**前言：**

**环境python3.11.2（vscode）**

**需要有selenuim-4.22.0，urllib3-2.2.2等依赖（新老版本语法存在区别）**

浏览器驱动版本 edge 126.0.2592.87 (正式版本) (64 位),其他版本需求请前往官网自行下载

**源码上传github（待完善）**

**地址:**

**首先，这个项目是尝试写sqlmap时改造的，所有侧重点是对单个参数进行检测，没有全页面扫描定位功能**

**其次对于xss注入，只实现了get传参的检测，因为根据推理requests模块是不能完成xss检测的，其返回的前端代码对于反射性xss代码无法捕获，而存储型代码即使前端回显也不一定能成功，所有检测方式应当利用弹窗，明显requests是不能拿到弹窗的，这时我们引入了selenuim模块，这样就能拿弹窗，但selenuim没有post传参的封装，就算是get传参也是写在url里面带过去的，不给把seleuim和requests结合应该可以实现，但截至目前为止我没用实现。而且selenuim需要在项目里引入对应浏览器版本的驱动器，而且不同浏览器语法也有一定区别，可能兼容性不是很大，当然可以通过if判断来支持多种浏览器，但实话就是我不想搞**

**最后，就像开头所说，我这个本来是一个sqlmap，所以实现了一个半成品的sql报错注入检测，支持post和get因为requests很轻松能实现两种方式的互转**

**思路：**

**就是通过post，get的方式将参数传过去，在想办法拿到对应的回复，检查其中是否有我们想要的值**

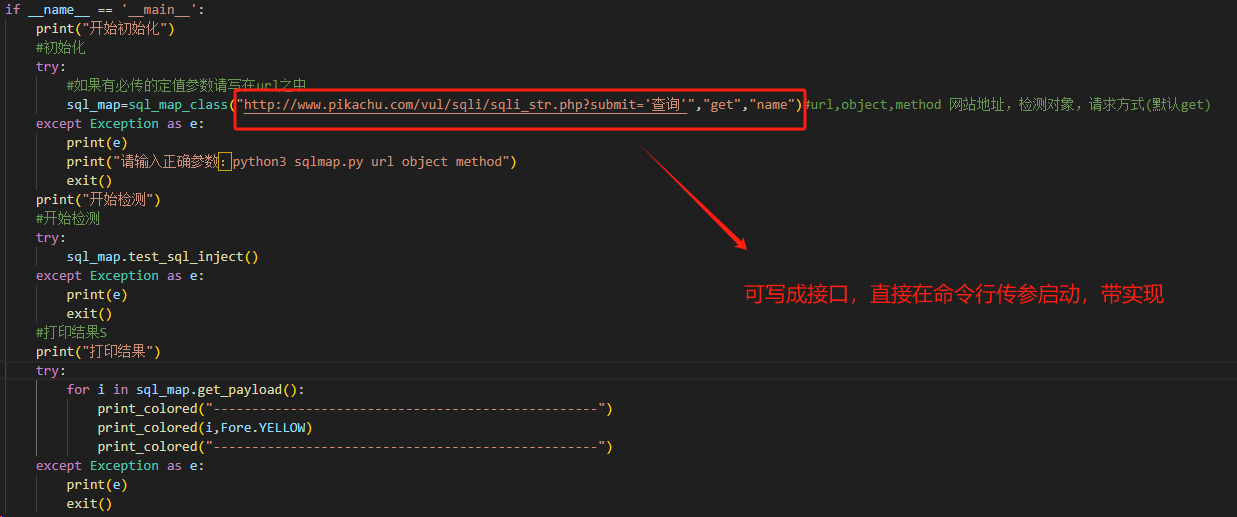
**比如sql注入时或xss弹窗时，我们将MD5（zx）的值作为payload的一部分，报错注入或弹窗的回显就会证明确实触发了相应的漏洞**

**（报错注入会有可能只是在前端展示，但并不是想要的报错，不过无论是updatexml还是extractvalue都会把前面的！或~一起带上，所以判断时应该是！+MD5（zx）的前32位，因为他们只显示前32位。，同样的存储型xss如果只是单纯判断前端页面是否有相应代码也是不行的，有时候回显的代码是处理过的，你能看见不代表能执行，所以我才会使用弹窗判断，只要能捕获到弹窗，且弹窗内容是对应的MD5值那就肯定是执行了，当然还有一总思路就是通过执行ajax向python这边传数据，如果接收到数据说明ajax执行了，那就是xss漏洞没跑了,但是这样要改大量的payload，太多了不想改了）**

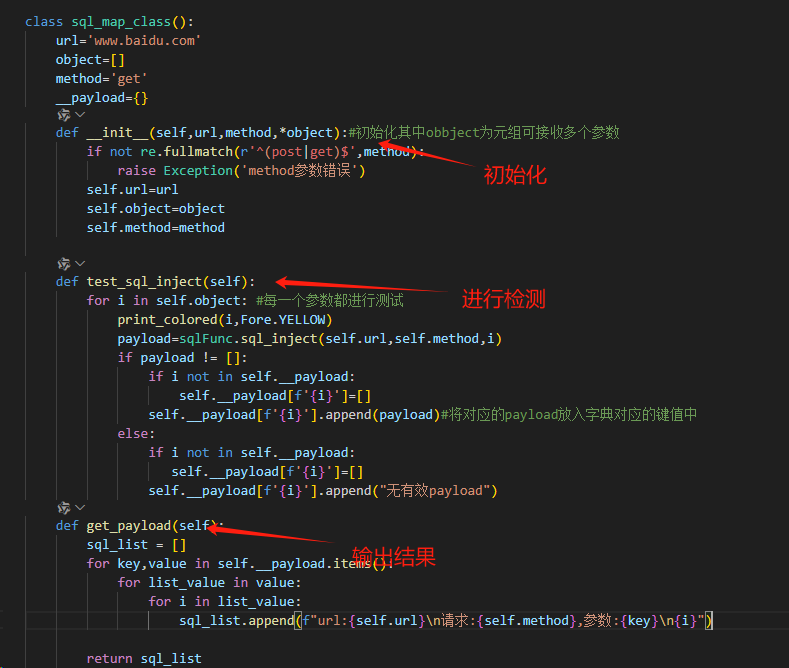
**结构：**

**主函数main传递参数url method object**

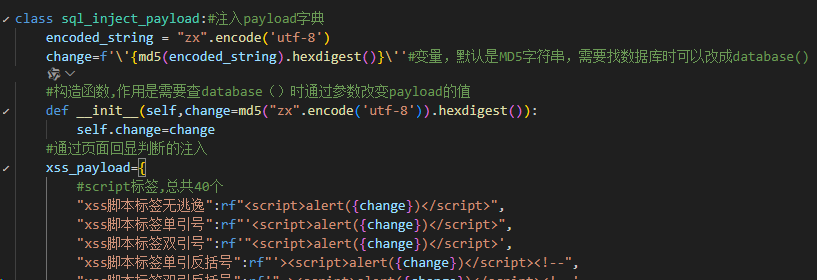
**分别为访问的网址url，参数提交的方式，参数名字**

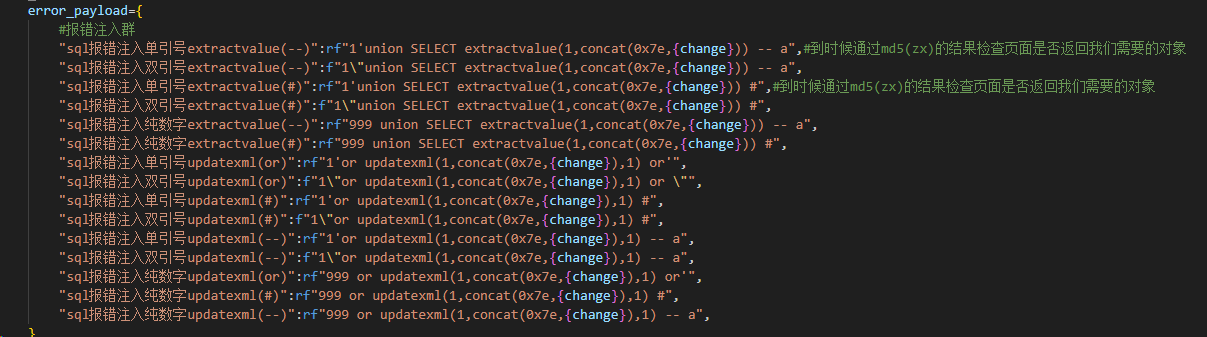


**我们需要一个类对这些参数进行实例化和控制，比如输出最后的payload合集，作为中转向实际工作函数传递参数**

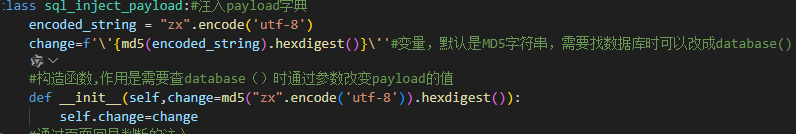


**同时写一个sql（xss）类用来调用函数，和存储payload方便调用**



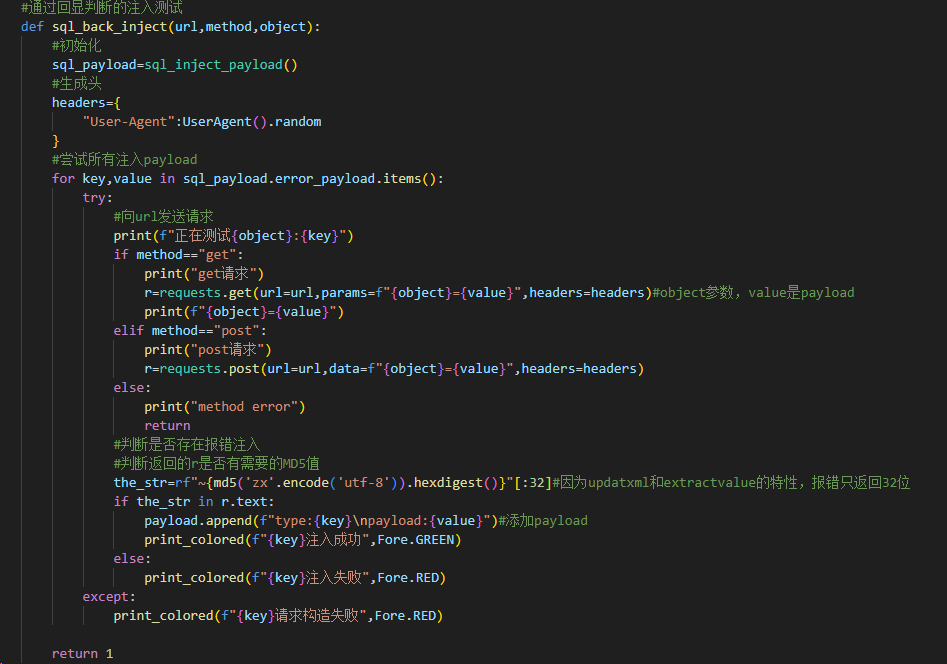


**为什么这里的MD5要写成可变参数呢，因为在做sqlmap时想的是到时候把MD5替换为database（）就能拿数据库名**



**将实际检测的函数单独封装在一起（实话是我没有封装，不过是单独成一页的）方便管理，还能加强代码保密性**

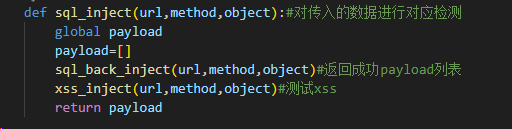
**关于报错注入部分requests模块**



**关于xss部分，selenuim模块(这个很慢，所有100多条payload我注释了八十条，方便测试，按道理说可以用多线程来同时执行，但selenuim是浏览器操作我怕搞不好起一百多个页把电脑搞崩，当然这里用的是无头模式不会打开浏览器，但我感觉没区别)**



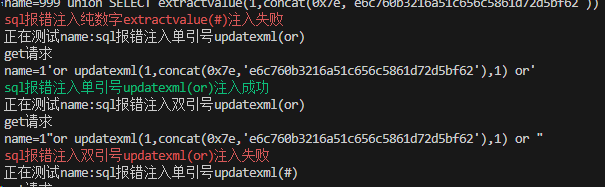
**总控函数，所有测试函数都在这里调用，方便管理。按道理说这里也可以用多线程来分别执行，不过这就要考虑插入payload列表时的冲突问题了**



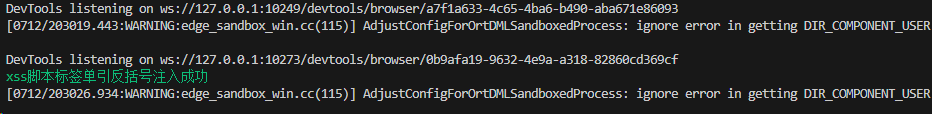
**全局变量payload列表，让测试函数只要判断对了就可以直接往里面插，省去了返回值处理的麻烦**

**效果展示**

**sql报错注入运行展示**



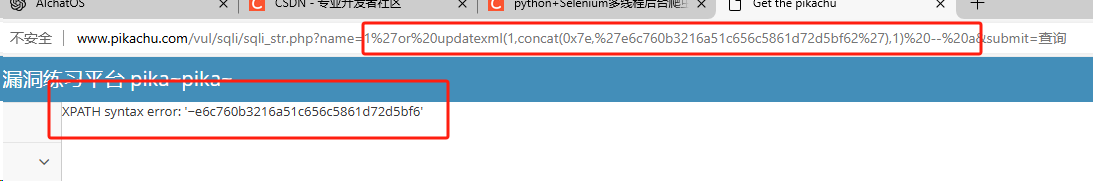
**Xss注入展示**



**结果展示**



**这些结果真实有效，前往pikachu测试确实有用**



**而且我扫的参数还不是xss靶场的，是sql注入靶场的所以证明这代码还是具有一定通用性的**

