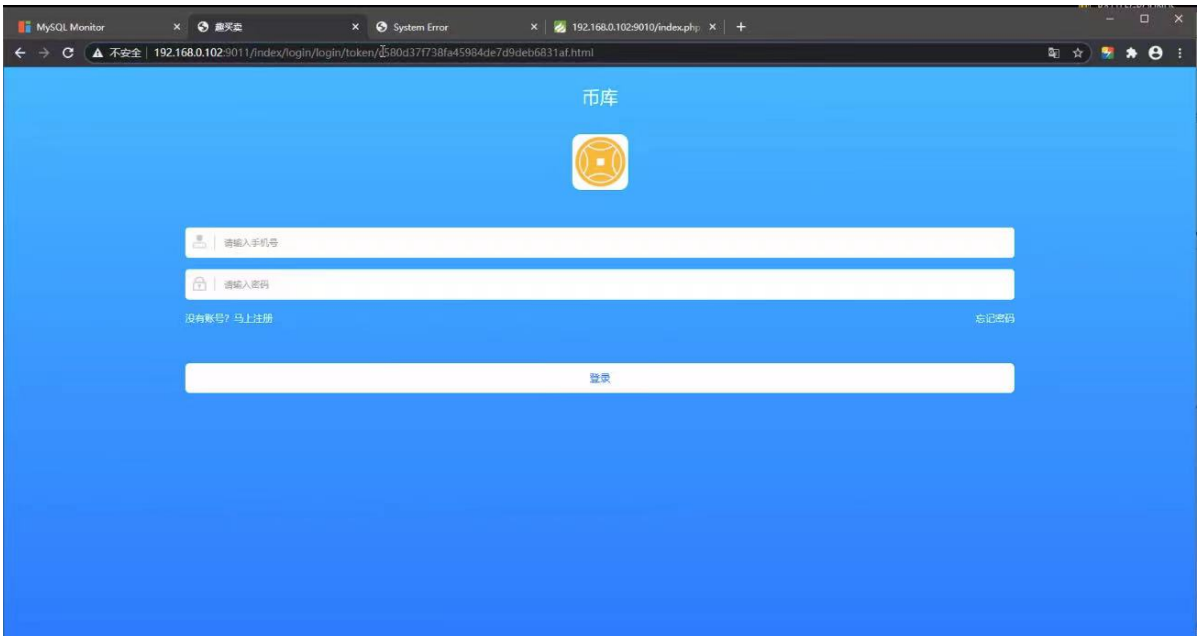


对url中的id参数进行改造，发现存在注入：



54.3 weipan21-TP 框架-规则写法-内置过滤

进入该程序首页，发现url地址没有进行路由处理：



经过审计发现没有漏洞，此时需要先知道该框架的版本，然后查询专门的框架漏洞平台，查找对应漏洞

结论：对于自己审计的代码框架如果没有漏洞，可以去查看官方的安全更新，对比前后两个版本的代码有何差异从而查找相关的框架漏洞

资源：

```
https://github.com/top-think
```