Damocles

Abyssworld 游戏分析报告

2024.03.21

Senna

DAMOCLES LABS

Damocles

目录

- 概要
- 游戏背景
  - 游戏版本
  - 游戏类型&游戏引擎
  - 游戏玩法可能存在的问题
- 游戏安全分析
  - 游戏代码保护
  - 游戏基础反作弊
  - 游戏逻辑问题
  - 游戏协议&Server 安全性分析
- Web3 安全分析
  - 代币合约安全
  - 游戏内经济系统安全
- 关于 Damocles

Damocles

一、 概要

作为一款 ARPG 品类的游戏 Abyssworld 在其 Client,WebServer，Game Server 上的安全性为 0，存在 RCE，结算宝箱重放攻击以及敏感信息泄露风险，并且存在任意用户邮箱爆破登录，Client 任意作弊问题，由于其合约功能单一，链上风险较小。

安全性评分： ★ ☆ ☆ ☆ ☆

二、 游戏背景

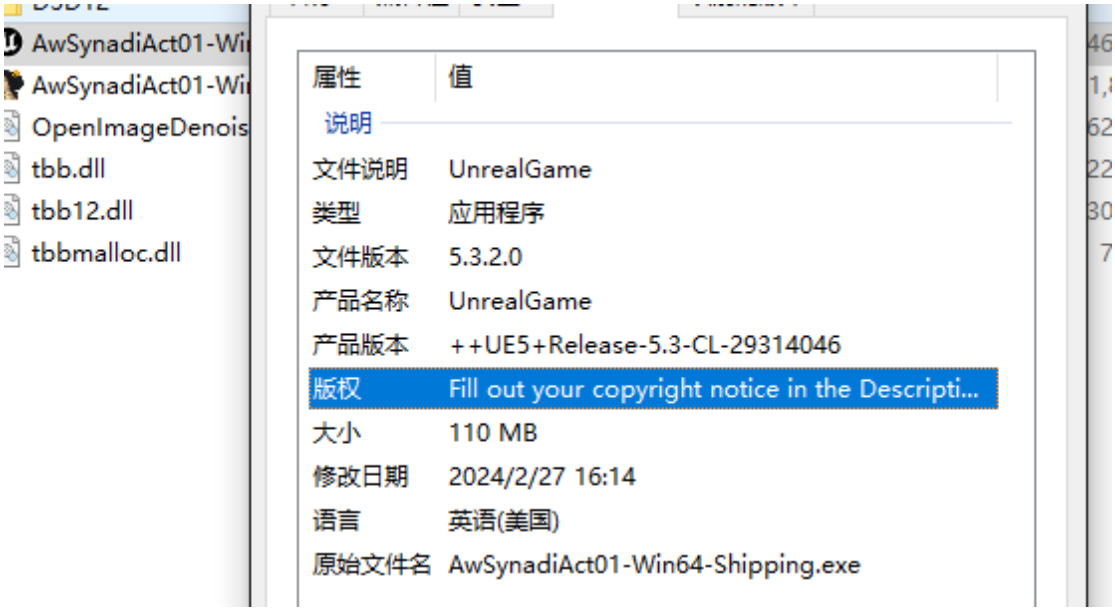
- 进行评估的游戏版本：ACT1
- 游戏类型&游戏引擎：ARPG，UE5
- 游戏玩法可能存在的问题：
  - 非法移动(修改本地人物属性进行加速)
  - 攻击加倍
  - 瞬移
  - 伤害减免
  - 结算重放攻击
  - 跳跃等一些本地人物属性的修改

三、 游戏安全性分析

游戏代码保护：

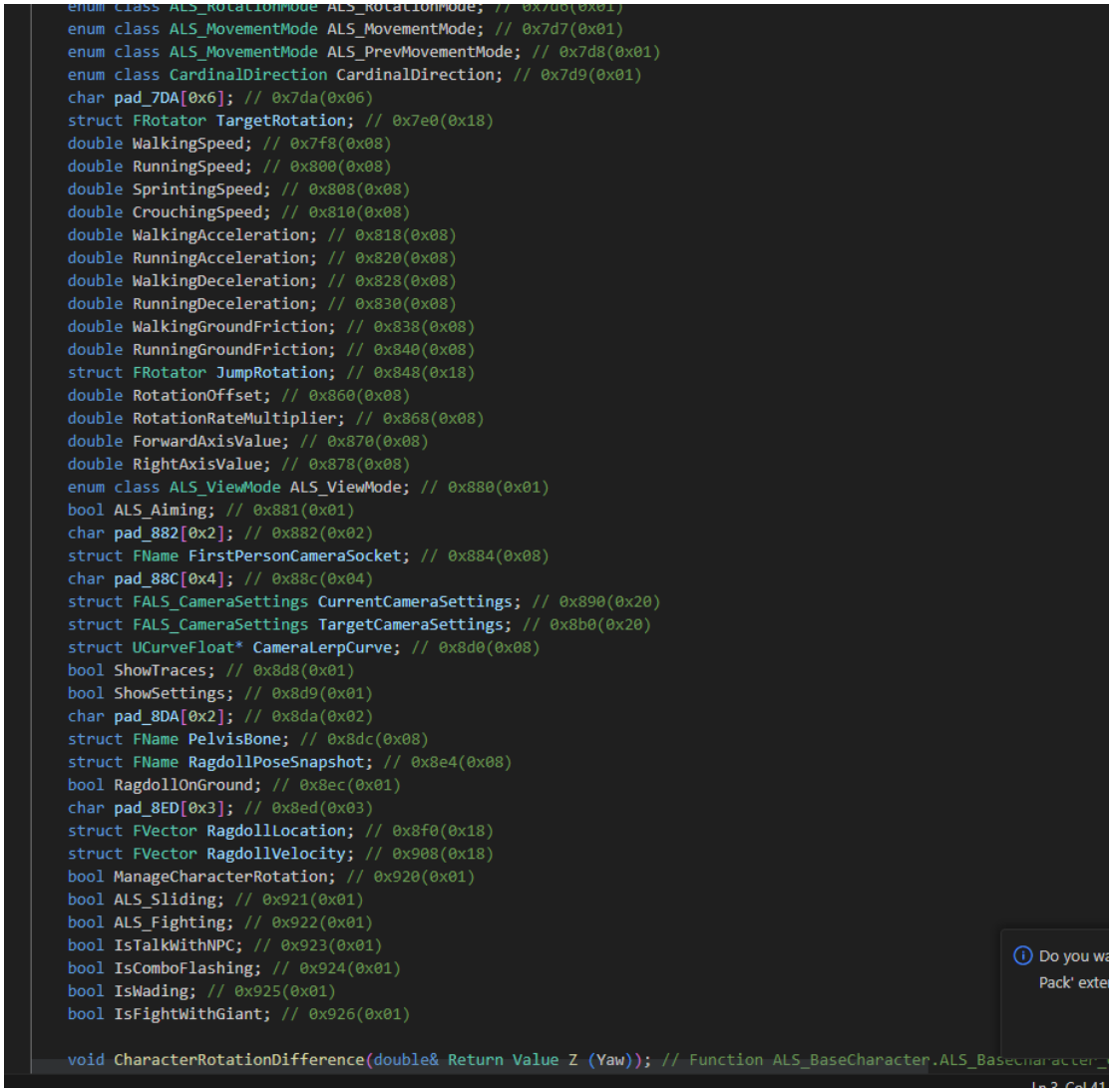
分析过程：

1. 由于不同的引擎有不同的分析模式，所以在获取到游戏 EXE 后首先需要确定游戏使用的引擎，通过对游戏基础信息识别我们可以确定该游戏是使用 UE5 进行开发。



2. 通过工具进行 dump UE 的人物结构进行快速定位，定位以后通过 UE 特有的链表结构进行索引与修改

Damocles



```
enum class ALS_RotationMode ALS_RotationMode; // 0x7d6(0x01)
enum class ALS_MovementMode ALS_MovementMode; // 0x7d7(0x01)
enum class ALS_MovementMode ALS_PrevMovementMode; // 0x7d8(0x01)
enum class CardinalDirection CardinalDirection; // 0x7d9(0x01)
char pad_7DA[0x6]; // 0x7da(0x06)
struct FRotator TargetRotation; // 0x7e0(0x18)
double WalkingSpeed; // 0x7f8(0x08)
double RunningSpeed; // 0x800(0x08)
double SprintingSpeed; // 0x808(0x08)
double CrouchingSpeed; // 0x810(0x08)
double WalkingAcceleration; // 0x818(0x08)
double RunningAcceleration; // 0x820(0x08)
double WalkingDeceleration; // 0x828(0x08)
double RunningDeceleration; // 0x830(0x08)
double WalkingGroundFriction; // 0x838(0x08)
double RunningGroundFriction; // 0x840(0x08)
struct FRotator JumpRotation; // 0x848(0x18)
double RotationOffset; // 0x860(0x08)
double RotationRateMultiplier; // 0x868(0x08)
double ForwardAxisValue; // 0x870(0x08)
double RightAxisValue; // 0x878(0x08)
enum class ALS_ViewMode ALS_ViewMode; // 0x880(0x01)
bool ALS_Aiming; // 0x881(0x01)
char pad_882[0x2]; // 0x882(0x02)
struct FName FirstPersonCameraSocket; // 0x884(0x08)
char pad_88C[0x4]; // 0x88c(0x04)
struct FALS_CameraSettings CurrentCameraSettings; // 0x890(0x20)
struct FALS_CameraSettings TargetCameraSettings; // 0x8b0(0x20)
struct UCurveFloat* CameraLerpCurve; // 0x8d0(0x08)
bool ShowTraces; // 0x8d8(0x01)
bool ShowSettings; // 0x8d9(0x01)
char pad_8DA[0x2]; // 0x8da(0x02)
struct FName PelvisBone; // 0x8dc(0x08)
struct FName RagdollPoseSnapshot; // 0x8e4(0x08)
bool RagdollOnGround; // 0x8ec(0x01)
char pad_8ED[0x3]; // 0x8ed(0x03)
struct FVector RagdollLocation; // 0x8f0(0x18)
struct FVector RagdollVelocity; // 0x908(0x18)
bool ManageCharacterRotation; // 0x920(0x01)
bool ALS_Sliding; // 0x921(0x01)
bool ALS_Fighting; // 0x922(0x01)
bool IsTalkWithNPC; // 0x923(0x01)
bool IsComboFlashing; // 0x924(0x01)
bool IsWading; // 0x925(0x01)
bool IsFightWithGiant; // 0x926(0x01)

void CharacterRotationDifference(double& Return Value Z (Yaw)); // Function ALS_BaseCharacter.ALS_Base
```

并且使用 IDA 进行静态代码分析时发现游戏代码结构完整且毫无保护

Damocles

```
v1 = *(_DWORD *)(a1 + 124);
while ( 1 )
{
  v3 = 2048;
  if ( v1 != *(_DWORD *)(a1 + 120) )
    v3 = 0;
  v4 = v1 + v3;
  if ( (unsigned int)v4 > *(_DWORD *)(a1 + 120) )
    sub_14199C790(a1 + 112, v4, 1024i64);
  v5 = *(unsigned int *)(a1 + 124);
  v6 = (char *)(v5 + (*(_QWORD *)(a1 + 112) & 0x7FFFFFFFFFFFFFFFi64));
  v7 = recv(*(_QWORD *)(a1 + 96), v6, *(_DWORD *)(a1 + 120) - v5, 0);
  v8 = v7;
  if ( v7 < 0 )
  {
    if ( WSAGetLastError() == 10035 )
    {
      *(_BYTE *)(a1 + 2) = 1;
      return 1i64;
    }
    v15 = "Error occurred on socket recv";
    goto LABEL_46;
  }
  *(_DWORD *)(a1 + 124) += v7;
  v9 = 3;
  v1 = *(_DWORD *)(a1 + 124);
  v10 = (char *)(*(_QWORD *)(a1 + 112) & 0x7FFFFFFFFFFFFFFFi64);
  v11 = v6 - 3;
  v12 = 4;
  if ( v6 - 3 < v10 + 24 )
    v9 = 0;
  if ( v6 - 3 < v10 + 24 )
    v11 = v6;
  v13 = v7 + v9;
  if ( v13 >= 4 )
  {
    while ( *(_DWORD *)&v11[v12 - 4] != 168626701 )
    {
      v14 = 1;
      if ( v11[v12 - 1] > 13 )
        v14 = 4;
      v12 += v14;
      if ( v12 > v13 )
        goto LABEL_18;
    }
    if ( v12 >= 0 )
      break;
  }
LABEL_18:
  if ( v1 > 0x2000 )
  {
    v15 = "Headers have grown larger than expected";
    goto LABEL_46;
  }
  if ( !v8 )
  {
    v15 = "ATH0.RecvMessage";
LABEL_46:
```

因此可以结合 dump 的 UE 数据结构和 IDA 对游戏代码逻辑进行基本的静态分析。

分析结论：

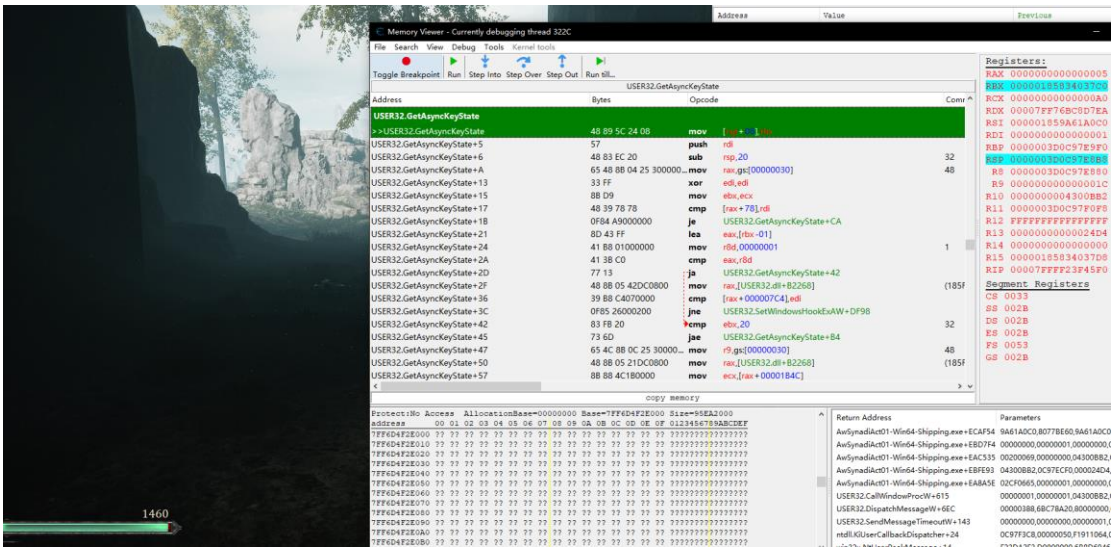
Abyss World 在游戏代码保护方面得分为 0 分，其 client 代码没有任何保护，也就是说没有任何的门槛手段对抗恶意玩家。

Damocles

游戏基础反作弊：

分析过程：

1. 在基础反作弊检测方面，我们主要从两个方面进行测试，一个是游戏是否存在反调试，另一个是游戏是否存在读写保护。
2. 在游戏打开状态下使用 CE 进行附加，并且对通用函数进行下断点，发现游戏并没有退出，或者提示



3. 可以直接修改游戏内的人物属性，并且 GS 不会进行踢出操作。

分析结论：

1. Abyss World 在反作弊对抗上基本保护为 0，缺少针对动态调试，动态分析的对抗，因为对于想作恶的玩家来说成本很低，并且缺少对已经作弊的玩家的检测能力。
2. 只测试反调试和读写保护两个方面的原因是对于一块外挂来说，找数据与实现功能只需要通过调试和读写就可以实现。如果最基础的两个保护能力都缺失的话，那么一些注入、hook 等检测也毫无意义。

游戏逻辑问题

分析过程：

由于该游戏属于单机游戏，所有的作弊手段均可以判定为本地的代码问题，由于其本身并没有任何的代码保护，所以可以通过修改坐标的位置进行人物的瞬移等等问题，单机游戏的可操作性过多，再次并不进行深入的展示。

分析结论：

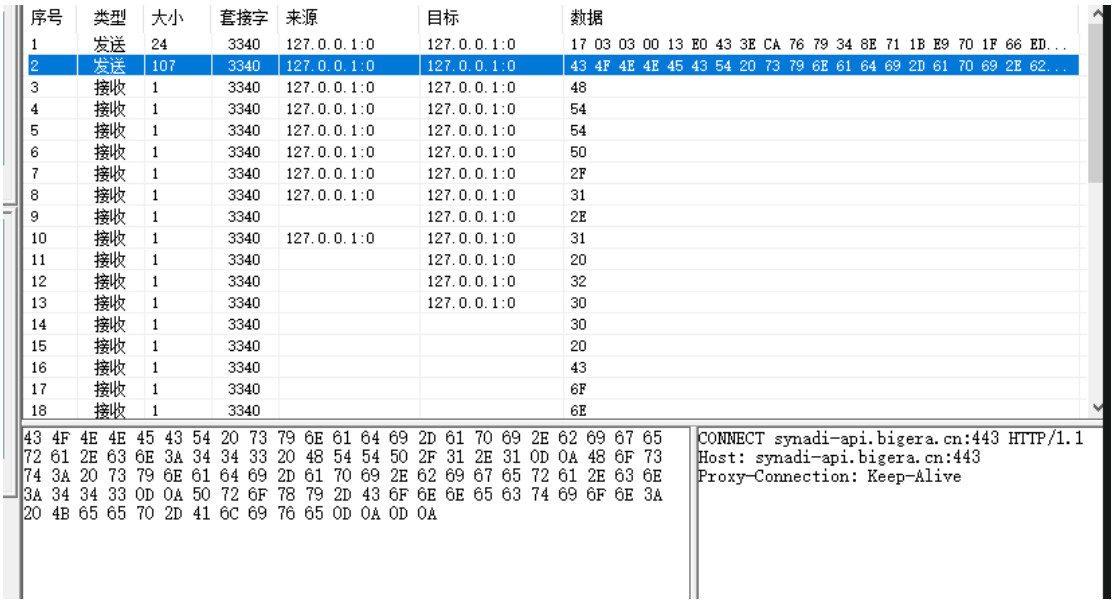
1. 对于一款游戏来说，其本地逻辑的安全性与 GS 判定和本地的安全手段息息相关，但是 AbyssWorld 这款游戏却是属于与 GS 只交互一次的单机游戏，所以很多数据无法由服务器进行判定，client 可以随意进行逻辑修改只需要想象力够丰富，因为该游戏的游戏逻辑安全评分为 0。

Damocles

游戏协议&Server 安全性分析

本地游戏协议分析

1. AbyssWorld 在整体游戏过程中仅与服务器交互一次，交互的时间就是在结算时，交互时会携带一些本地数据，并且在整体游戏过程中并没有数据采集的行为，所以在结算时的发包完全可以伪造，但是由于游戏的结算下发逻辑并不是实时触发需要在 3.31 日才会发放奖励，所以无法判断重放的有效性，但是通过整体逻辑来推测，其存在直接进行重放就可以获得奖励的风险隐患。



序号	类型	大小	套接字	来源	目标	数据
1	发送	24	3340	127.0.0.1:0	127.0.0.1:0	17 03 03 00 13 E0 43 3E CA 76 79 34 8E 71 1B E9 70 1F 66 ED...
2	发送	107	3340	127.0.0.1:0	127.0.0.1:0	43 4F 4E 4E 45 43 54 20 73 79 6E 61 64 69 2D 61 70 69 2E 62...
3	接收	1	3340	127.0.0.1:0	127.0.0.1:0	48
4	接收	1	3340	127.0.0.1:0	127.0.0.1:0	54
5	接收	1	3340	127.0.0.1:0	127.0.0.1:0	54
6	接收	1	3340	127.0.0.1:0	127.0.0.1:0	50
7	接收	1	3340	127.0.0.1:0	127.0.0.1:0	2F
8	接收	1	3340	127.0.0.1:0	127.0.0.1:0	31
9	接收	1	3340		127.0.0.1:0	2E
10	接收	1	3340	127.0.0.1:0	127.0.0.1:0	31
11	接收	1	3340		127.0.0.1:0	20
12	接收	1	3340		127.0.0.1:0	32
13	接收	1	3340		127.0.0.1:0	30
14	接收	1	3340			30
15	接收	1	3340			20
16	接收	1	3340			43
17	接收	1	3340			6F
18	接收	1	3340			6E

```
43 4F 4E 4E 45 43 54 20 73 79 6E 61 64 69 2D 61 70 69 2E 62 69 67 65
72 61 2E 63 6E 3A 34 34 33 20 48 54 54 50 2F 31 2E 31 0D 0A 48 6F 73
74 3A 20 73 79 6E 61 64 69 2D 61 70 69 2E 62 69 67 65 72 61 2E 63 6E
3A 34 34 33 0D 0A 50 72 6F 78 79 2D 43 6F 6E 6E 65 63 74 69 6F 6E 3A
20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 0D 0A
```

```
CONNECT synadi-api.bigera.cn:443 HTTP/1.1
Host: synadi-api.bigera.cn:443
Proxy-Connection: Keep-Alive
```

游戏 Server 端安全性分析

针对 abyssworld.games 相关的一些列服务器资产我们进行了浅层的渗透测试，在测试当中发现了一些列问题，其中包括闲置的 ***Admin 服务外放，Mysql 端口公开，S3 服务可遍历，任意用户登录注册码爆破****等一系列问题，其中任意用户的注册码爆破登录问题是最严重的，因为其可以会对用户的资产造成损失。*

Damocles

1. 对外公开的敏感资产汇总：

闲置 Django 后台公开	https://abyssworld.games/admin/login/?next=/admin/
S3 存储桶遍历	http://bucket.portal.abyssworld.games/
Mysql 端口公开	目前已开启 Filtered

2. **任意用户枚举和爆破登录问题：**

举例任意用户登录：

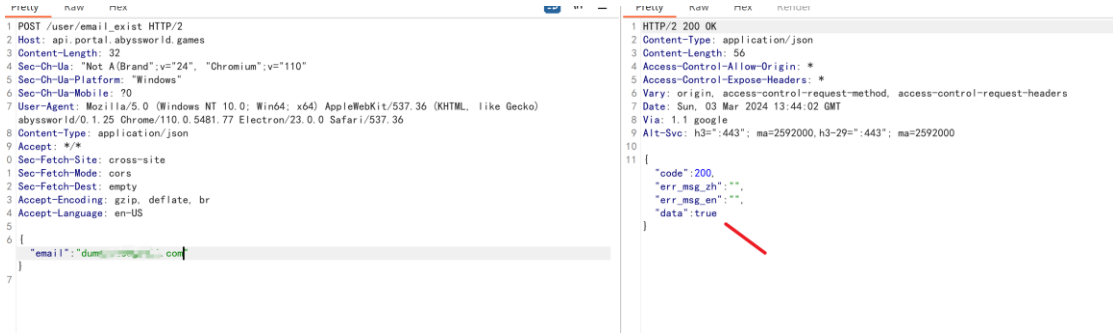
登录抓包，发现登录会验证邮箱是否存在，并且登录验证码没有限制，通过爆破可以获取用户的登录认证信息，实现任意账号登录。

存在的安全问题：

1. 用户邮箱枚举
2. 登录验证码没有限制，可以爆破获取用户的登录凭证

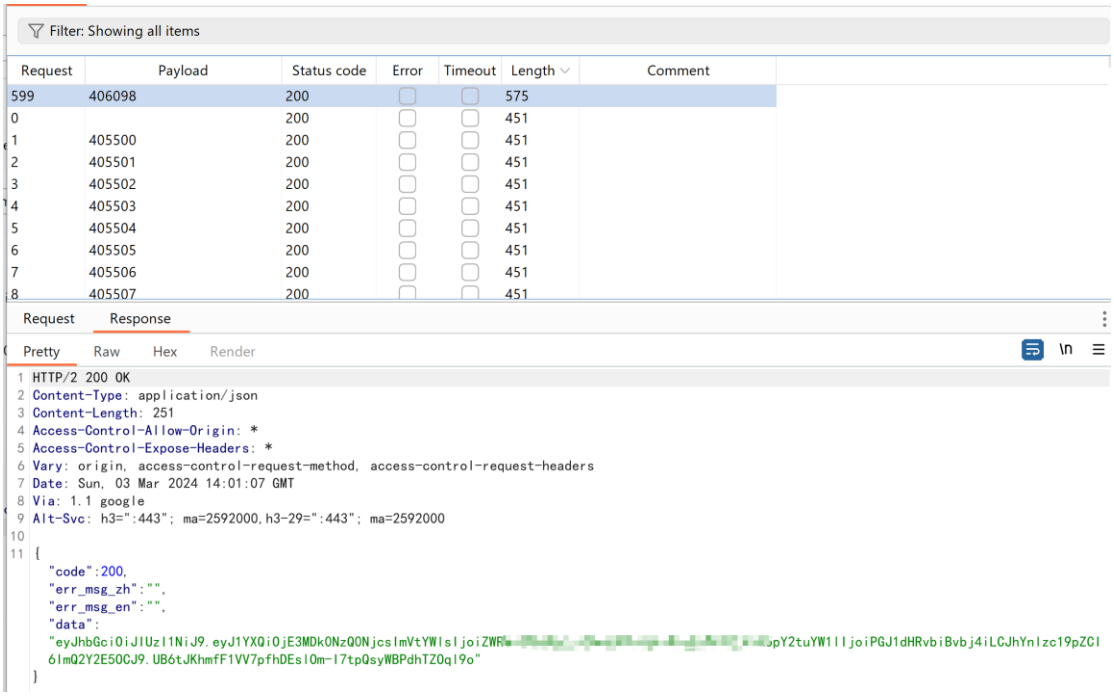
漏洞复现：

a) 通过 https://api.portal.abyssworld.games/user/email_exist 接口判断用户是否存在。



b) 用户若存在，发送登录验证码，通过接口：https://www.abyssworld.games/user/email_login 爆破获取到用户的 jwt token。

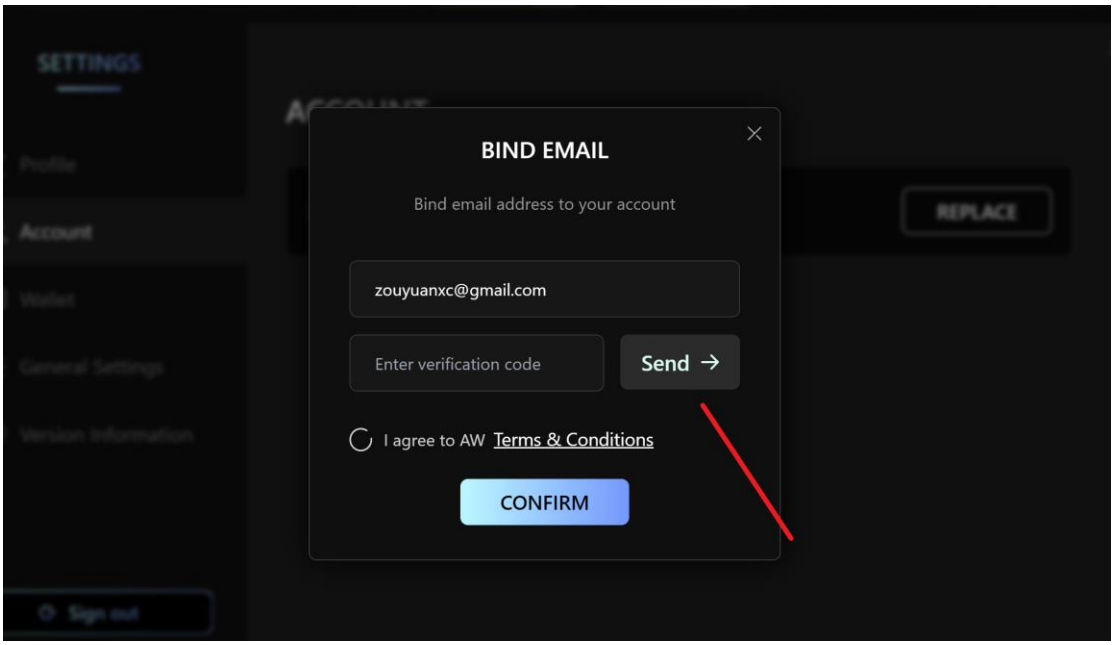
Damocles



c) 执行登录

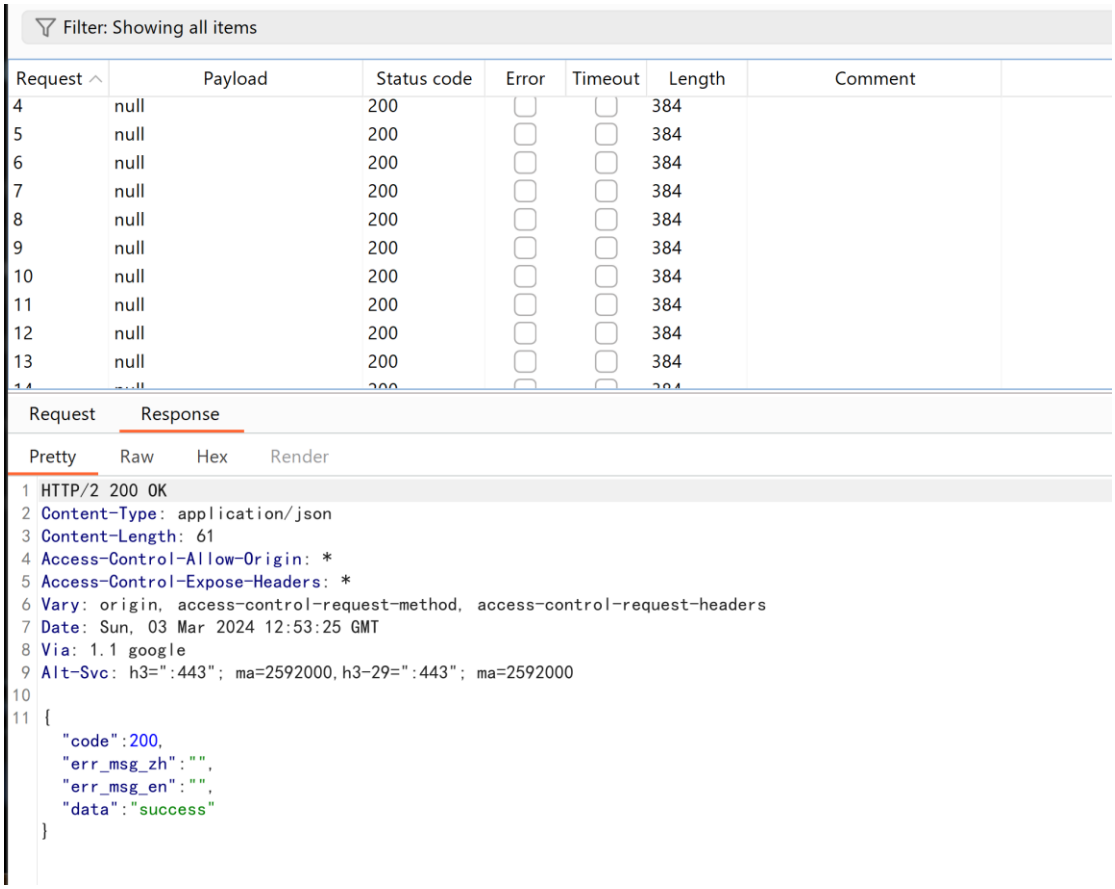
3. 垃圾邮件轰炸

可以占用游戏服务器的邮件发送接口对其他用户进行垃圾邮件轰炸，危害程度比较小但是可以占用服务器资源。

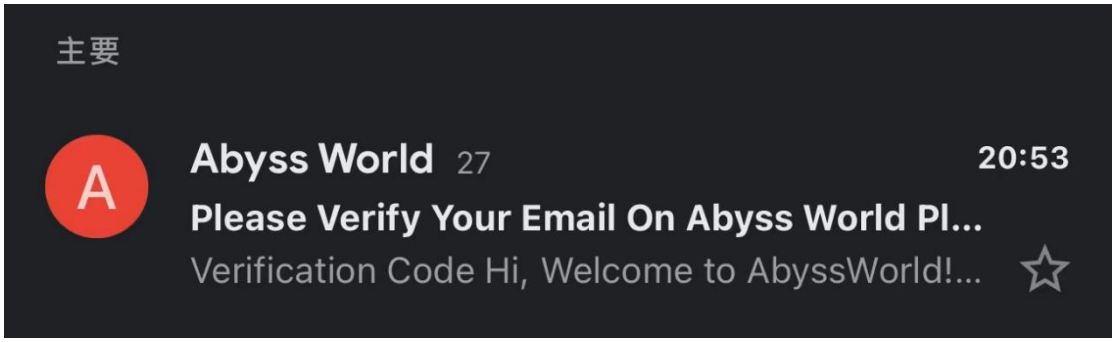


Send 处进行抓包

Damocles



收到 27 封邮件：



WEB3 安全分析：

AbyssWorld 可供参考的资料中能提供的只有其在 polygon 上发行的 AWC 和 AWT Token,

```
1411 contract AWC is ERC20, ERC20Snapshot, Ownable {
1412     constructor() ERC20("AWC", "AWC") {
1413         _mint(msg.sender, 20000000 * 10 ** decimals());
1414     }
1415
1416     function snapshot() public onlyOwner {
1417         _snapshot();
1418     }
1419
1420     function mint(address to, uint256 amount) public onlyOwner {
1421         _mint(to, amount);
1422     }
1423
1424     // The following functions are overrides required by Solidity.
1425
1426     function _beforeTokenTransfer(address from, address to, uint256 amount)
1427         internal
1428         override(ERC20, ERC20Snapshot)
1429     {
1430         super._beforeTokenTransfer(from, to, amount);
1431     }
1432 }
```

File 1 of 12 : AWT.sol

```
2  pragma solidity ^0.8.9;
3
4  import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
5  import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
6  import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Snapshot.sol";
7  import "@openzeppelin/contracts/access/Ownable.sol";
8
9  contract AWT is ERC20, ERC20Burnable, ERC20Snapshot, Ownable {
10     constructor() ERC20("AWT", "AWT") {
11         _mint(msg.sender, 10000000000 * 10 ** decimals());
12     }
13
14     function snapshot() public onlyOwner {
15         _snapshot();
16     }
17
18     // The following functions are overrides required by Solidity.
19
20     function _beforeTokenTransfer(address from, address to, uint256 amount)
21         internal
22         override(ERC20, ERC20Snapshot)
23     {
24         super._beforeTokenTransfer(from, to, amount);
25     }
26 }
```

其中 AWC Token 有增发风险。

Damocles

关于 Damocles

Damocles labs 是成立于 2023 年的安全团队，专注于 Web3 行业的安全，业务内容包括：合约代码审计，业务代码审计，渗透测试，GameFi 代码审计，GameFi 漏洞挖掘，GameFi 外挂分析，GameFi 反作弊。

我们会在 Web3 安全行业持续发力，并且尽可能多的输出分析报告，提升项目方和用户对 GameFi 安全的感知度，以及促进行业的安全发展。