Seraph Security Analysis

2023.11.24

Senna

DAMOCLES LABS

Contents

一、 Summary

Seraph opened its third beta test on November 22, 2023. On November 24, the Damocles team conducted a security analysis and assessment of the game. However, the assessment results were not satisfactory. Firstly, the project contained a significant amount of logging information in the code, which allowed the inference that the project team was not from Korea but from China. Additionally, the game adopted the Unity engine to load Lua scripts without any protection measures or the use of techniques like Lua JIT to enhance reverse engineering difficulty and protect the source code. This resulted in the complete exposure of the source code, as it could be dumped from memory by simply hooking the load function. However, since the game belongs to the ARPG genre, it has a natural advantage against cheating. Most of the game data is synchronized through the server, which mitigates the security issues to some extent.

Security Rating: ★★★☆☆

二、 Game Background

- Game Version: v0.0.0.6
- Genres & Engine: ARPG, Unity
- Possible Issues in Gameplay:
  - Teleportation
  - Accelerated (Accelerated Movement, Accelerated Skill Usage)

- Auto-Grinding
- Multiplier Modification
- Invincibility
- Buff Modification（Buff Modification to Increase Soul Crystal Output or Other Benefits）

三、 Game Security Analysis

Game Code Protection：

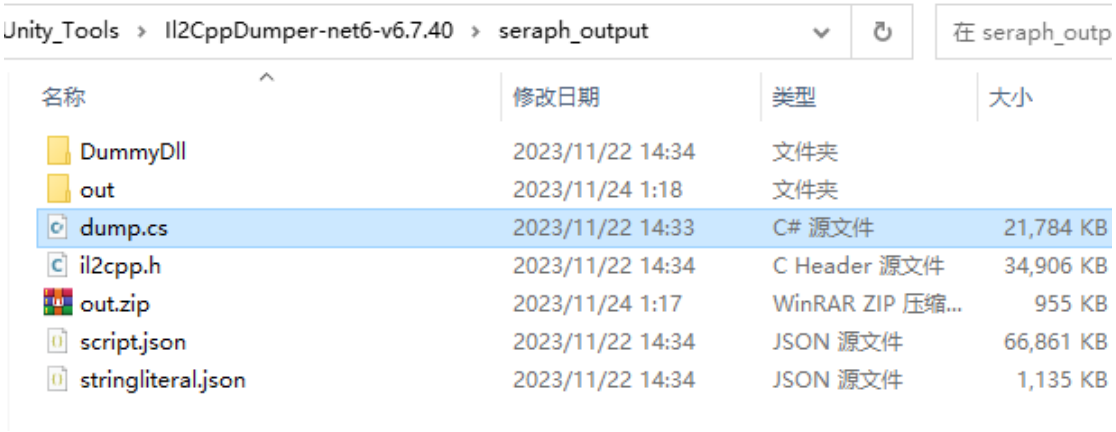
Analysis Process：

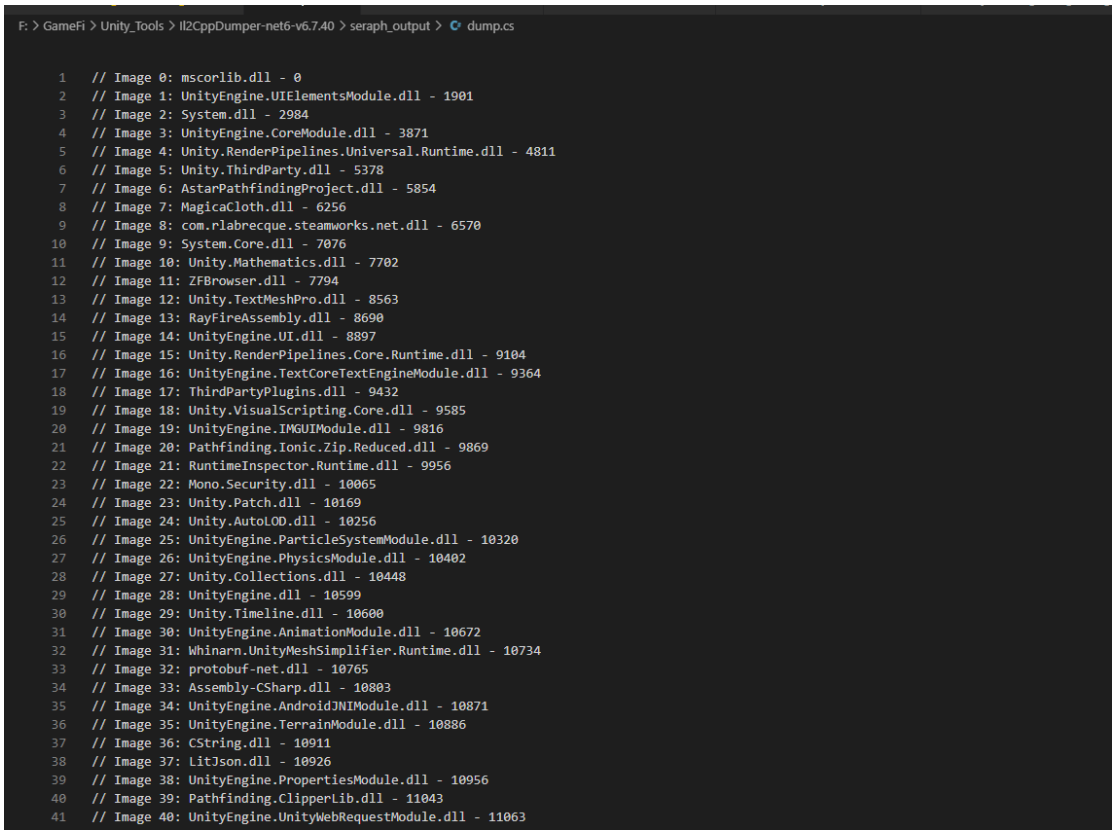
1. Since different engines have different analysis modes, it is important to determine the game engine used after obtaining the game EXE. By analyzing the basic information of the game, we can determine that this game was developed using Unity。

out	2023/11/29 16:22	文件夹	
out - 副本	2023/11/29 12:30	文件夹	
SeraphGamma_Data	2023/11/21 23:52	文件夹	
baselib.dll	2023/11/21 22:51	应用程序扩展	410 KB
GameAssembly.dll	2023/11/21 22:52	应用程序扩展	45,209 KB
out.zip	2023/11/25 18:26	WinRAR ZIP 压缩...	955 KB
SeraphGamma.exe	2023/11/21 22:51	应用程序	651 KB
UnityCrashHandler64.exe	2023/11/21 22:51	应用程序	1,089 KB
UnityPlayer.dll	2023/11/21 22:51	应用程序扩展	29,905 KB

2. By examining the GameAssembly.dll and global-metadata.dat files in the game directory, it can be determined that the game was

compiled using the il2cpp compilation mode. Therefore, the source code can be recovered using iL2Cppdumper.





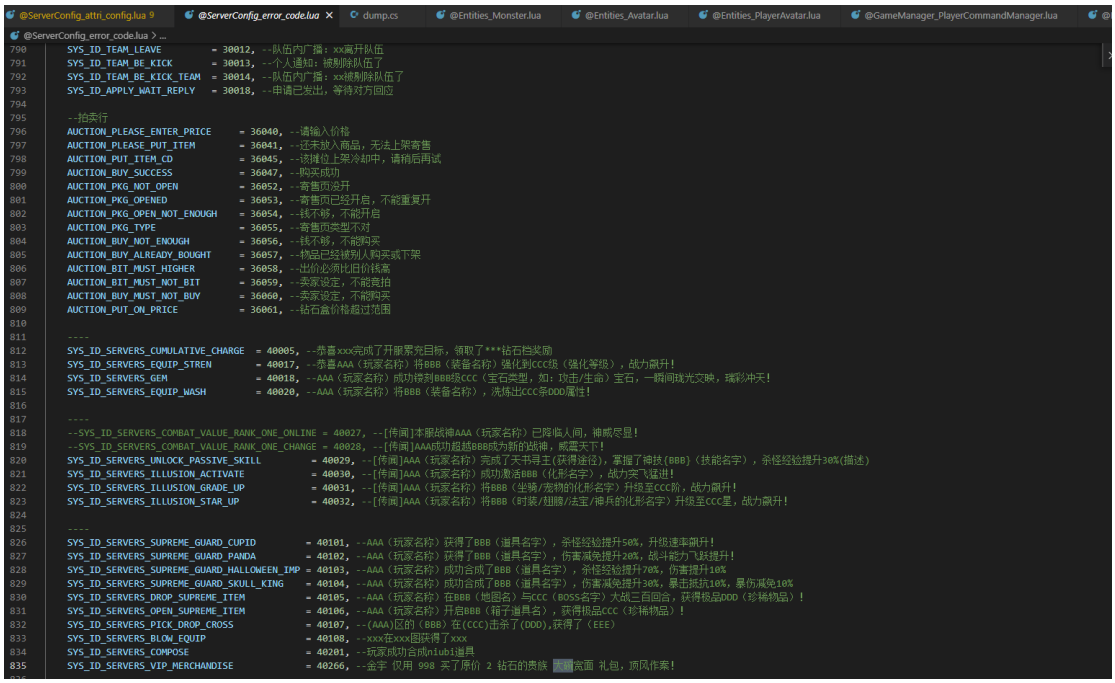
However, in the dump.cs file, no strongly relevant code logic related to the game was found. Therefore, it is speculated that the game may not have been developed using C#, but rather loaded through Lua. As a result, by hooking into the functions related to loading buffs in the game's code, the actual source code of the game was obtained.

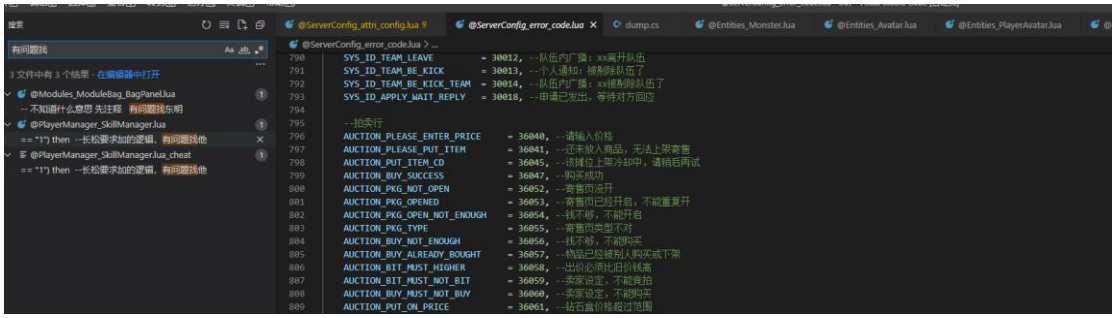
@AI_ai_utils.lua	2023/11/29 11:56	Lua 源文件	3 K
@AI_bt_action_nodes.lua	2023/11/29 11:56	Lua 源文件	39 K
@AI_bt_base_nodes.lua	2023/11/29 11:56	Lua 源文件	13 K
@AI_bt_condition_nodes.lua	2023/11/29 11:56	Lua 源文件	12 K
@AI_mgr_entity_aggro_cell.lua	2023/11/29 11:56	Lua 源文件	8 K
@AI_mgr_entity_ai_cell.lua	2023/11/29 11:56	Lua 源文件	202 K
@BT_bt94.lua	2023/11/29 12:06	Lua 源文件	3 K
@BT_bt20003.lua	2023/11/29 12:03	Lua 源文件	2 K
@BT_bt20004.lua	2023/11/29 12:06	Lua 源文件	1 K
@BT_bt21000.lua	2023/11/29 12:16	Lua 源文件	1 K
@BT_bt21015.lua	2023/11/29 12:15	Lua 源文件	1 K
@BT_bt23007.lua	2023/11/29 12:11	Lua 源文件	1 K
@BT_bt23020.lua	2023/11/29 12:08	Lua 源文件	2 K
@Core_Class.lua	2023/11/29 11:56	Lua 源文件	4 K
@Core_Entity.lua	2023/11/29 11:56	Lua 源文件	7 K
@Core_EventDispatcher.lua	2023/11/29 11:56	Lua 源文件	2 K
@Core_FrameTimerHeap.lua	2023/11/29 11:56	Lua 源文件	1 K
@Core_FunctionInvoker.lua	2023/11/29 11:56	Lua 源文件	1 K
@Core_GameWorld.lua	2023/11/29 11:56	Lua 源文件	52 K
@Core_LoggerHelper.lua	2023/11/29 11:56	Lua 源文件	3 K
@Core_LuaBehaviour.lua	2023/11/29 11:56	Lua 源文件	2 K
@Core_PrintTable.lua	2023/11/29 11:56	Lua 源文件	4 K
@Core_Queue.lua	2023/11/29 11:56	Lua 源文件	2 K
@Core_Stack.lua	2023/11/29 11:56	Lua 源文件	2 K
@Core_StringEx.lua	2023/11/29 11:56	Lua 源文件	4 K
@Core_TableEx.lua	2023/11/29 11:56	Lua 源文件	6 K
@Core_TimerHeap.lua	2023/11/29 11:56	Lua 源文件	4 K
@Core_XmlSimple.lua	2023/11/29 11:56	Lua 源文件	6 K
@CSFacade.lua	2023/11/29 11:56	Lua 源文件	25 K
@Dramas_drama_trigger_cg.lua	2023/11/29 11:56	Lua 源文件	1 K
@Dramas_drama_trigger_guide.lua	2023/11/29 11:56	Lua 源文件	3 K
@Entities_Avatar.lua	2023/11/29 16:19	Lua 源文件	16 K

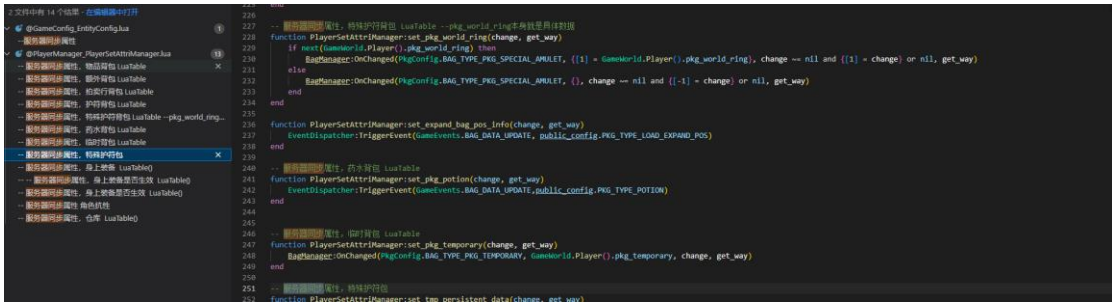
```lua
require "Core.GameWorld"
require "GameDataHelper.GameDataHelper"
require "GameManager.GameManager"
require "GameConfig.GameConfig"
require "GameUtil.GameUtil"
require "PlayerManager.PlayerManager"
require "CSFacade"
require "AI/bt_base_nodes"
require "AI/bt_action_nodes"
require "AI/bt_condition_nodes"

local Application = UnityEngine.Application
local OperationLogManager = GameManager.OperationLogManager
local SystemConfig = GameLoader.SystemConfig
local DataParseHelper = GameDataHelper.DataParseHelper

--主入口函数。从这里开始lua逻辑
function Main()
    GameManager.OperationLogManager.BaseLogData(GameWorld.public_config.OPERATION_LOG_14)
    --package.cpath = package.cpath .. ';C:/Users/admin/AppData/Roaming/JetBrains/IdeaIC2020.1/plugins/intellij-emmylua/classes/debugger/emmy/windows/x64/?.dll'
    --local dbg = require('emmy_core')
    --dbg.tcpListen('localhost', 9966)
    Init()
    Start()
end

function Init()
    local platformType = GameWorld.GetPlatformSetting()
    if platformType == "pc" then
        GameWorld.platformType = 1
    elseif platformType == "mobile" then
        GameWorld.platformType = 2

        GameManager.BagItemManager:Init()
    else
        GameWorld.platformType = 3
    end
    GameConfig.Global.SetLang()
    GameWorld.LoggerHelper.Init()
    GameManager.XMLManager.CheckReloadXML()
    GameManager.GUIManager.Init()
    GameManager.ModuleManager.Init()
    GameManager.LanguageManager:Init()
    GameManager.CreateRoleManager:Init()
    GameManager.DramaManager:Init()
    GameManager.TimelineManager:Init()
    GameManager.LoadingBarManager:Init()
    RegisterEntities()
--GameManager.ModuleManager.TestF = function() end
--LoggerHelper.Log("Init1:::",GameManager.XMLManager.map[1])
--LoggerHelper.Log("Init2:::",GameManager.XMLManager.spell[21].group)
end
```

And interesting comments were found in the game source code:

Damocles







Analysis Conclusion：

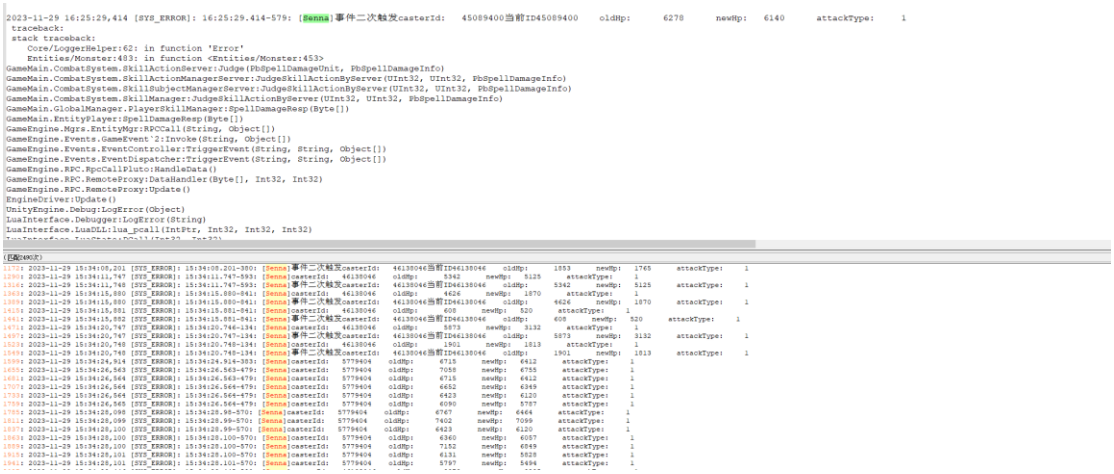
Seraph receives a score of 0 in terms of game code protection, indicating a complete lack of protection. In traditional Lua-based games, custom Lua interpreters are often used with LuaJIT to provide a certain level of code protection.

However, since Seraph lacks a robust code protection mechanism, it results in a low barrier and cost for malicious players to analyze the code. This can lead to unfairness for regular players if cheats or hacks are introduced, and there is a possibility of impacting the game's economic model.

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. After injecting the DLL using CE injection tool, we examine the game's log files to see if third-party logs are being printed.



3. By modifying the Lua logic, it was possible to modify in-game data such as critical hit rate, and the modifications were effective without any checks in the game. (Modifying attribute data only affects the visual display, as these fields are typically stored on the server-side, and local modifications usually do not have any actual effect.)

Analysis Conclusion:

1. Seraph scores 0 in terms of anti-cheat capabilities, which means that malicious users can cheat without any restrictions.

The main reason for specifically testing the reloading of Lua in the game is that this action serves as the foundation for cheating in Lua-based games. If this aspect cannot be effectively addressed, it indicates a weak point in the anti-cheat measures, and it is likely that other aspects of anti-cheat will also be lacking or ineffective.

Game Logic Issues

Analysis Process:

Since we have obtained the game's source code, we conducted a security analysis focusing on the logic layer. We did not analyze the protocol layer. In terms of the logic layer, we primarily performed security testing on the following aspects:

Attribute tampering during character initialization: (It was found that there are not many sensitive attributes in this section, and it does not yield significant benefits.)

```
function PlayerAvatar:OnEnterWorld()
    self.canPickUpStatus = 0
    self:AddListeners()
    GameManager.OperationLogManager.BaseLogData(GameWorld.public_config.OPERATION_LOG_10)
    self.cs.isLoadImmediately = true
    PlayerAvatar._base.OnEnterWorld(self)
    GameWorld.SetAccount(self.account_name)
    GameWorld.SetAvatarName(self.name)
    GameWorld.CheckFullUploadLog()
    --self._stateManager:SetDispatchEnterDead(true)
    -- self:CheckNewPlayer()
    -- self:SetPosToSlot()
    RedPointManager:OnEnterWorld()
    PlayerDataManager:OnPlayerEnterWorld()
    --EquipManager:OnPlayerEnterWorld();
    BagManager:OnPlayerEnterWorld()
    PlayerActionManager:OnPlayerEnterWorld()
    -- PlayerCommandManager:Init()
    PlayerCommandManager:OnPlayerEnterWorld()
    ResourceMappingManager:OnPlayerEnterWorld()
    CopperAvatarManager:OnPlayerEnterWorld()
    TrainManager:OnPlayerEnterWorld()
    self:LoadPlayerBleedingUIFx()
    self:SetPreloadSkill(true)
    self:SetLearnedSkillDict(self:GetEquipSkillIdList())
    self:PlayerEnterWorldLog()
    GameWorld.SetMaxScreenMonsterCount(GlobalParamsHelper.GetParamValue(916))
    if GameStateManager.GetState() ~= GAME_STATE.SCENE then
        --遮盖创角选角短暂穿帮问题
        --LoadingBarManager:ShowProgressBar(0.1)
    end
    --self:CheckLoadEmptyWeapon()
    self:SetFunctionOpenActive()
    self:SetServerInfo(LocalSetting.settingData.SelectedServerID, ServerListManager:SelectedServerName())

    --EventDispatcher:TriggerEvent(GameEvents._GameON_FUNCTIONPREVIEWPANEL) --进入游戏后刷新functionPreview
    if GUIManager.hasHandleEscape then
        self._clearStackTimer = TimerHeap:AddSecTimer(0, self._clearStackTime, 0, function()
            GUIManager.ClearClosePanelStack()
        end)
    end
    --GameWorld.ShowMainCamera()
    self:SetQualitySettingValue(0)

    if(GameLoader.SystemConfig.IsUsePlatformSdk and GameManager.LoginManager.accessToken) then
```

The next aspect we tested was the manipulation of skills related to active attacks. It was found that this part is only for display purposes and does not actually participate in damage validation.

```lua
function SkillManager:GetSkillShowDamage(data,level)
    --技能主要伤害
    local skillId = data.skillId
    local owner = data.owner--owner: 1=玩家,2=佣兵,3=其他玩家
    local cal = SkillDataHelper:GetDamageCal(skillId)
    if next(cal) then
        local param = self:DescParam(data,level)
        local minDamage
        local maxDamage
        local factor
        if owner == 1 then
            minDamage = AttributeManager:GetAttributeByAttID(attri_config.ATTRI_ID_DMG_MIN)
            maxDamage = AttributeManager:GetAttributeByAttID(attri_config.ATTRI_ID_DMG_MAX)
            factor = AttributeManager:GetAttributeByAttID(attri_config.ATTRI_ID_NON_WEAPON_ED)
        elseif owner == 2 then
            if data.mercenaryAtt then
                minDamage = data.mercenaryAtt[attri_config.ATTRI_ID_DMG_MIN] or 1
                maxDamage = data.mercenaryAtt[attri_config.ATTRI_ID_DMG_MAX] or 1
                factor = data.mercenaryAtt[attri_config.ATTRI_ID_NON_WEAPON_ED] or 0
            else
                minDamage = 1
                maxDamage = 1
                factor = 0
            end
        elseif owner == 3 then
            local attData = OtherPlayerManager:GetAllAttributesData()
            minDamage = attData[attri_config.ATTRI_ID_DMG_MIN] or 1
            maxDamage = attData[attri_config.ATTRI_ID_DMG_MAX] or 1
            factor = attData[attri_config.ATTRI_ID_NON_WEAPON_ED] or 0
        end
        local min = 0
        local max = 0
        for i,v in ipairs(cal) do
            if v == 0 then
            elseif v == 1 then
                min = min + param[i][1]/100 * minDamage * (1 + factor/100)
                max = max + param[i][1]/100 * maxDamage * (1 + factor/100)
            elseif v == 2 then
                if #param[i] ~= 2 then
                    LoggerHelper.Error("Config Error! ! ! ! ! ! Excel:spell.xml ---- damage_cal ----id:"..skillId)
                else
                    min = min + param[i][1]
                    max = max + param[i][2]
                end
            end
        end
        min = min - min % 0.1
        max = max - max % 0.1
        max = max * 100
        min = min * 100
        return {min,max}
    else
```

Lastly, we tested the modification of logic when monsters are attacked. It was found that modifying this aspect did not have any practical significance. It is speculated that the primary purpose of this module during development was to trigger events for recording purposes, rather than participating in actual calculations.

```lua
--这个方法，目前都是技能伤害导致血量变化。
function Monster:OnChangeHp(casterId, oldHp, newHp, attackType)

    -- 修改，只要血量变化 就会杀怪
    -- LoggerHelper.Error("[Senna]casterId:   " .. tostring(casterId) .. "    oldHp:      " .. tostring(oldHp) .. "      newHp:   " .. tost
    if casterId == GameWorld.Player().id and newHp <= 0 then
        EventDispatcher:TriggerEvent(GameEvents.KillMonster_Exp)
    end

    if casterId ~= GameWorld.Player().id then
        return
    end

    if newHp <= 0 then
        return
    end
    -- LoggerHelper.Error("casterId:   " .. tostring(casterId) .. "    oldHp:      " .. tostring(oldHp) .. "      newHp:   " .. tostring(ne
    EventDispatcher:TriggerEvent(GameEvents.MONSTER_HP_CHANGE_EVENT, self.id)


    if self.behavior_state == 0 then
        local exit_action = MonsterDataHelper:GetExitAction(self.monster_id)
        if exit_action > 0 then
            local player = GameWorld.Player()
            player.server.set_monster_behavior_state(self.id, 1)
        end
    end

    if self:GetEntityAI() then
        self:GetEntityAI():on_change_hp(casterId, tonumber(newHp) - tonumber(oldHp), attackType)
    end
    --LoggerHelper.Error("[Senna]事件二次触发casterId:   " .. tostring(casterId)..."当前ID"..GameWorld.Player().id .. "    oldHp:      " .. to
    casterId = GameWorld.Player().id
    newHp = 0
    attackType = 30
    if casterId == GameWorld.Player().id and newHp <= 0 then
        EventDispatcher:TriggerEvent(GameEvents.KillMonster_Exp)
        return
    end
end
```

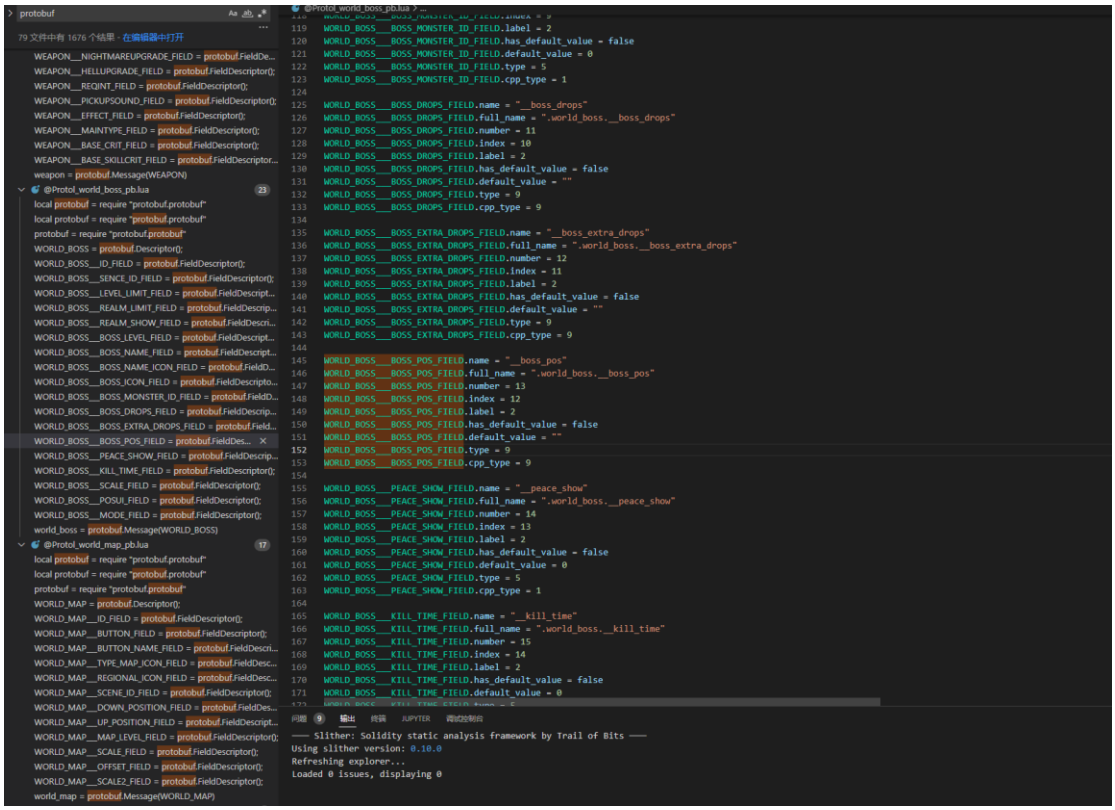
Analysis Conclusion：

1. Seraph did not show any effect on the three points we randomly tampered with, indicating that the damage calculation and display are separated, or the calculations are performed by the server, which provides a certain level of security. It receives a score of 3 out of 10.
2. However, some damage calculations are stored locally, leaving room for cheating.

Damocles

Game RPC Analysis

The game utilizes Protocol Buffers (protobuf) for protocol interaction, including interactions related to Web3. Currently, detailed testing has not been conducted specifically for this part, but there may be plans for thorough testing of the protobuf portion in the future.

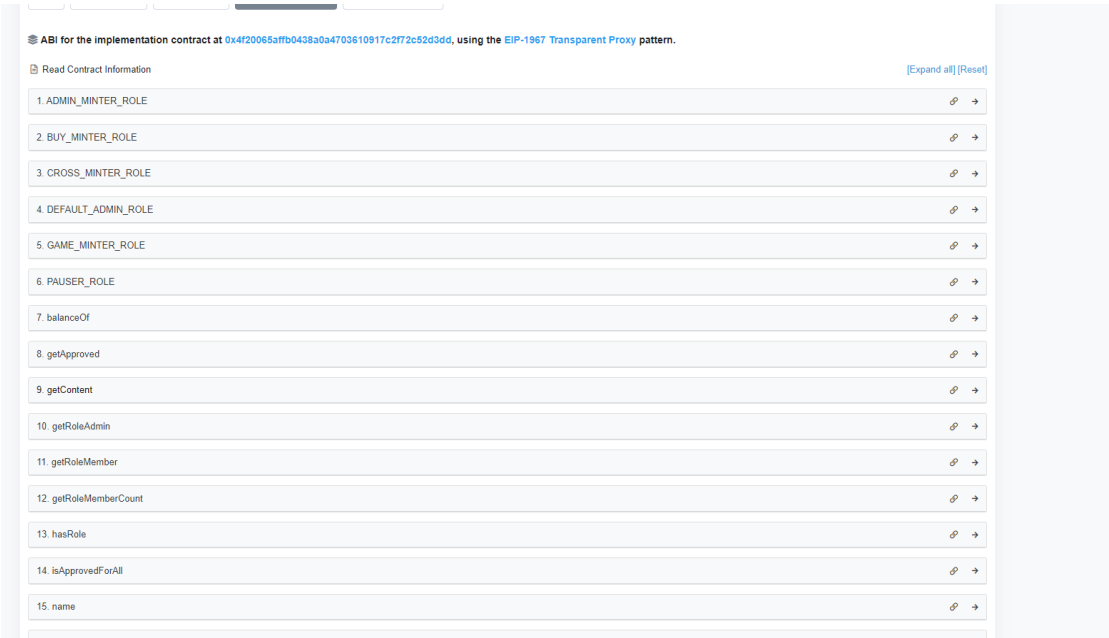


Damocles

WEB3 Security Analysis:

Summary:

Currently, Seraph has not issued its own token. The Mint contract is a standard NFT721 contract with a total supply of 3225, and both the Mint and cross-chain functionalities are controlled by roles. The on-chain security is controllable.



Game Economy System Security Analysis:

Currently, the primary method for earning currency in Seraph is through Soul Crystals. Whether it is crafting Soul Caskets or opening them, the server is responsible for the determination, while the client initiates the requests. The security is primarily controlled by the server, so the security assessment falls outside the scope of client security evaluation. In the future, Damocles may perform a comprehensive review and black-box testing of all requests.

```
        end
    elseif act_id == action_config.CHAOS_GET_MONEY_REQ then --获取账号pkg_chaos的货币数量
        self._XGoldCount = args
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_XGoldCount,args)
        EventDispatcher:TriggerEvent(GameEvents.RefreshXCoinsNum)
    elseif act_id == action_config.CHAOS_ADD_MONEY_REQ then --游戏内往pkg_chaos存货币
        ChaosWarehouseManager:RequestXGoldCount()
    elseif act_id == action_config.CHAOS_REMOVE_MONEY_REQ then --游戏内从pkg_chaos取货币
        ChaosWarehouseManager:RequestXGoldCount()
    elseif act_id == action_config.CHAOS_ITEM_UPDATE_RESP then --web端更新了pkg_chaos
        ChaosWarehouseManager:RequestPKGChaos(1,self.OnePageCount)
    elseif act_id == action_config.CHAOS_MONEY_UPDATE_RESP then --web端更新了货币
        ChaosWarehouseManager:RequestXGoldCount()
    elseif act_id == action_config.CHAOS_GET_PACK_Y_CNT then --获取铸造一个灵魂之匣需要的y币数量
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_YCoinToSoulBoxExchangeProportion,GameWorld.Player().tmp_persistent_data[public_config.TMP_PERSISTENT_DATA_KEY_DAY_PACK_Y_CNT])
    elseif act_id == action_config.CHAOS_PACK_Y_COIN then --铸造灵魂之匣
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_UpdateSoulPackingView)

        GUIManager.ShowText(11,LanguageDataHelper.CreateContent(30335,{["0"]=ChaosWarehouseManager.itemCountText}))
    elseif act_id == action_config.CHAOS_UNPACK_SOUL_BOX then --开启灵魂之匣
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_UpdateSoulPackingView)

        GUIManager.ShowText(11,LanguageDataHelper.CreateContent(30337,{["0"]=ChaosWarehouseManager.itemCountText}))
    elseif act_id == action_config.CHAOS_SRF_SMELT then --灵魂之匣冶炼SRF
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_UpdateSPFSmeltView)

        GUIManager.ShowText(11,LanguageDataHelper.CreateContent(30338,{["0"]=ChaosWarehouseManager.itemCountText}))
    elseif act_id == action_config.CHAOS_SRF_EXCHANGE then --SRF兑换灵魂之匣
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_UpdateSPFSmeltView)

        GUIManager.ShowText(11,LanguageDataHelper.CreateContent(30340,{["0"]=ChaosWarehouseManager.itemCountText}))
    elseif act_id == action_config.CHAOS_CHARGE_REQ then --充值
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_Recharge,args)
    elseif act_id == action_config.CHAOS_WITHDRAW_REQ then --提币
        EventDispatcher:TriggerEvent(GameEvents.ChaosWarehouse_UpdateSPFSmeltView)
    end
end
end
```

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