

HGAME 2022 Week2 writeup by D3ic1de

HGAME 2022 Week2 writeup by D3ic1de

Web

Git Leakage

V2Board

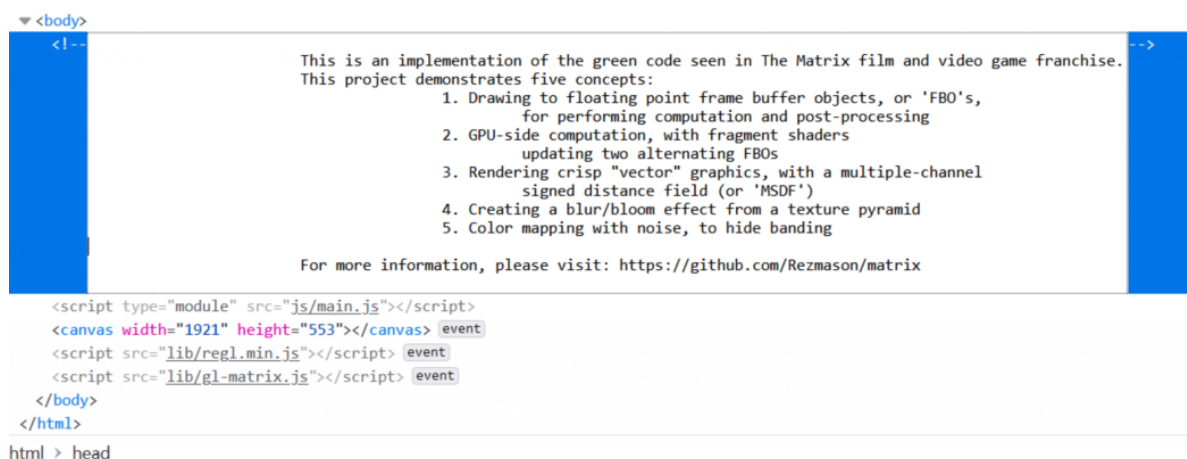
Search Commodity

Web

Git Leakage

开靶机进去之后就是一片绿色的代码流

然后没啥好动的，F12看看源码，发现有注释，感觉似乎有用



```
<!--
This is an implementation of the green code seen in The Matrix film and video game franchise.
This project demonstrates five concepts:
  1. Drawing to floating point frame buffer objects, or 'FBO's,
     for performing computation and post-processing
  2. GPU-side computation, with fragment shaders
     updating two alternating FBOs
  3. Rendering crisp "vector" graphics, with a multiple-channel
     signed distance field (or 'MSDF')
  4. Creating a blur/bloom effect from a texture pyramid
  5. Color mapping with noise, to hide banding

For more information, please visit: https://github.com/Rezmason/matrix
-->

<script type="module" src="js/main.js"></script>
<canvas width="1921" height="553"></canvas>
<script src="lib/regl.min.js"></script>
<script src="lib/gl-matrix.js"></script>
</body>
</html>
```

然后去那个github仓库看了看，emm没啥用，就是这个项目的创作过程和一切其他的不同样式的代码流。没啥思路。。。

又回来看看题目Git? Leakage(泄露)? (度娘救我!)

Git简介

官方给出的解释是：Git是一个开源的分布式版本控制系统,我们简单的理解为Git 是一个内容寻址文件系统，也就是说Git 的核心部分是键值对数据库。当我们向 Git 仓库中插入任意类型的内容(开发者们在其中做的版本信息修改之类的操作)，它会返回一个唯一的键，通过该键可以在任意时刻再次取回该内容

Git是一个可以实现有效控制应用版本的系统，但是在一旦在代码发布的时候，存在不规范的操作及配置，就很可能将源代码泄露出去。那么，一旦攻击者或者黑客发现这个问题之后，就可能利用其获取网站的源码、数据库等重要资源信息，进而造成严重的危害。

想要确定是否存在这个漏洞，可以通过以下方式。首先是看看有没有**提示**醒目地指出 Git，如果有就考虑存在。如果没有也可以使用 **dirsearch** 工具扫描后台，如果存在则会扫描出 .git 目录如图所示。

那就后端扫描一下

```
Windows PowerShell
Target: http://week-2.hgame.lwsec.cn:32427/

[10:31:48] Starting:
[10:31:49] 302 - 0B - /js -> /js/
[10:31:52] 302 - 0B - /.git -> /.git/
[10:31:52] 200 - 23B - /.git/HEAD
[10:31:52] 200 - 25B - /.git/COMMIT_EDITMSG
[10:31:52] 200 - 259B - /.git/config
[10:31:52] 200 - 73B - /.git/description
[10:31:52] 302 - 0B - /.git/logs/refs -> /.git/logs/refs/
[10:31:52] 200 - 240B - /.git/info/exclude
[10:31:52] 200 - 625B - /.git/logs/HEAD
[10:31:52] 200 - 65KB - /.git/branches/
[10:31:52] 200 - 68KB - /.git/
[10:31:52] 200 - 625B - /.git/logs/refs/heads/master
[10:31:52] 302 - 0B - /.git/logs/refs/remotes/origin -> /.git/logs/refs/remotes/origin/
[10:31:52] 200 - 167B - /.git/logs/refs/remotes/origin/HEAD
[10:31:52] 200 - 41B - /.git/refs/heads/master
[10:31:52] 302 - 0B - /.git/refs/remotes/origin -> /.git/refs/remotes/origin/
[10:31:52] 302 - 0B - /.git/logs/refs/heads -> /.git/logs/refs/heads/
[10:31:52] 302 - 0B - /.git/logs/refs/remotes -> /.git/logs/refs/remotes/
[10:31:52] 302 - 0B - /.git/refs/heads -> /.git/refs/heads/
[10:31:52] 302 - 0B - /.git/refs/remotes -> /.git/refs/remotes/
[10:31:52] 200 - 584B - /.git/packed-refs
[10:31:52] 200 - 21KB - /.git/index
[10:31:52] 200 - 68KB - /.git/hooks/
[10:31:52] 200 - 65KB - /.git/refs/
[10:31:52] 200 - 65KB - /.git/info/
[10:31:52] 200 - 65KB - /.git/logs/
[10:31:52] 200 - 32B - /.git/refs/remotes/origin/HEAD
[10:31:52] 302 - 0B - /.git/refs/tags -> /.git/refs/tags/
[10:31:52] 200 - 67KB - /.git/objects/
[10:31:52] 200 - 81B - /.gitmodules
[10:31:59] 200 - 1KB - /LICENSE
[10:32:00] 200 - 20KB - /README.md
```

发现一堆.git下的文件

访问下/.git试试，果然有很多文件，接下来使用GitHack工具

```
PS D:\GitHack\GitHack-master\GitHack-master> python GitHack.py http://week-2.hgame.lwsec.cn:32427/.git/
[+] Download and parse index file ...
[+] .gitmodules
[+] LICENSE
[+] README.md
[+] TODO.txt
[+] This_is_flag
[+] assets/Matrix-Code.ttf
[+] assets/Matrix-Resurrected.ttf
[+] assets/coptic_msdf.png
```

就找到flag了，这个This_is_flag文件是放在和GitHack同一个文件夹下的，然后打开这个This_is_flag的文件就可以看到flag了

V2Board

打开靶机之后就是一个登录界面，也不知道干啥，先注册一个账号，登进去看看

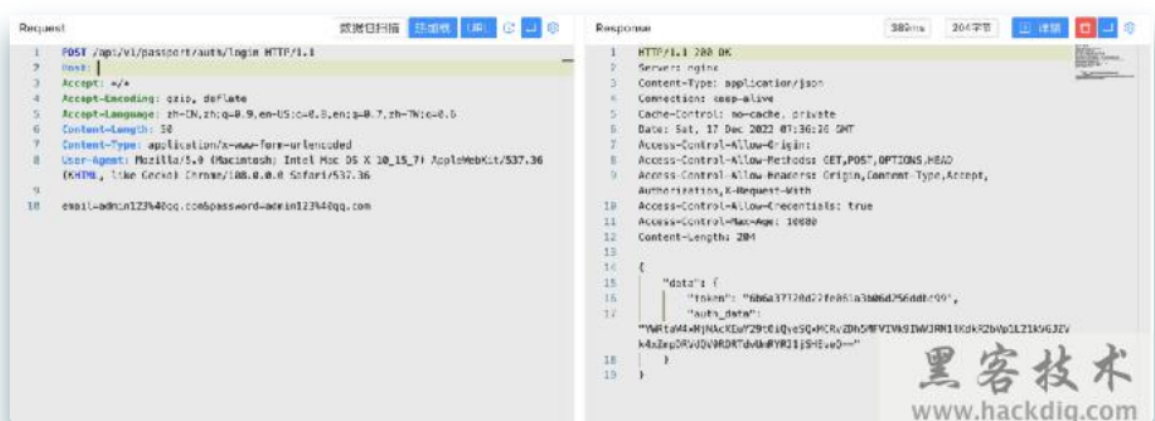


看着这个url的#就很激动，就想着前几天刚看的DOM型xss，但是不管咋用都不行，然后问了ek1ng学长，只是个hash型的前端路由，注意这道题的版本1.6.1，然后就去百度了一下1.6.1版本的V2Board，存在一个越权漏洞。

越权漏洞可分为水平越权和垂直越权，水平越权指攻击者尝试访问与他拥有相同权限的用户资源。垂直越权指权限较低的用户去执行高权限用户的操作。

然后我找了一些原版V2Board的漏洞

在登陆验证的代码中，成功使用 email 和 password 登陆后会返回 token 和 auth_data



同时 auth_data 会缓存于 Redis 中



由于 Admin.php 文件中只验证了 authorization 是否在 Redis 的缓存中，所以当注册任意一个用户进行登陆后获取到 auth_data 就可以任意调用 管理员的接口

我重开一个靶机，重新在登录界面进行bp抓包，登陆之后确实会返回一个token和auth_data。

先看看token是啥，Token是服务端生成的一串字符串，以作客户端进行请求的一个令牌,当第一次登录后,服务器生成一个Token便将此Token返回给客户端,以后客户端只需带上这个Token前来请求数据即可,无需再次带上用户名和密码。

但是当我想把auth_data里的值给到/api/v1/user/后的任意一个有token返回值的authorization标题头的时候发现这authorization的值和auth_data的值是一样的(恼),然后又问了eking学长，请求头里本身的authorization是你自己的，你需要通过越权漏洞拿到admin的authorization。

因为只有在登陆之后才会返回auth_data，那既然要拿到admin的authorization那也就是得在登录页面，然后试了下在用户登录的接口添加一个authorization，返回的和原来是一样的，没思路了...

Search Commodity

没有密码，那就用bp抓包然后用自己的找的字典慢慢爆破，爆破了几个字典都没爆破出来，直到R1esbyfe学长放出了个hint，再找个字典爆破一下，总算出来了

