Cradlese Analysis Report

2023.11.22

Senna

DAMOCLES LABS

Contents

一、 Summary

Cradles was made available for download on November 15th. On November 16th, the Damocles team conducted an in-depth security analysis of the game. During the analysis, it was discovered that the game contains a significant amount of debug information that has not been removed. Based on the debug logs, it was inferred that the game's development team is from China. Furthermore, during testing, it was found that the game lacks any form of security protection. Additionally, certain logic checks in the game's communication protocol were deemed inadequate. As a result, it is not recommended for users to play or experience the game.

Security Ratings: ★ ☆ ☆ ☆ ☆

二、 Game Background

- Game Version: 20231115
- Genres & Engine：MMORPG, Unity 2021.3.x
- Possible Issues in Gameplay：
  - Illegal Movement (malicious packet manipulation through RPC for teleportation, speed hacks, etc.)
  - Speed hacking (manipulating in-game world time or using time functions in the UE framework)
  - Aimbot/Auto-lock
  - Unlimited stamina

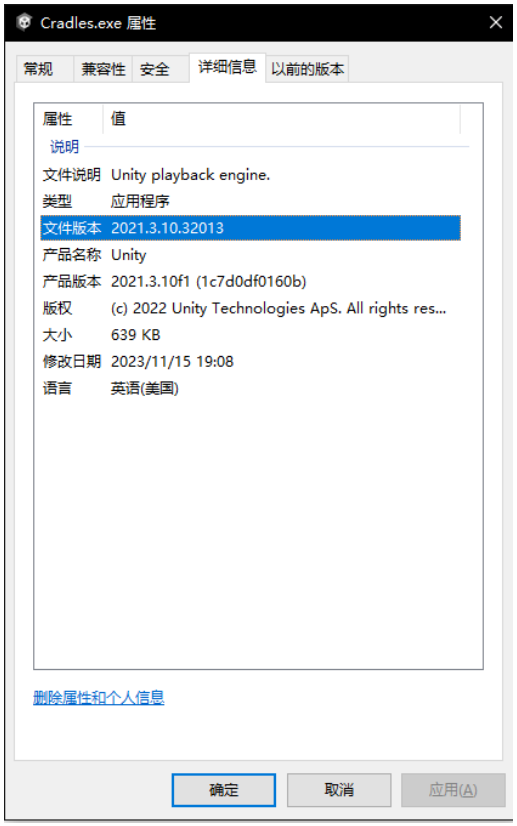
- Mining speed manipulation

三、 Game Security Analysis

Game Code Protection：

Process Analysis：

1. Since different game engines have different analysis modes, it is necessary to determine the engine used by the game after obtaining the game EXE. By identifying the basic information of the game, we can confirm that the game was developed using by Unity3d 2021.3.x

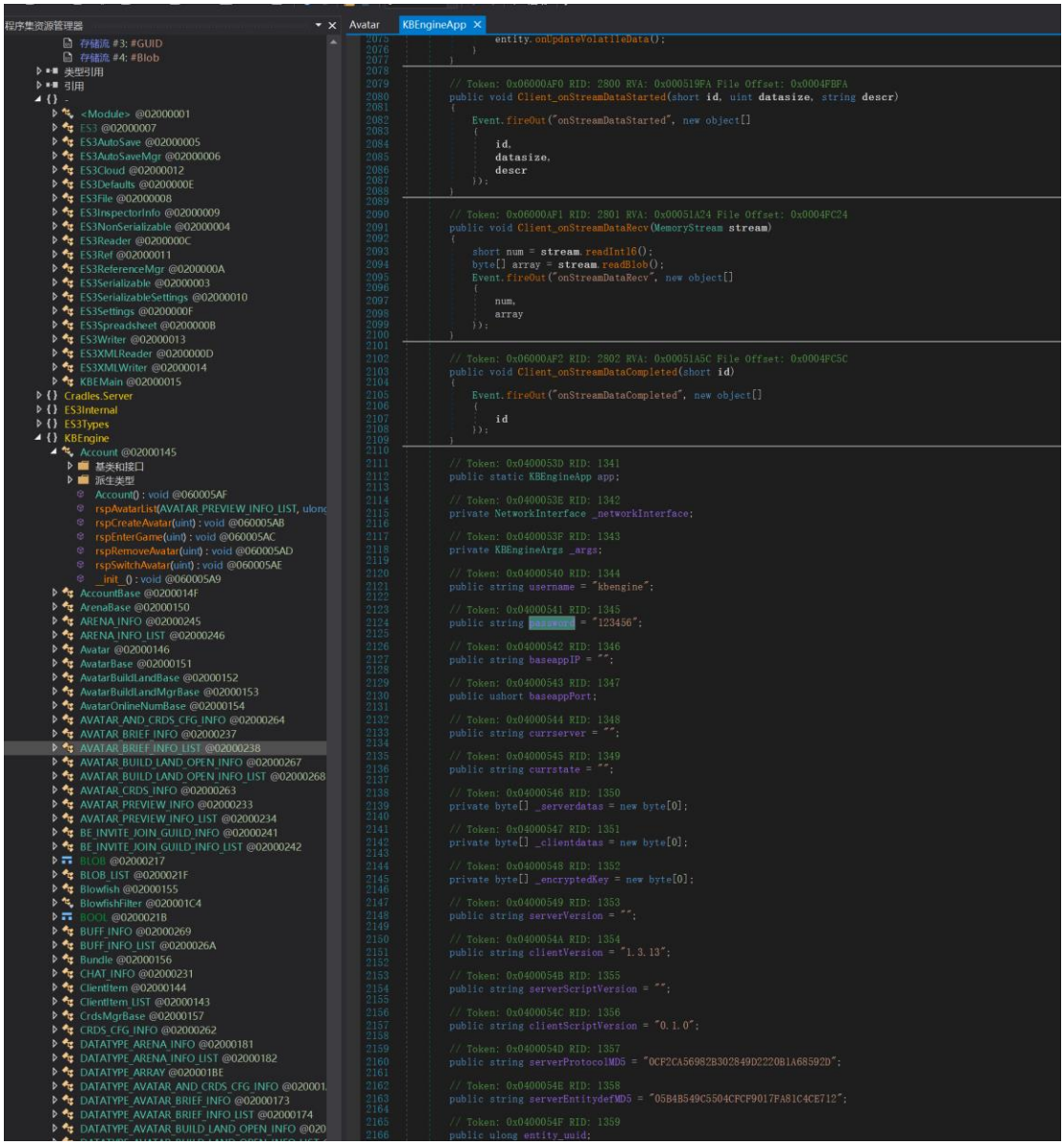


2.By examining the files released with the game, it can be determined that the game utilizes the Mono framework instead of the iL2Cpp mode for development. Games developed using this approach generally have lower overall security and are easier to analyze.

```
17                  return this.m_bag;
18              }
19          }
20
21          // Token: 0x1700001D RID: 29
22          // (get) Token: 0x060005B1 RID: 1457 RVA: 0x000264F1 File Offset: 0x000246F1
23          public Dictionary<int, ClientItem> WareHouse
24          {
25              get
26              {
27                  return this.m_warehouse;
28              }
29          }
30
31          // Token: 0x060005B2 RID: 1458 RVA: 0x000264F9 File Offset: 0x000246F9
32          public override void __init__()
33          {
34              if (KBEngineApp.app.player().id != this.id)
35              {
36                  return;
37              }
38              this._isPlayer = true;
39          }
40
41          // Token: 0x060005B3 RID: 1459 RVA: 0x0002651A File Offset: 0x0002471A
42          public int Currency(Avatar.CurrencyType type)
43          {
44              if (this.m_money.ContainsKey((int)type))
45              {
46                  return (int)this.m_money[(int)type].num;
47              }
48              return 0;
49          }
50
51          // Token: 0x060005B4 RID: 1460 RVA: 0x0002653E File Offset: 0x0002473E
52          public MONEY_INFO CurrencyInfo(Avatar.CurrencyType type)
53          {
54              if (this.m_money.ContainsKey((int)type))
55              {
56                  return this.m_money[(int)type];
57              }
58              return null;
59          }
60
61          // Token: 0x060005B5 RID: 1461 RVA: 0x0002655C File Offset: 0x0002475C
62          public override void ntfError(uint errcode)
63          {
64              Event.fireOut("OnServerErrcode", new object[]
65              {
66                  errcode
67              });
68              Debug.Log(">>>>>>>>>OnServerErrcode!!!!! :" + errcode.ToString());
69          }
70
71          // Token: 0x060005B6 RID: 1462 RVA: 0x00026590 File Offset: 0x00024790
72          public override void ntfTeamMemberSyncInfo(TEAM_MEMBER_SYNC_INFO arg)
73          {
74              if (this.teamMemberState.Count == 0)
75              {
76                  this.teamMemberState.Add(arg);
77              }
78              for (int i = 0; i < this.teamMemberState.Count; i++)
79              {
80                  if (arg.dbid == this.teamMemberState[i].dbid)
81                  {
82                      this.teamMemberState[i] = arg;
83                  }
84                  else if (i >= this.teamMemberState.Count - 1)
85                  {
86                      this.teamMemberState.Add(arg);
87                  }
88              }
89              Debug.Log("队伍成员状态返回");
90              Debug.Log(arg);
91              Event.fireOut("updateTeamMemberInfo", new object[]
92              {
93                  ""
94              });
95          }
96
97          // Token: 0x060005B7 RID: 1463 RVA: 0x00026638 File Offset: 0x00024838
98          public override void onNameChanged(string oldValue)
99          {
100             Event.fireOut("OnNameChanged", new object[]
101             {
102                 this
103             });
104         }
105
106         // Token: 0x060005B8 RID: 1464 RVA: 0x0002664E File Offset: 0x0002484E
107         public override void rspExecDuelFieldNextState(uint arg)
108         {
109             Debug.Log("决斗场状态");
110             Debug.Log(arg);
111         }
112
113         // Token: 0x060005B9 RID: 1465 RVA: 0x00026665 File Offset: 0x00024865
114         public override void rspChangeGuildIconID(uint a)
115         {
```

```
分析器
KBEngine.KBEngineApp.password : string @04000541
KBEngine.KBEngineApp.createAccount(string, string, byte[]) : void @06000A99
ClientApp.Register(string, string, byte[]) : void @06000149
KBEngine.AvatarBase.hp : uint @0400044F
  读取于
    Cradles.Character.FolkControl.DoTakeDamage(int, Transform) : void @06001077
    Cradles.Character.FolkControl.DoUpdate() : void @0600107A
    Cradles.Character.HumanControl.DoSetAnimator() : void @06001095
    Cradles.Character.HumanControl.DoStates() : void @06001093
    Cradles.Character.HumanControl.SummonPetProcess(string) : IEnumerator @06001097
    Cradles.Character.PlayerControl.DoTakeDamage(int, Transform) : void @060010D5
    Cradles.Character.PlayerControl.DoUpdate() : void @060010D9
    Cradles.Character.PlayerControl.StartCollecting() : IEnumerator @060010DE
    Cradles.Character.PlayerControl.StartInteracting() : IEnumerator @060010E2
    Cradles.Character.PlayerControl.StartLooting() : IEnumerator @060010E3
```

And the game utilizes the KBEngine protocol framework.



Therefore, it is possible to access the source code of KBEngine through open-source repositories like GitHub, as well as other publicly available resources. This can significantly expedite the process of analyzing the game.

Damocles

kbengine_unity3d_plugins / KBEngine.cs

Code Blame 3059 lines (2439 loc) · 84.9 KB Code 55% faster with GitHub Copilot

```
25          public class KBEngineApp
296                 public virtual void process()
305                         // 向服务端发送心跳以及同步角色信息到服务端
306                         sendTick();
307                 }
308
309                 /*
310                         当前玩家entity
311                 */
312                 public Entity player()
313                 {
314                         Entity e;
315                         if(entities.TryGetValue(entity_id, out e))
316                                 return e;
317
318                         return null;
319                 }
320
321                 public void _closeNetwork(NetworkInterface networkInterface)
322                 {
323                         networkInterface.close();
324                 }
325
326                 /*
327                         向服务端发送心跳以及同步角色信息到服务端
328                 */
329                 public void sendTick()
330                 {
331                         if(_networkInterface == null || _networkInterface.connected == false)
332                                 return;
333
334                         if(!loginappMessageImported_ && !baseappMessageImported_)
335                                 return;
336
337                         TimeSpan span = DateTime.Now - _lastTickTime;
338
339                         // 更新玩家的位置与朝向到服务端
340                         updatePlayerToServer();
341
342                         if(_args.serverHeartbeatTick > 0 && span.Seconds > _args.serverHeartbeatTick)
343                         {
344                                 span = _lastTickCBTime - _lastTickTime;
345
346                                 // 如果心跳回调接收时间小于心跳发送时间，说明没有收到回调
347                                 // 此时应该通知客户端掉线了
348                                 if(span.Seconds < 0)
349                                 {
350                                         Dbg.ERROR_MSG("sendTick: Receive appTick timeout!");
351                                         _networkInterface.close();
352                                         return;
```

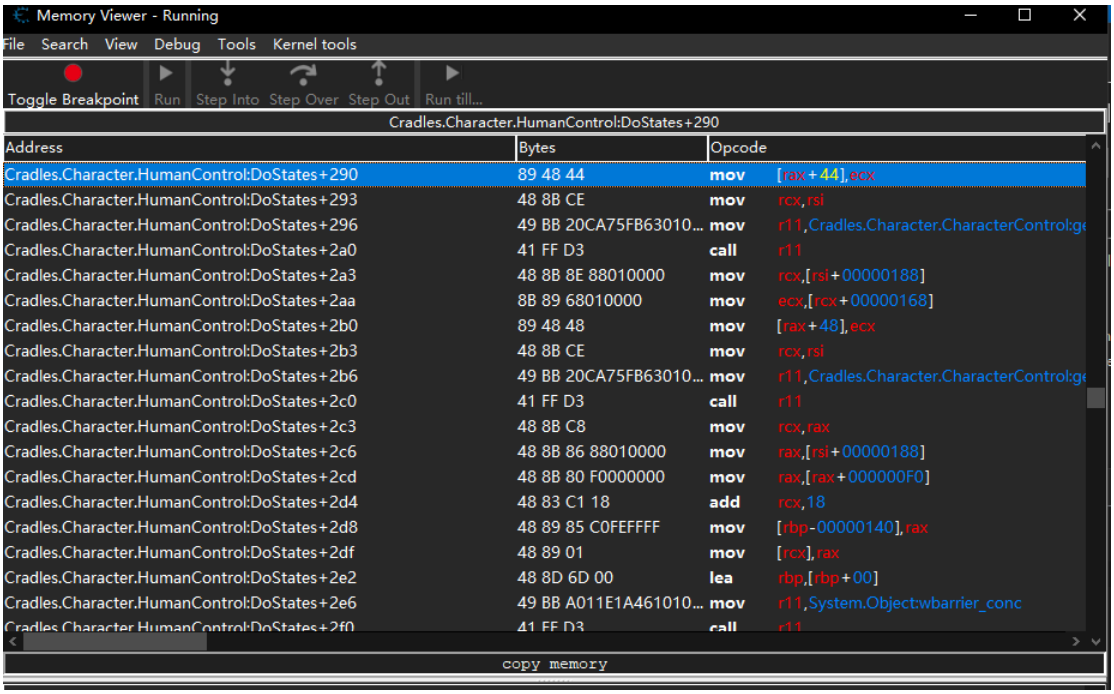
Analysis of conclusion：

Cradles receives a score of 0 in terms of game code protection, with no protection measures in place. In traditional games, custom encryption and obfuscation techniques are often used to protect the source code. Additionally, traditional games rarely utilize the Mono mode for compilation. Due to Cradles' lack of robust game code protection and its use of outdated compilation techniques, the barrier and cost for malicious players to analyze the code are very low. This creates a highly unfair situation for regular players if cheats are introduced. In areas where players can engage in free duels, malicious players have a greater advantage over their opponents.

Game Basic Anti-Cheat:

Process Analysis:

1. In terms of basic anti-cheat detection, we primarily test two aspects: anti-debugging and read/write protection in the game.
2. When attaching Cheat Engine (CE) to the game while it is running, and setting breakpoints on common functions, it was observed that the game did not exit or display any prompts.



3. By using Cheat Engine (CE) to modify the in-game stamina and HP, it was observed that the modifications took effect, and the game did not display any pop-ups or prompts. This indicates that the game lacks proper integrity checks or protection mechanisms to prevent such modifications. Modifying stamina allows for unlimited stamina/mana, and locking HP can provide an advantage within a 10-second

timeframe.

Analysis of conclusion：

1. Cradles receives a score of 0 in terms of anti-cheat capabilities, as it lacks the ability to effectively prevent cheating. This means that malicious users can cheat in the game without any restrictions..
2. The reason for focusing on anti-debugging and read/write protection in testing is that for a cheat tool, finding data and implementing functionalities typically involve debugging and memory manipulation. If the most basic protection capabilities are missing, other detection methods such as injection and hooking become irrelevant.

Game Logic Issues

Process Analysis：

For an MMORPG game compiled using the Mono framework, directly

modifying data is generally not very effective. However, in our testing, we found that certain data, such as health and stamina, can be modified and have an effect. Specifically, changes to health are effective for 9 seconds, after which the player is unable to attack monsters. This suggests that there may be a server-side restriction on the duration of damage. On the other hand, stamina modifications can persist for a longer time. We speculate that the server does not perform any checks in this regard. The reason for this speculation is that when the local character's stamina is depleted, players can pause their movement to recover stamina. If local recovery is possible, it would eliminate the need for server verification steps.

Stamina update logic:

```
case 168:
{
    ushort oldValue46 = this.stamina;
    this.stamina = stream.readUint16();
    if (prop.isBase())
    {
        if (this.inited)
        {
            this.onStaminaChanged(oldValue46);
            return;
        }
    }
    else if (this.inWorld)
    {
        this.onStaminaChanged(oldValue46);
        return;
    }
    break;
}
case 169:
```

HP update logic:

```
case 132:
{
    uint oldValue33 = this.hp;
    this.hp = stream.readUint32();
    if (prop.isBase())
    {
        if (this.inited)
        {
            this.onHpChanged(oldValue33);
            return;
        }
    }
    else if (this.inWorld)
    {
        this.onHpChanged(oldValue33);
        return;
    }
    break;
}
```

And there are many other manipulable points within the attributes related to the character in the Avatar class.

```
public ushort minPoisonDmg;

// Token: 0x04000467 RID: 1127
public sbyte moveH;

// Token: 0x04000468 RID: 1128
public sbyte moveLevel;

// Token: 0x04000469 RID: 1129
public sbyte moveV;

// Token: 0x0400046A RID: 1130
public string name = "";

// Token: 0x0400046B RID: 1131
public uint netDelayMs;

// Token: 0x0400046C RID: 1132
public uint nextSwtichRedNameTime;

// Token: 0x0400046D RID: 1133
public ushort perl0sRecoverStamina;

// Token: 0x0400046E RID: 1134
public ushort poisonResist;

// Token: 0x0400046F RID: 1135
public byte proficiencyTalentPoint;

// Token: 0x04000470 RID: 1136
public ushort robotUID;

// Token: 0x04000471 RID: 1137
public short sceneGroupID;

// Token: 0x04000472 RID: 1138
public uint sceneID;

// Token: 0x04000473 RID: 1139
public uint serverTime;

// Token: 0x04000474 RID: 1140
public byte sex;

// Token: 0x04000475 RID: 1141
public short speedIncPct;

// Token: 0x04000476 RID: 1142
public ushort stamina = 100;

// Token: 0x04000477 RID: 1143
public byte subState;

// Token: 0x04000478 RID: 1144
public int tavernRefTaskID;

// Token: 0x04000479 RID: 1145
public uint teamID;

// Token: 0x0400047A RID: 1146
public int vigor = -1;

// Token: 0x0400047B RID: 1147
public float walletHasCrdsNum;
```

Analysis of conclusion:

1. Cradles has serious overall game logic security issues, especially considering that the game involves a forced PvP mode. The low barrier to entry and high potential profits in cheat development allow for one-sided domination once a fully functional cheat is developed.
2. There is a lack of awareness of game data and detection of other vulnerable points within the game. Additionally, using an open-source engine with completely open protocols poses a high risk for games

that are susceptible to mining activities.

Game RPC Analysis

Cradles utilizes the KBEngine engine as its protocol foundation, and there are readily available online resources for reference regarding this engine.。

Reference:

1、KBEngineTechnical Overview

2、KBEngine MMORPG Demo

3、KBEngine unity3d plugins

Damocles

WEB3 Security Analysis:

Summary:

BigTime, as a blockchain game, can be divided into two parts in terms of Web3 design: the foundational BigTime token component and the in-game Web3 economic system component. This design is relatively separate compared to other games. The in-game component is responsible for token generation and NFT forging, while a fixed-circulation token contract is deployed on the Ethereum network.

Token Contract Security Analysis:

Token base information as below:

Address	0x64Bc2cA1Be492bE7185FAA2c8835d9b824c8a194
Symbol	BIGTIME
Owner	0xc3322716475fba83bfc057112247a43f1a1f2c4c**(GnosisSafe)**
TotalSupply	5,000,000,000

File 5 of 5 : BigTimeToken.sol

```
//SPDX-License-Identifier: Unlicense
pragma solidity 0.8.4;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract BigTimeToken is ERC20 {
    constructor(
        string memory _name,
        string memory _symbol,
        uint _totalSupply,
        address _owner
    ) ERC20(_name, _symbol) {
        _mint(_owner, _totalSupply * 10 ** decimals());
    }
}
```

The BigTime token contract utilizes a multi-signature wallet to mint tokens and then deploys them with a fixed supply. Due to the simplicity of the current token

contract's functionality, the basic security of the contract is considered sufficient. By observing the transaction information of the Owner wallet, it can be seen that after acquiring the tokens, the Owner wallet transferred a portion of the tokens to several other wallets.

Latest 7 ERC-20 Token Transfer Events

Advanced Filter

Txn Hash	Method	Age	From		To	Value	Token
0xcf751dc512ae532e1...	Exact Input V...	22 days 47 mins ago	Uniswap V2: BABYBigT...	IN	0xC33227...1A1F2c4C	96,952,022.7685365	ERC-20: BAB.....ime
0xb020bc506679a69f5...	Exec Transact...	24 days 7 hrs ago	0xC33227...1A1F2c4C	OUT	0x46950B...E1fd7D37	500,000,000	Big Time (BIGTIM...)
0x143212da4904b110...	Exec Transact...	27 days 12 hrs ago	0x549AD7...eF5AC1F5	IN	0xC33227...1A1F2c4C	100	Big Time (BIGTIM...)
0x2ea518f26b17d3f9a...	Exec Transact...	27 days 21 hrs ago	0xC33227...1A1F2c4C	OUT	0x46950B...E1fd7D37	100	Big Time (BIGTIM...)
0x328624a5be5f0f736...	Exec Transact...	41 days 15 hrs ago	0xC33227...1A1F2c4C	OUT	0x549AD7...eF5AC1F5	500,000,000	Big Time (BIGTIM...)
0xbe427253ccfb7c04a...	Exec Transact...	41 days 15 hrs ago	0xC33227...1A1F2c4C	OUT	0x772f5e...4c7f998A	65,000,000	Big Time (BIGTIM...)
0x8115989a2546c700...	0x60806040	48 days 5 hrs ago	Null: 0x000...000	IN	0xC33227...1A1F2c4C	5,000,000,000	Big Time (BIGTIM...)

Most of these wallets are using Safe multi-signature wallets. Based on this, it can be observed that the overall security risks associated with the token mainly come from private key leaks and whether the project team has privileged accounts. Although multi-signature wallets are used, there is still a certain risk of funds being stolen if there is a leakage of the private key of a privileged account.

Game Economy System Security Analysis:

In BigTime, players can enter the space of the Time Guardians to forge time hourglasses, charge them, and perform other actions that directly affect market balance. Some of this functionality is executed locally, although it is unclear how the game server (GS) is designed. However, this behavior is considered high-risk.

```
nistrator\Downloads\UnrealDumper-4.25-main\bin\Debug\Games\TimeGameClient-Win64-Shipping\DUMP\BP_StoryUIComponent_classes.h
oid SpawnTestEncounter(struct FDataTableRowHandle Data, int32_t Enemy Level); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.Spawn
oid SpawnTestEnemy(struct FDataTableRowHandle Data, int32_t EnemyLevel, int32_t EnemyAmount); // Function BP_StoryUIComponent.BP_StoryUIC
oid Server_CompleteObjective(); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.Server_CompleteObjective // (Net|NetReliableNetServ
oid Server_DoOtherNamedAction(struct FString ActionString); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.Server_DoOtherNamedActi
oid Server_DialogueOptionChosen(struct AActor* DialogSource, int32_t Index); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.Server
oid GiveReward(float XPAmount, struct TArray<struct FTGItemGrantDatum>& Rewards); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.G
oid ShowVendorInventory(); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.ShowVendorInventory // (Net|NetReliableNetServer|Bluepri
oid TriggerQuestEvent(struct APawn* Instigator, struct FGameplayTag EventTag); // Function BP_StoryUIComponent.BP_StoryUIComponent_C.Trig
oid BeginQuest(struct UBTS_QuestDataBase* QuestData, struct ABTS_PlayerControllerBase* Player); // Function BP_StoryUIComponent.BP_StoryU
```

```
void ToServer_FinishCraft(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString idOfActiveCraftToFinish); // Function TimeGame.TG_PlayerController_InvBase.ToServer_FinishCraft // (Net|NetReliableNative|Eve
void StartDeferredOLCurrencyBackendUpdate(); // Function TimeGame.TG_PlayerController_InvBase.StartDeferredOLCurrencyBackendUpdate // (Final|Native|Public) // @ game+0x17d6ac0
void StartDeferredNFTBackendUpdate(); // Function TimeGame.TG_PlayerController_InvBase.StartDeferredNFTBackendUpdate // (Final|Native|Public) // @ game+0x17d6aa0
void SetupEquipmentReplicators(struct ATG_PlayerPawn* ForPawn, bool InHasStoredNFTData); // Function TimeGame.TG_PlayerController_InvBase.SetupEquipmentReplicators // (Final|Native|Public) // @ game+0x17d69d0
void OnTimePieceChanged(struct UTG_InventorySetComponent* InventoryComp, struct TArray<Int32_t>& ModifiedSlotIndexes); // Function TimeGame.TG_PlayerController_InvBase.OnTimePieceChanged // (Final|Native|Public|HasOutP
void OnServer_NotifyReceivedOpenLootCurrency(struct FTGNFTOpenLootCurrencyBalanceDatum& CurrencyDatum, struct FTGInventoryItemStorageDatum& InventoryDatum); // Function TimeGame.TG_PlayerController_InvBase.OnServer_Not
void OnServer_NotifyReceivedNFT(struct FTGNFTItemDatum& NFTDatum, struct FTGInventoryItemStorageDatum& InventoryDatum); // Function TimeGame.TG_PlayerController_InvBase.OnServer_NotifyReceivedNFT // (Native|Event|Public
void OnRep_ShopInventoryComp(); // Function TimeGame.TG_PlayerController_InvBase.OnRep_ShopInventoryComp // (Final|Native|Public) // @ game+0x17d6320
void OnRep_ItemChildInventories(); // Function TimeGame.TG_PlayerController_InvBase.OnRep_ItemChildInventories // (Final|Native|Public) // @ game+0x17d6300
void OnRep_HeldSPACEInventoryComp(); // Function TimeGame.TG_PlayerController_InvBase.OnRep_HeldSPACEInventoryComp // (Final|Native|Public) // @ game+0x17d62e0
void OnRep_HeldDECOInventoryComp(); // Function TimeGame.TG_PlayerController_InvBase.OnRep_HeldDECOInventoryComp // (Final|Native|Public) // @ game+0x17d62c0
void OnRep_HeldCosmeticsInventoryComp(); // Function TimeGame.TG_PlayerController_InvBase.OnRep_HeldCosmeticsInventoryComp // (Final|Native|Public) // @ game+0x17d62a0
void OnNFTNoLongerOwnedByPlayer(struct FGuid ItemGUID); // Function TimeGame.TG_PlayerController_InvBase.OnNFTNoLongerOwnedByPlayer // (Final|Native|Protected|HasDefaults) // @ game+0x17d6200
bool OnClient_UpgradeWorkshop(struct ATG_CraftingTable_NFTWorkshopBase* Workshop); // Function TimeGame.TG_PlayerController_InvBase.OnClient_UpgradeWorkshop // (Final|Native|Public|BlueprintCallable) // @ game+0x17d607
bool OnClient_UnLockRecipeRoll(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug, int32_t RollIdx); // Function TimeGame.TG_PlayerController_InvBase.OnClient_UnLockRecipeRoll // (Final|Native
bool OnClient_StartCraft_WithGameCost(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug, struct TArray<struct FTGInventorySlotHandle>& InputItemsToFeedTheRecipe, struct TArray<struct FTGInver
bool OnClient_StartCraft(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug, struct TArray<struct FTGInventorySlotHandle>& InputItemsToFeedTheRecipe); // Function TimeGame.TG_PlayerController
bool OnClient_SpeedUp(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString idOfActiveCraftToSpeedUp); // Function TimeGame.TG_PlayerController_InvBase.OnClient_SpeedUp // (Final|Native|Public|BlueprintCall
bool OnClient_ResetRolls(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug); // Function TimeGame.TG_PlayerController_InvBase.OnClient_ResetRolls // (Final|Native|Public|BlueprintCallable) //
void OnClient_NotifyReceivedOpenLootCurrency(struct FTGNFTOpenLootCurrencyBalanceDatum CurrencyDatum, struct FTGInventoryItemStorageDatum InventoryDatum); // Function TimeGame.TG_PlayerController_InvBase.OnClient_Notify
void OnClient_NotifyReceivedNFT(struct FTGNFTItemDatum NFTDatum, struct FTGInventoryItemStorageDatum InventoryDatum); // Function TimeGame.TG_PlayerController_InvBase.OnClient_NotifyReceivedNFT // (Net|NetReliableNative
bool OnClient_LockRecipeRoll(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug, int32_t RollIdx); // Function TimeGame.TG_PlayerController_InvBase.OnClient_LockRecipeRoll // (Final|Native|Pu
bool OnClient_GetNewRecipeRolls(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString RecipeSlug); // Function TimeGame.TG_PlayerController_InvBase.OnClient_GetNewRecipeRolls // (Final|Native|Public|Bluepri
bool OnClient_FinishCraft(struct ATG_CraftingTable_NFTWorkshopBase* Workshop, struct FString idOfActiveCraftToFinish); // Function TimeGame.TG_PlayerController_InvBase.OnClient_FinishCraft // (Final|Native|Public|Bluep
void OnChildInventoriesReplicated(); // Function TimeGame.TG_PlayerController_InvBase.OnChildInventoriesReplicated // (Event|Public|BlueprintEvent) // @ game+0x1b3dc70
bool IsItemEquippable(struct FTGInventoryItemStorageDatum& StorageDatum); // Function TimeGame.TG_PlayerController_InvBase.IsItemEquippable // (Final|Native|Public|HasOutParms|BlueprintCallable|BlueprintPure|Const) // @
bool IsAllowedToChangePocketWatches(); // Function TimeGame.TG_PlayerController_InvBase.IsAllowedToChangePocketWatches // (Final|Native|Public|BlueprintCallable|BlueprintPure|Const) // @ game+0x17d5110
```

There are many RPC functions similar to this, and considering the high cost of testing, we currently do not perform any security testing. We hope that the project team can exercise strict judgment on this part of the content on the server.

About Damocles

Damocles Labs is a security team established in 2023, focusing on security in the Web3 industry. Their core services include contract code audits, business code audits, penetration testing, GameFi code audits, GameFi vulnerability discovery, GameFi cheat analysis, and GameFi anti-cheat solutions.

Their goal is to make continuous efforts in the Web3 security industry and generate as many analysis reports as possible. They aim to enhance the awareness of GameFi security among project teams and users, as well as promote the overall security development within the industry.