

Minecraft Bedrock Edition official server openerWindows Version

Plug-in development tutorial

Author:Player time: 2 0 1 9 year 1 1 month 7 day

```
NO LOG FILE! - setting up server logging...
[2019-12-08 01:47:31 INFO] Starting Server
[2019-12-08 01:47:31 INFO] Version 1.13.0.34
[2019-12-08 01:47:31 INFO] Session ID a3f02fd7-d719-48d0
[2019-12-08 01:47:31 INFO] Level Name: World
[2019-12-08 01:47:31 INFO] Game mode: 0 Survival
[2019-12-08 01:47:31 INFO] Difficulty: 1 EASY
[2019-12-08 01:47:42 INFO] IPv4 supported, port: 19132
[2019-12-08 01:47:42 INFO] IPv6 supported, port: 19133
[2019-12-08 01:47:42 INFO] IPv4 supported, port: 51459
[2019-12-08 01:47:42 INFO] IPv6 supported, port: 51460
[2019-12-08 01:47:44 INFO] Server started.
```

I. Introduction

Minecraft Bedrock Edition official server opener (hereinafter referred to as BDS) As the only official bedrock edition

The open server is relatively late compared to the bedrock version of the game client. Previously, rot

Bamboo wants to open a bedrock version of the server, there are two main options, pay in the in-game mall

Fee to buy domain service, or use PocketMine-MP(Abbreviated PMMP)or Nukkit

Third-party server opening device. For some reason, playing field service experience without accelerator

Very bad, there is always delay, connection failure or disconnection; and the third-party server update

If it is not in time, the game content cannot keep up with the official version, and you cannot enjoy the new gameplay in the first time.BDS

The emergence of solves this problem and allows players to play happily in China, but as

A new server opener is naturally not more mature than a third-party open server that has existed for many years.

Device. It cannot restrict player behavior, cannot monitor player behavior, and cannot let yuba

Convenient to manage the server, does not support extended scripts, and cannot realize the game itself through normal channels

Gameplay other than mechanics (such as command blocks). In addition,BDS Not an open source software in itself

The source code of the software cannot be obtained, and the release is still a test version.

Since it's a test version, it's in the open BDS in the software package, the program debugging information is attached,

And this information is enough for us to design the right BDS Plug-in mechanism for self-adjustment

Now we have plug-ins for the functions we want. This article will talk about Windows Version BDS Plug-in open

Hair method.

2 . Basic cognition

(One)Minecraft Bedrock Edition official server opener

by C++Written in the language, the compilation target is x86-64(Abbreviated x64) Platform, internal use

The characters used areUTF-8 Encoding, there are two versions, one is Ubuntu Version, another

Windows Version, through the analysis of the internal content, it is roughly judged that it is compiled by the same set of source code

Can be translated.

Ubuntu Version: can be in Ubuntu 18.04 Above ("above" includes the current version)

Version running, in other Linux Derivative version evenWindows Provided by the systemWSL ring

It can also run on the environment. If it prompts that the library file is missing during startup, add the missing library file that is

can. The software package does not come with a special debugging file, and the debugging information is concentrated in the main program

bedrock_server In the file, almost all function symbols can be found in the export table. Compile

There is almost no optimization program, and the reverse analysis result is very close to the source code. When the program is running

CPU Load ratioWindows The version is high.

Windows Version: can be inWindows 10,Windows Server 2016 or

Windows Server 2019 Run on. allowableLinux Platformwine Run under the environment.

Can't be inWindows 7 Previous versionWindows Run on. The main program file is

bedrock_server.exe, Debugging information file bedrock_server.pdb, With matching

pdb The file can reversely analyze the code, but the compiler has performed some advantages during the compilation process.

The program is compared with almost no optimizationUbuntu Version is more difficult to reverse analysis. RuntimeCPU

Load ratio Ubuntu The version should be lower.

If you open it directlyWindowsVersionBDS, The system will start as a consoleBDS.

The default encoding of the console isGBK, And this option cannot be changed on the property page.BDS In printing

Chinese characters will appear garbled, but BDS Normally won't print Chinese unless it's

Configuration file server.properties There are configuration items in "language=zh_CN". In the opening service

Execute the command before starting the "chcp 65001"Will change the console output encoding to UTF-8,

But when inputting Chinese characters, the encoding is stillGBK, So Chinese characters cannot be BDS

Caught.

apart from BDS The main program, its configuration file also uses UTF-8 Encoding, included in the settings

After the server name or map name with Chinese characters, be sure to end with UTF-8 Code save

file.

BDS The package itself comes with a How to Documentation, which explains BDS

How to configure and use it is very useful for novices who just started the server.BDS Currently only

Beta version, and follow the client version is constantly updated.

(2) Server starter and plug-in

due to BDS It does not support scripting or plug-in mechanism itself, we need to use technical hands

Segment to achieve the BDS The process is modified so that BDS Can accept custom

This requires us to have a dedicated launcher that can load plug-ins.

The plug-in in this article refers to the dynamic link library file in the executable file. If the launcher does not

The extension of the plug-in file is clearly specified, the defaultWindows Version BDS The plug-in is the suffix name DLL

document,Ubuntu Version BDS Extension so document.

(three)Minecraft for Windows 1 0 version

Due to reverse analysis tools, development tools and BDS Can run onWindows 1 0

On, useMinecraft for Windows 1 0 Correct BDS Testing is very convenient. single

The machine runs all software, so there is no need to worry about network connection problems. Previously purchasedJava Edition will give away

Windows 1 0 Version activation code, although there is no such activity now, but you can pass

E-commerce platforms such as Taobao buy activation codes at very low prices.

The problem that needs attention is,Windows 1 0 The version is running on Microsoft UWP platform,

UWP By default, the platform prohibits applications from accessing the local network loopback address (IPv4: 1 2 7 .0.0.1),

But if you want to open it in this machine BDS For testing, this limitation must be broken.BDS itself

Comes with aHow to The document explains how to break this restriction. if necessary,

Please read carefullyHow to File and follow the steps above.

3 . Preparation

(1) Prepare one for operation 6 4 BitWindows 1 0 Operating system computer

because BDS The requirements of the operating environment, so choose 6 4 BitWindows 1 0 Operating system

System is most suitable. But if development and testing are carried out on different computers, the

As long as the computer is installed with integrated SP1 of 6 4 BitWindows 7 The operating system is sufficient.

During development, you need to open multiple software that occupies memory. In actual testing, reverse analysis software

Pieces IDA Pro On opening Ubuntu Version BDS Will be occupied 2 GB Around the memory, by

in Windows The memory footprint of its own boot may be greater than 1 GB, So used for development

The memory capacity of the computer should not be less than 4 GB, It is recommended to be greater than or equal to 8 GB.

(2) Download and install Microsoft Visual Studio 2019 development tools

Official website: <https://visualstudio.microsoft.com/>

Hereinafter referred to as VS2019. Download any version can be used for development, including the community version.

Sign in with a Microsoft online account Windows 10 And authorized to log in to the account in the community version

Later, Microsoft allowed the community edition to be used indefinitely.

When installing, check "Use C++ Desktop development":



Then click "Single Component" and find the title "Compiler, Build Tool and Runtime",

View "C++ 2019 Redistributable Update", if not checked, please check:



In this way, we have both installed VS2019 middle Visual C++ Development environment, installed

Microsoft VC++ Run-time library. Our tools and plugins will use this library.

(3) Install Microsoft separately VC++ 2015-2019 Version runtime library

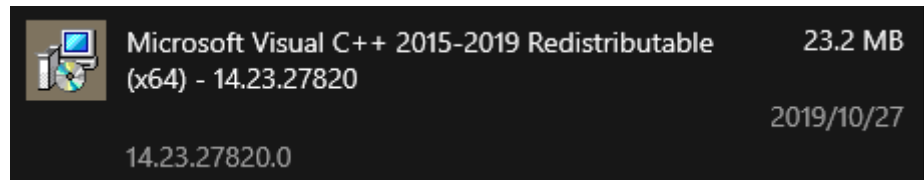
As a yuba, for players' gaming experience, they may choose to purchase an operating system for

Windows Server The series of cloud servers are launched. Relative comparison of the operating environment of cloud servers

Clean, usually not used as a personal computer, and will not be considered for installation VS2019. that

So there is no Microsoft installed VC++In the case of the runtime library, in this case you need to

The runtime library is installed separately on the cloud server.



Its download page:

<https://visualstudio.microsoft.com/zh-hans/downloads/>

Pull down "Other Tools and Frameworks" in "All Downloads" to see. In addition,

There is also Microsoft in the development kit of this tutorial VC++ 2015-2019 The installation package of the runtime library.

(4) Install reverse analysis tools IDA Pro

IDA Pro Is a very famous reverse analysis tool, we need to use it to BDS

Perform analysis. The development kit for this tutorial comes with IDA Pro 7.2 Version of the installation package. If there is new

The version appears, you can consider using the new version. The newer the version and the more features, the more efficient the analysis

Higher. Please note that you must enter the installation password during installation.

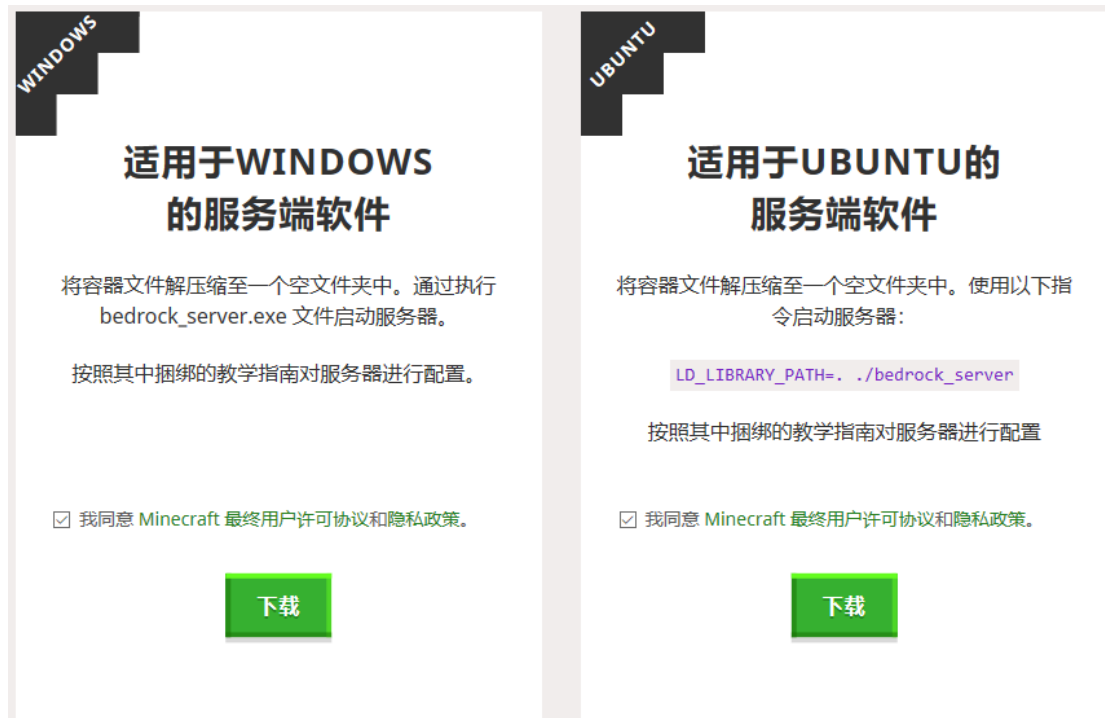
(5) Download BDS Software package

Open the official homepage of Minecraft International Edition: <https://www.minecraft.net/>

Find the help page, there will be information about the official server FAQ.

It is recommended to visit the download page directly and select the version you want to download:

<https://www.minecraft.net/zh-hans/download/server/bedrock/>



Due to the two versions BDS Basically the same set of source code is compiled and generated, in order to facilitate reverse engineering

For analysis, I recommend downloading both versions so that the two versions refer to each other during the analysis process.

(6) Download the development kit attached to this tutorial

The development kit contains the main tools used to develop plug-ins, including:

- Plugin solutions.zip: Development of plug-ins VS2019 Solution package, use after decompression
- VS2019 Open the suffix sln File to open the solution.
- IDA.Pro.v7.2.Windows.zip:software IDA Pro 7.2(Windows)Installation package.
- VC_redist.x64.exe:Microsoft VC++ 2015-2019 Offline installation package of runtime library.
- BDS Plug-in development assistant.exe: Comprehensive auxiliary tool for plug-in development, providing export

BDS Function and start of symbol information for internal debugging BDS And load the function of the test plug-in

can. Adapt to any currently knownBDS Version, and supports crash restart. Please note

Be sure to use input stop Instruction closed BDS, Otherwise it will be recognized as a server crash and

Pulled up again. Hereinafter, it is called "Development Assistant".

- `cvdump.exe`: Microsoft debugging information export tool, it is used to cooperate with the development assistant to achieve

Do not delete the export function, otherwise the export function will not work!

- `nanolauncher.exe`: Command line interface BDS Plug-in launcher for cloud services

The device is officially prescribed for taking, please use the `"/?"` parameter to view the help.

In addition, in order to facilitate the research and study of developers, the development kit may include some plug-in cases.

The content of the actual development kit shall prevail.

Fourth, the basic operation of software tools

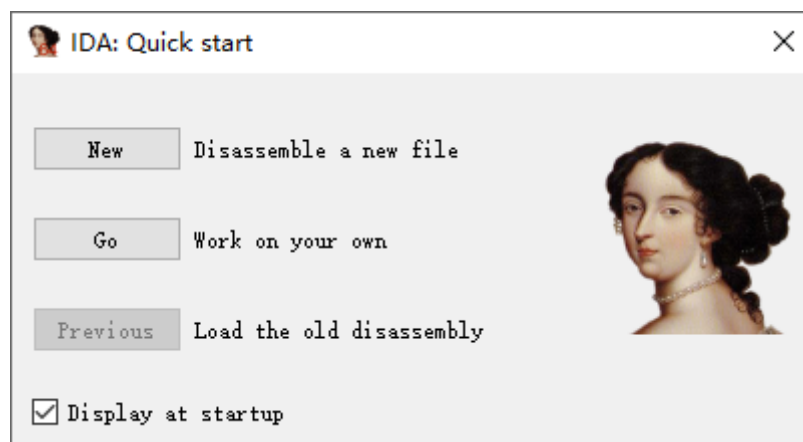
(1) Reverse analysis tools IDA Pro

After completing the installation IDA Pro After that, two shortcuts will be generated on the desktop:



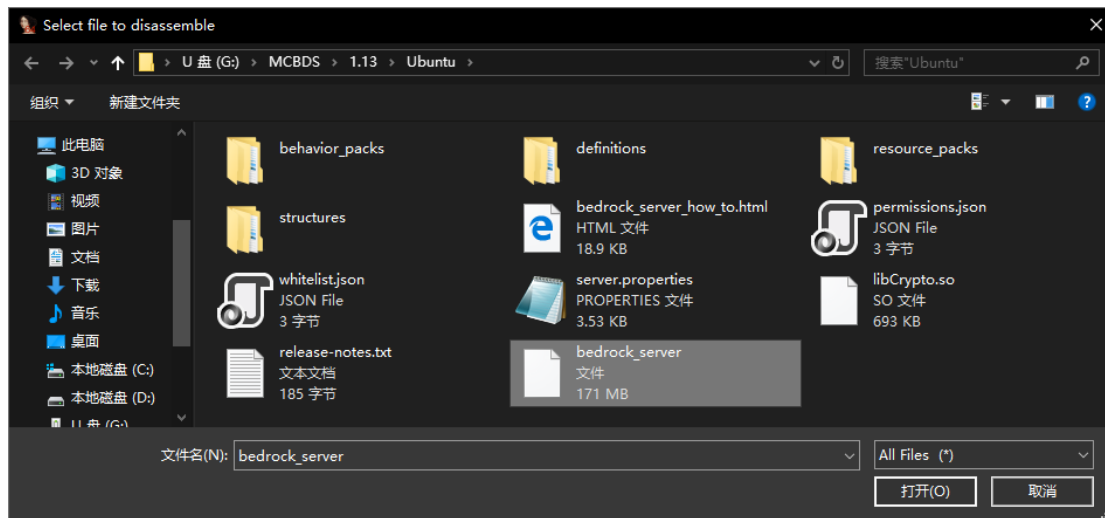
These two icons are activated separately 3 2 Bit or 6 4 Bit IDA Pro Main program, recommended here

use 6 4 The main program of the bit. Open below IDA Pro, The startup interface appears:

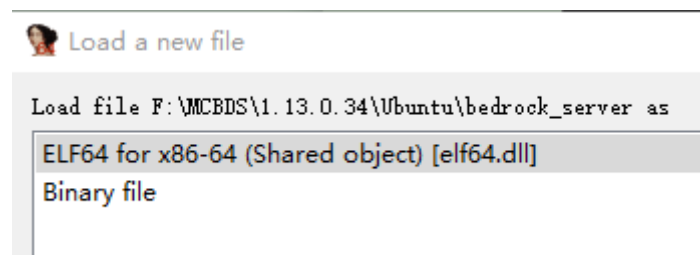


You can choose to open a file for analysis, or directly open the main interface. Here we choose

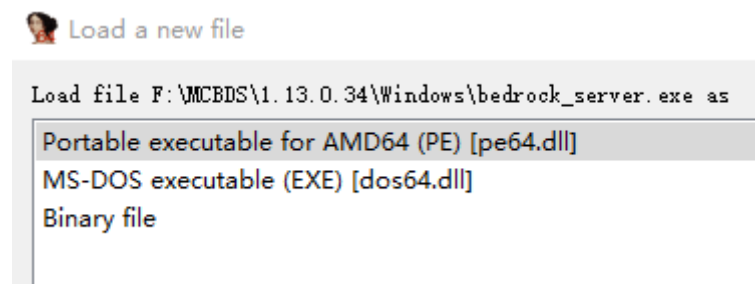
Choose the analysis result closer to the source code Ubuntu Version BDS The main program:



IDA Pro Will ask what file format to open:



The picture below is openWindows Version BDS when:



Here we all follow IDA Pro The default options are not changed because IDA Pro already

For us to do a preliminary analysis, it gives the default options closest to the real format of the file. another

External analysisWindows Version BDS A dialog box will pop up and ask if you want to load PDB Text

Pieces, this time you have to select "Yes".

IDA Pro There will be an automatic analysis stage when the file is loaded for the first time. The more content

More program files will be slower to analyze. analysisBDS The main program may take up to an hour

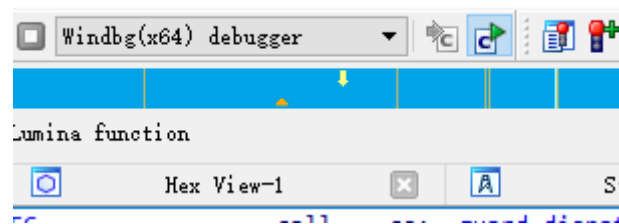
Please wait patiently for the above time.

Here is a little trick: look IDA The color bar on the main interface, except for the light yellow above

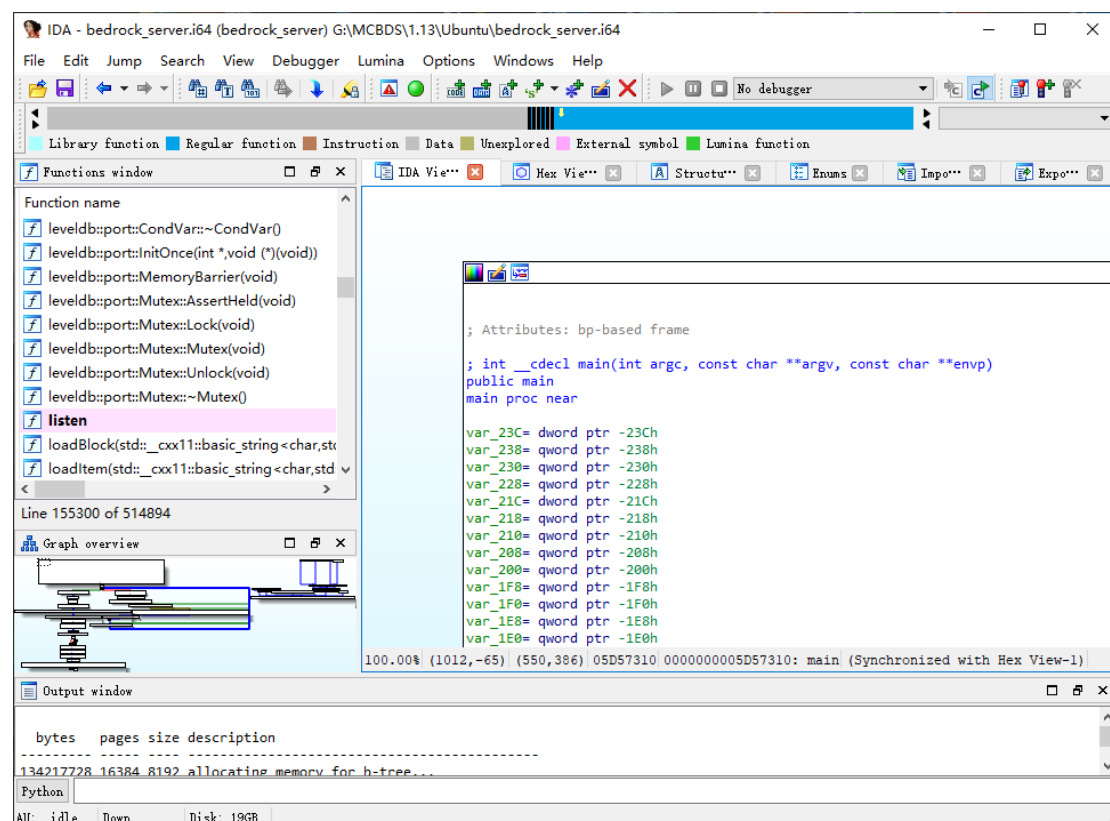
In addition to the arrow, if there is a small orange triangle below the color bar, it means IDA

Pro Still in the analysis, wait until the triangle disappears or the output window prompts the preliminary analysis

The automatic analysis is actually completed when the work has been completed. Don't hurry up until you are done!



After the analysis is complete, you can operate smoothly:



In the main interface, the upper part is the content layout color bar, and the light yellow down arrow is the current mouse

The position of the focus in the overall layout. On the left are the function list and the process structure diagram

Thumbnails of IDA Pro Has been reverse-analyzed 5 1 Ten thousand functions. Function in the middle

Process structure diagram, below is the output window.

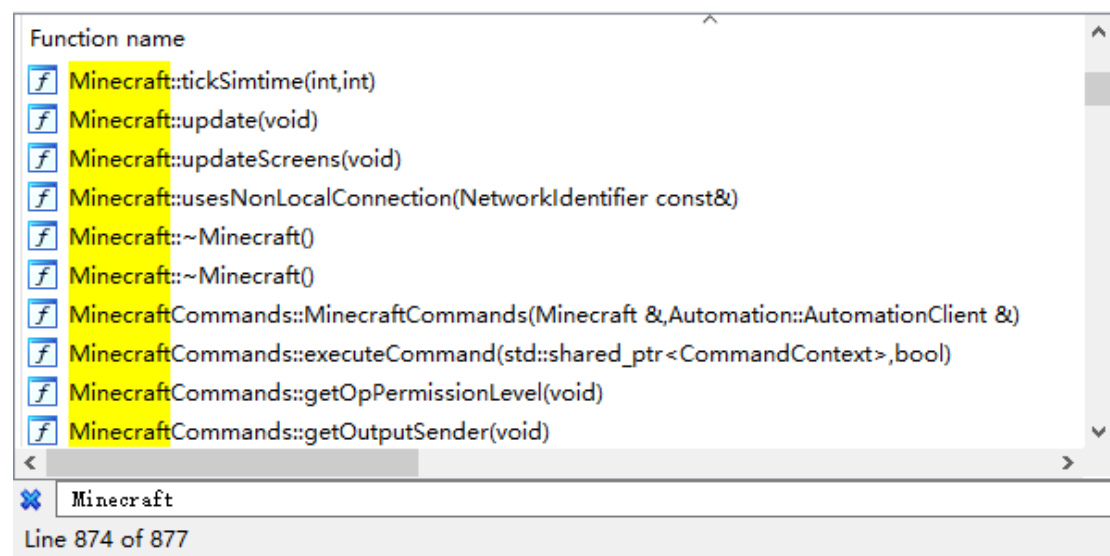
due to IDA Pro The number of functions automatically analyzed is 5 1 Many, as a person,

It is difficult to have a complete understanding of such a large number of function sets. Search at this time

Function becomes very important. Place the mouse focus in the function list, and then press Ctrl-F

Combination key, a search box will appear, after filling in the search item, IDA Pro Will be automatically filtered out

result:



If you click on the "Function name", IDA Pro Will follow the letter

The alphabetical order of the beginning of the number is sorted, which is convenient for searching in alphabetical order.

The parent order will also be the same C++ The class methods of the class are put together to facilitate the search for other similar

Other methods.

Now double-click to open a function, for example:

Minecraft::usesNonLocalConnection

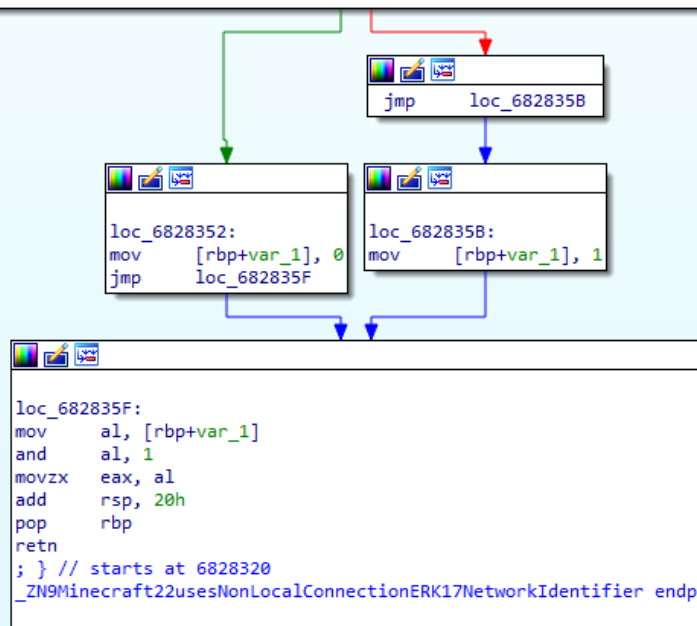
```

; __int64 __fastcall Minecraft::usesNonLocalConnection(Minecraft * __hidden this, const NetworkIdentifier
public _ZN9Minecraft22usesNonLocalConnectionERK17NetworkIdentifier
_ZN9Minecraft22usesNonLocalConnectionERK17NetworkIdentifier proc near

var_18= qword ptr -18h
var_10= qword ptr -10h
var_1= byte ptr -1

; __unwind {
push    rbp
mov     rbp, rsp
sub     rsp, 20h
mov     [rbp+var_10], rdi
mov     [rbp+var_18], rsi
mov     rdi, [rbp+var_10] ; this
call    _ZN9Minecraft17getNetworkHandlerEv ; Minecraft::getNetworkHandler(void)
mov     rsi, [rbp+var_18] ; NetworkIdentifier *
mov     rdi, rax          ; this
call    _ZNK14NetworkHandler19isLocalConnectionIdERK17NetworkIdentifier ; NetworkHandler::isLocalConne
test    al, 1
jnz     loc_6828352

```



In the flowchart, IDA Pro uses some prefixes to indicate the classes of the analyzed components

Don't, for example, function parameters will be prefixed with "arg_", and the variable of the function will be prefixed

"var_", the function that cannot know the name will automatically get a prefix of "sub_"'s name

Weigh and so on.

IDA Pro It clearly shows the outline of the flow of this function for us. But assembly language

The readability of the language is very poor, and a more intuitive way is needed to reflect the content here. Here we

select IDA Pro Built-in decompilation function. Place the mouse focus in the process view, and

Press after F5 key:

```
1 BOOL8 __fastcall Minecraft::usesNonLocalConnection(Minecraft *this, const NetworkIdentifier *a2)
2 {
3     NetworkHandler *v2; // rax
4
5     v2 = (NetworkHandler *)Minecraft::getNetworkHandler(this);
6     return (NetworkHandler::isLocalConnectionId(v2, a2) & 1) == 0;
7 }
```

same,F5 It will also name the analyzed components with unknown names. Function parameters

Numbers or variables will be prefixed with "a"or"v"the difference.

A new column is generated:Pseudocode-A, Which is similar to this function C++

Language pseudo code. The original huge flow chart has become a few lines of code, so I

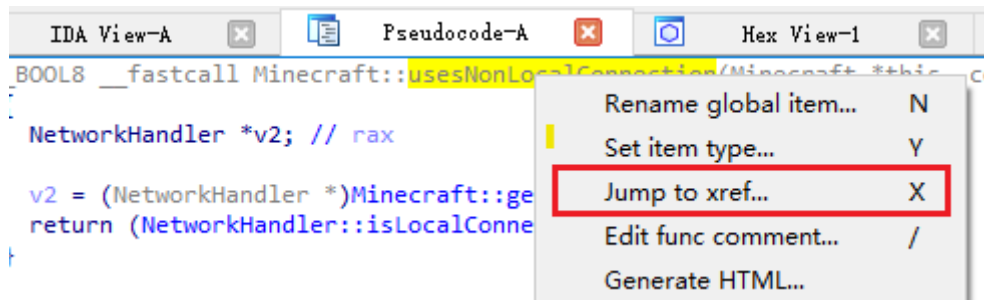
We now see the flow of this function.F5 The function will give us the effectiveness of our analysis work

The rate brings a great improvement, please use it as much as possible when analyzing.

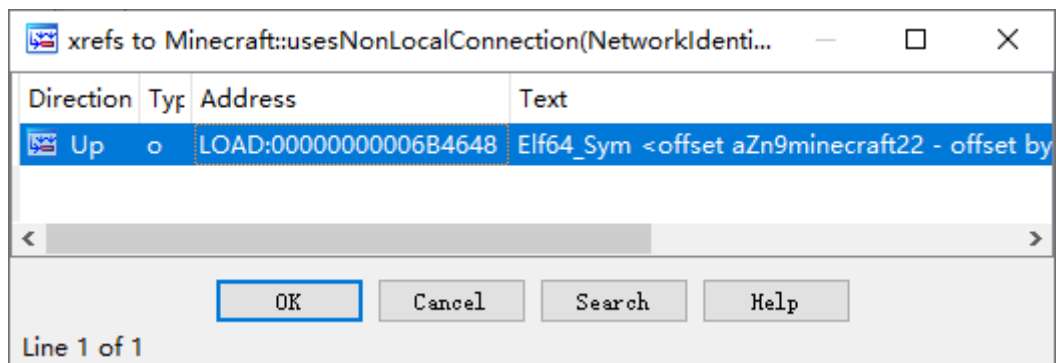
In the program, functions are not islands, a function usually has a function that it calls

Count, there are also functions that call it. To know where this function is referenced, or

After being called, we can right-click the function name and click "Jump to xref"item:



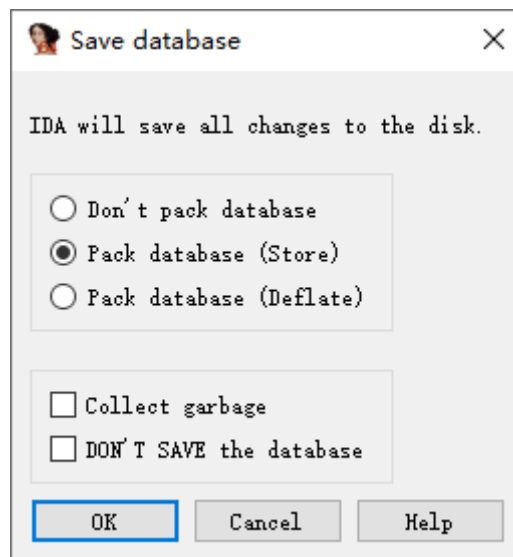
Then you can see where this function has been referenced:



"Jump to xrefThe result of "is not necessarily in the function, it may be a certain

Some memory addresses, or export tables. Double-clicking an item in the list will jump to that point.

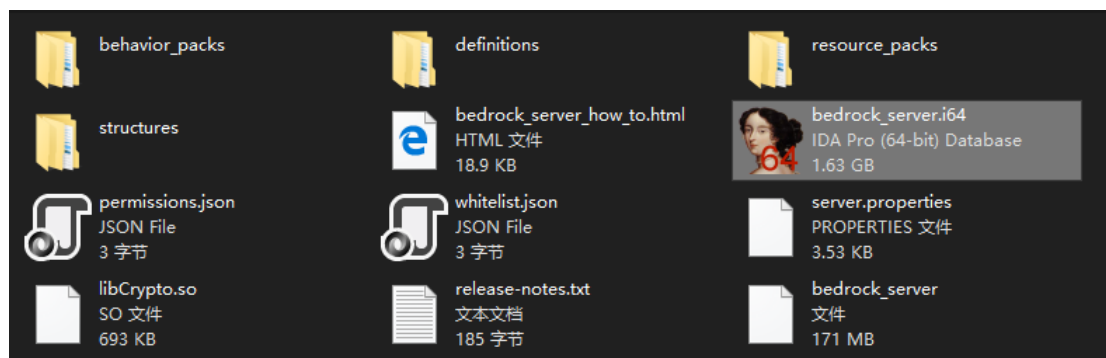
Finally, before we need to close IDA Pro when, IDA Pro will prompt us to save:



Normally click "OK" Only, the program will save the current result. If you save

In the case of insufficient storage space, you can choose Pack database (Deflate) Item pressure

Shrink and save. Here we directly click "OK" save:



If you select "DON'T SAVE the database", IDA Pro meeting

Just discard the results this time. The analysis content that has not been saved to the file will be lost. Saved

To the file, IDA Pro The original saved file will not be cleaned up, but the modification made this time

Will not be attached to the file. When it needs to be closed, if it is opened, it has been saved

The huge analysis results, then select this option directly, IDA Pro It won't be huge

The analysis results are repackaged, which greatly saves the time of closing the program and also saves the life of the hard disk.

Life.

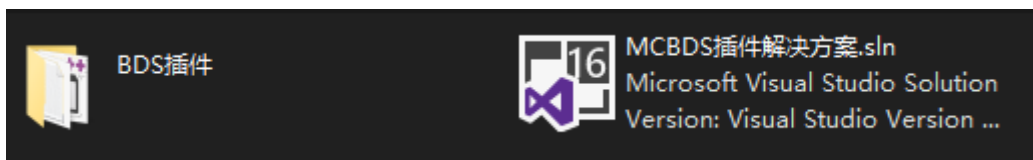
unless BDS The version is updated, otherwise the analysis results will be valid, and you can directly next time

Open the saved analysis result file to continue viewing, without re-analyzing every time.

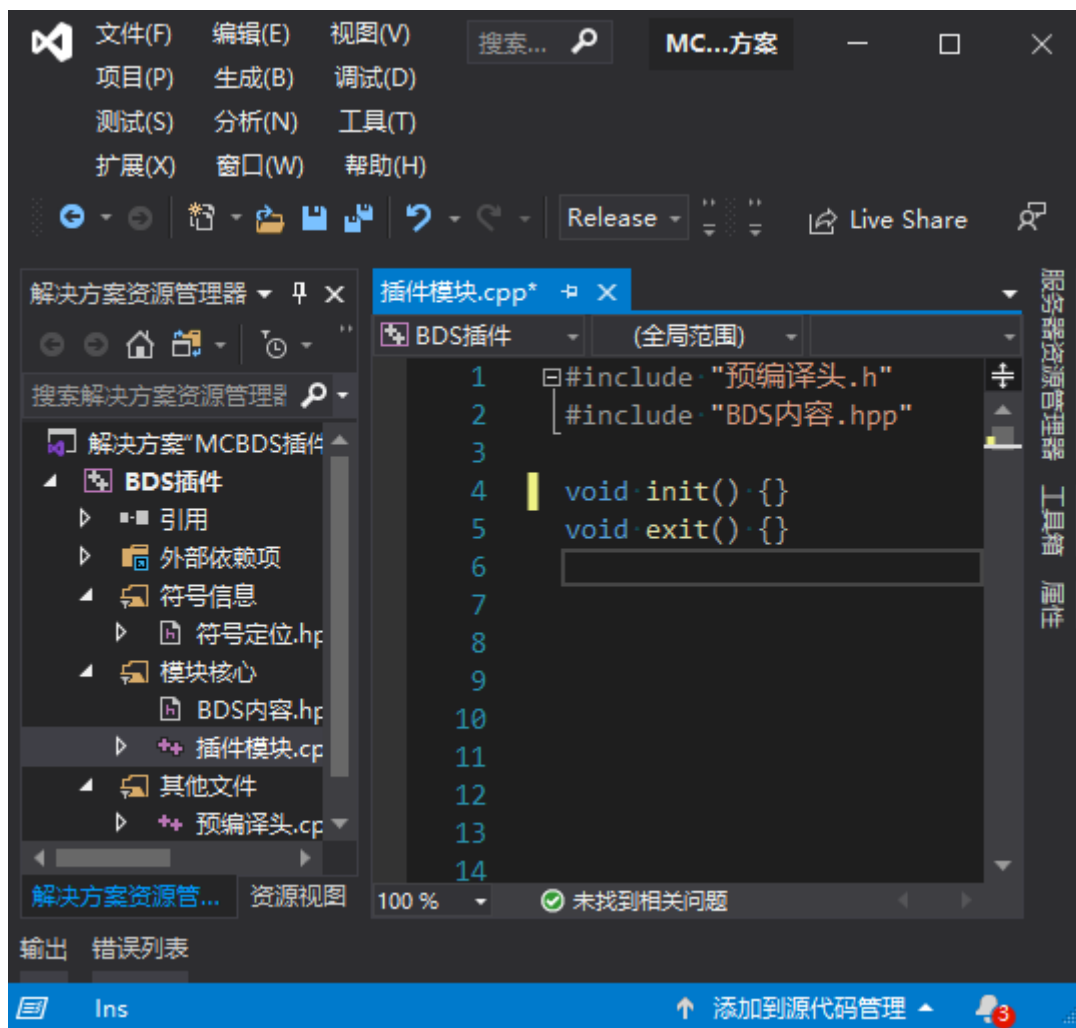
(2) Microsoft development tools Visual Studio 2019

Since the plug-in solution is already provided in the development kit, there is no need to create a new solution here

Program. Unzip the "MCBDS Plug-in solution" and open the suffix as
sln document, VS2019 Will load this solution:



After opening, we can see the main interface:



During installation VS2019 The user will be asked to choose the layout. The layout in the picture above: on the left is

Within the project C++The file list, the open file is in the middle.VS2019 There are many operations

Place, such as changing the window layout, changing the window color theme. Text font and size, etc.,

Adjust it according to the user's habits, as long as it suits you.

Because the plug-in is a dynamic link library file and is loaded with the launcher, we

It is impossible to make it directly start the debugging function for testing like ordinary programs. To test the plugin,

After compiling it, take it directly to the actual server environment.

Select the file you want to edit in the file list on the left, open it, and then write a program code

code. After the development is completed, you need to generate the plug-in, click the menu "Generate" → "Generate solution

Solution":



Then view the generated results:

```
显示输出来源(S): 生成
1>—— 已启动生成: 项目: BDS插件, 配置: Release x64 ——
1>预编译头.cpp
1>插件模块.cpp
1>正在生成代码
1>95 of 96 functions (99.0%) were compiled, the rest were copied from pre
1> 77 functions were new in current compilation
1> 0 functions had inline decision re-evaluated but remain unchanged
1>已完成代码的生成
1>BDS插件.vcxproj -> G:\MCBDS插件解决方案\x64\Release\BDS插件.dll
===== 生成: 成功 1 个, 失败 0 个, 最新 0 个, 跳过 0 个 =====
|
```

Finally, find the plug-in file just generated, and there is a complete path in the output window.



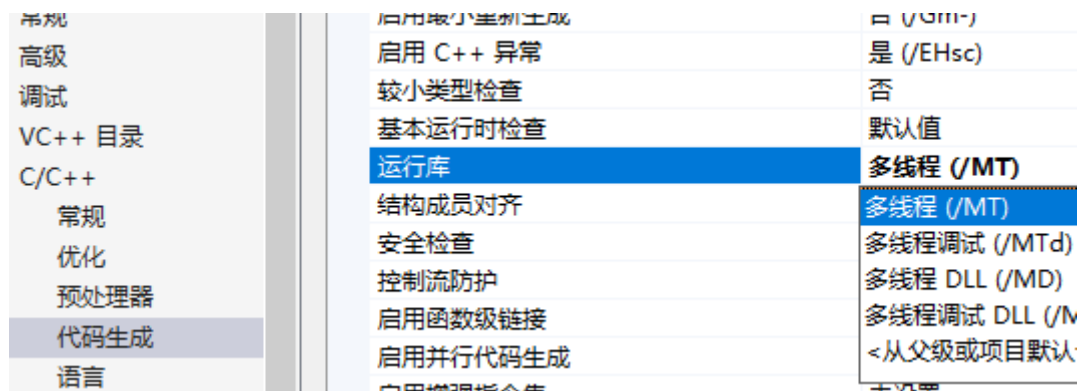
file"BDS Plugins.dll"Is the generated plug-in.

If the plug-in does not want to rely on Microsoft VC++Run-time library, in the plug-in solution, click

Click the menu "Project", click "Properties", click "C/C++", click "Code Generation",

Change the "Runtime Library" option from the original "Multithreaded DLL(/MD)" to "Multithreaded (/MT)",

Finally save.



As a price, the plug-in changed from the original empty plug-in 4 0 KB Size increased to 2 0 0 KB the above.

The plug-in generated in this way can be used in any BDS Load into the environment where it can run BDS in.

Don't care about the dependent Microsoft VC++Has the runtime library been installed in the runtime environment?

Five, the developer's technical foundation

Before plug-in development, developers need to master the following.

(1) Basic skills requirements

Because the plug-in mechanism is the launcher attached to BDS , So used to write plug-in programming

Language must be able to easily cooperate with this mechanism, then C++Naturally, language is the best choice.

The basic requirements are listed here:

- Familiar with basic computer knowledge.
- Skilled operationWindows operating system.
- Skilled use C++Programming language.
- can be used IDA Pro Reverse analysis tool.
- Correct x86-64 Have a basic understanding of the architecture.
- Ability to reverse analysis autonomously x86-64 program.

If you feel that you don't have a good grasp of the above items, you can search for relevant information on the Internet in advance

Materials to learn on their own. It is recommended to have a basic understanding of the above content before reading the rest of this tutorial.

grasp. Problems encountered in the plug-in development process can be solved smoothly.

(2) Process execution environment

The process is started by the operating system loading the application program. When starting, the operating system

The system will allocate memory space to the process. Since modern operating systems use memory paging management,

The operating system uses page tables to map blocks of physical memory pages to the process memory address space

Above, the address accessed by the process in its memory address space is not necessarily the real physical memory

Address, so I call it a virtual address (VA).

The program file contains a lot of content, such as machine code, data, resources, etc., save

In different sections of the file format. In the file, in order to save storage space, these

The contents are placed together compactly. When the process starts, the operating system will

The content is loaded into a suitable location to form a specific process memory layout, which is called a process module.

But the memory space that a process can theoretically access is very large. 64 Bit CPU Machine

Word length is 64 Bit, 64 Bit programs can theoretically access 2 of 64 Power of memory,

Much larger than the program file. The operating system can place process modules in this huge space

In a certain location, but if necessary as a dynamic link library, or security

In consideration of performance (prevention of virus attacks, etc.), the location of the process module loaded into the memory is added every time

The loading time may be different. The virtual address where the process module is located is called the base address, or base address for short.

The value of the virtual address minus the base address in the process module is called the relative virtual address (RVA).

For example, the process module is loaded into 0x400000 Address, then its base address is

0x400000, The virtual address of a function in the module (VA)for 0x401520, Then it

Relative virtual address (RVA) Is 0x401520 minus 0x400000, equal 0x1520.

(3) Plug-in execution process

Theoretically, the launcher will determine the plugin to load into BDS The way and timing of the process, but considering

Considering the actual situation, all launchers should allow the plug-in to complete the launch before the server starts.

Move so that it won't appear with BDS Conflicts caused by thread operations accessing the same address. As open

Developers should also follow this convention when writing plug-ins. When the plug-in starts

All the data that can be accessed is not passed BDS The process is initialized, do not start in the plug-in

Time to visit and wait till BDS Data generated after startup. If it is necessary to visit such

Data, please use the below mentioned Hook Technology intercepts the generation of these data C++function.

The general execution process:

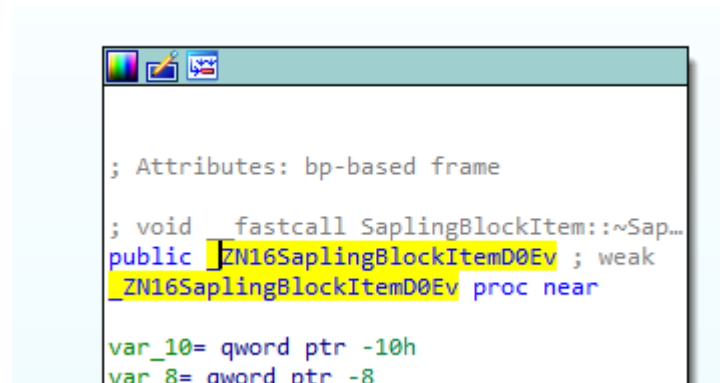
-
- ↓ The launcher starts with the option that the startup state is suspended BDS process.
 - ↓ Operating system loading BDS Process, suspend the process after completion.
 - ↓ The launcher loads the plugin into BDS process.
 - ↓ The plug-in module is initialized.
 - ↓ The launcher will BDS The process changes from the suspended state to the executing state.
 - ↓ BDS Module initialization.
 - ↓ BDS running.
 - ↓ BDS The module is closed.
 - ↓ The plug-in module is closed.
 - ↓ The process is recycled and all resources are released.

(four)C++Symbol of the function

C++The compiler generates a unique "ID card" for each entity when compiling,

This entity can be a function, a constant or a variable. The symbol of the function is the utilization function

Different compilers will have different generation rules, and the generated symbols will not be the same. inIDA Pro In the process view, move to the beginning of the function process, you can find Its symbol:



```
; Attributes: bp-based frame
; void __fastcall SaplingBlockItem::~~Sap...
public ZN16SaplingBlockItemD0Ev ; weak
_ZN16SaplingBlockItemD0Ev proc near

var_10= qword ptr -10h
var_8= qword ptr -8
```

We choose another representative function prototype (class method) to demonstrate:

```
bool Block::hasState(const struct ItemState * )
```

MSVC symbol: ?hasState@Block @@ QEBA_NAEBVItemState@ @@@Z

GCC symbol: _ZNK5Block8hasStateERK9ItemState

As can be seen MSVC (Microsoft Visual C++ Abbreviation) produces a symbol ratio GCC Produce

The symbol of life is longer, and relatively, the more information it stores. because MSVC Symbol meeting

Almost records the specific type of each parameter of the function or its return value, and GCC produced

Symbols do not have such information, so when encountering non-built-in types, such as

"std::string" Types of, MSVC The symbol can restore the parameter type very well, and

GCC The symbol cannot.

BDS Attached to the program PDB The relative virtual address of each symbol is stored in the file

(RVA), and BDS Module base address can be provided by the platform API Function acquisition, so it's easy

The content behind the symbol (function, constant or variable) is easily accessible BDS Process memory space

Virtual address. This is also the main way for the plug-in to obtain the target address.

(Fives)C++Function calling convention

in C++Among them, the function is an important part of the class, it receives the passed parameters, Perform its function, and can provide a return value after the execution is complete. The function asks to receive The parameters are called "formal parameters", and the caller gives the function according to the formal parameters of the called function The variable or value assigned by this function is called the "actual parameter". After being compiled by the compiler, The function becomes a series of machine code that is difficult to read.

When generating machine code, the compiler will comply with a series of target machine platform-related The rules to generate machine code, including the calling convention. The calling convention stipulates that in the special set CPU Under the execution environment, what kind of agreement should a program composed of machine code comply with? Call a function that is also compiled into machine code correctly. The main contents of the calling convention are:

- At the beginning of the call, how does the caller pass the parameters to the called function correctly?
- Which caller's environment should be saved by the called function and restore the scene when returning.
- After the called function is executed, how to correctly pass the return value to the caller?
- Who will release the system stack space used in parameter passing.

Ubuntu Version BDS by GCC Compile it, use System V AMD64 ABI provide The calling convention.Windows Version BDS byMSVC Compiled, using Microsoft x64 transfer Agreementx86-64 platform fastcall).

here has IDA Pro Help us analyze, in most cases where reverse analysis is needed No need for the developer to analyze it himself only atIDA Pro of F5 Function cannot be analyzed normally In the case of out of the function process, the developer needs to analyze it personally.

(six)C++Function prototype

to get BDS The prototype of an internal function, these methods are used to obtain information:

- The symbol of the function.
- Analyze the function call relationship.
- Analyze the internal flow of the function.

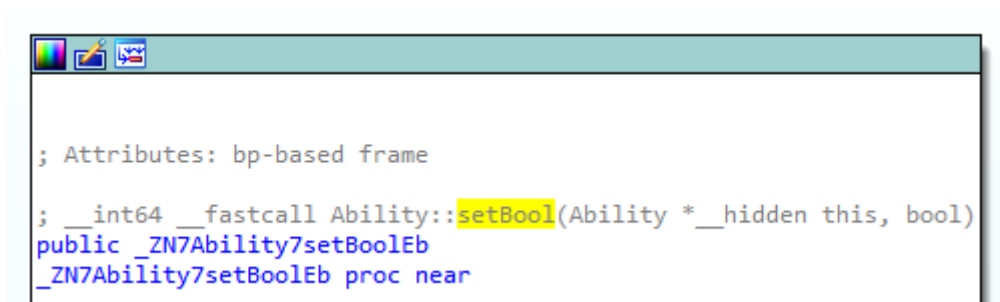
In general, the symbol of a function is a major and very accurate way to obtain parameter information.

way. However, because the compiler may not generate the corresponding function prototype information provided by the symbol

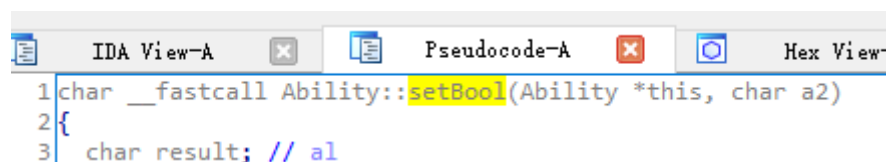
Machine code, so the symbol may not accurately represent the function prototype. At this time, it needs to be equipped

Close the other two analysis results.

IDA Pro The function prototype after symbol resolution is displayed above the symbol in the function head:



But when IDA Pro use F5 After function decompilation analysis:



visible,F5 The function does not use the type information carried by the function symbol, but only

It is the result of analysis. look inF5 Pay attention when analyzing.

See the function prototype derived from symbolic analysis:

```
__int64 __fastcall Ability::setBool(Ability
* __hidden this, bool)
```

The first "__" that appears from left to rightint64"Is the return value of the function, becauseUbuntu

Version BDS The return value information provided by the symbols used is limited, and the calling convention also stipulates that the return value is used

Pass by register (x86-64 The total length of the register under is 64 Bit), where "__int64"

Such a very general type represents the return value type.

The following "__fastcall" is the calling convention of this function, because x86-64 Under the platform

GCC Only one calling convention is used to generate the machine code of the function, regardless of IDA Pro for us

Marked as "__cdecl" or "__fastcall" Or is it "__usercall". We all

Don't worry about other calling conventions, just ignore them here. But in the rare

In this case, the compiler will not generate functions according to the standard calling convention, resulting in decompilation.

If it looks very strange, this kind of trap will be introduced in a later chapter.

Next "Ability::setBool" Means that this function is C++ Class method is compiled while

Come, this class name is called "Ability", the class method name is "setBool". Need to note

What is meant is, C++ A class is allowed to have multiple class methods with the same name, as long as the parameters between them

Or the return value type is different.

The last parenthesis is the function parameter list of the function. At this time, IDA Pro already

Mark "__hidden", which means in the class method this The pointer parameter is implicit. The second parameter

Number type "bool" Is the information analyzed from the symbol.

Now let's take a look F5 The result of the function analysis is the same as the above analysis from the symbol

The results are similar, the only difference is the second parameter and return value of the function it analyzes

Types are char Type, and the second parameter type obtained from the symbol is bool,

In addition, the name of this class method is also called getBool, So we conclude bool Type in

This function is based on char The form of the type is passed. This function is being compiled into a machine

The real function prototype after the code is:

```
bool Ability::setBool(Ability* this, bool)
```


(Seven)C++Class memory layout

C++Class in C++Very common in programs, classes can be regarded as a kind of closely related to each other

Variables or functions are wrapped together, and everything inside is a member of the class. Save

Lies in C++The "functions" in the class are called "class methods".

During the compilation process, for the member variable part, the compiler combines these parts into one

A structure. For non-static class methods, the compiler first adds one to the head of the parameter list

The parameters of the class pointer type are then generated in the same way as ordinary functions. For static class methods

Then it is directly generated according to the ordinary function method. For the virtual method in the class, the compiler will first be like

The corresponding function is generated in the same way as the ordinary class method, and a virtual table called "virtual table" will be built for this class.

(vtable)" structure, put the virtual function address in it, and then in the member variable

A virtual table pointer variable is added to the header of the combined structure to save the address of the virtual table

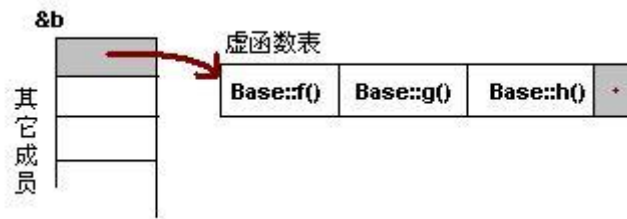
Go in. E.g:

```
class C {  
    public:  
        int a;  
        int add_a(int b) {return a + b;}  
};
```

After being processed by the compiler, the effect is similar to:

```
struct C {int a; };  
  
int C__add_a(C* p, int b) {return p->a + b;}
```

This is the virtual table icon:



All class methods will become de facto functions after being processed by the compiler

The machine code section in the executable file. You can use the same calling method as a normal function

Go call it.

C++ Allow classes to inherit from other classes, and also allow classes to contain members of other classes

Variable when C++ One class inherits another class, then this class has to inherit the parent class

All content, the content of the sub-category will only be more than the parent category, not less than the parent category. At the same time due to C++

Allow subclass pointers to be converted to parent class pointers, and to access the resources of the parent class, that is, subclass inheritance

After the parent class, its internal structural layout must completely accommodate the parent class' s structural layout, and

You cannot change the layout of the parent class, otherwise the class method of the parent class is converted from the subclass pointer

The parent class pointer access finds that the structure layout has changed, which violates the rules. When the subclass is initialized

At that time, because it contains the parent class, it needs to use the structure of the parent class in its own constructor

Function to initialize the parent class part. in C++ When the class uses single inheritance, the parent class always puts

At the head of the subclass, the subclass shares a virtual table with the parent class, so the subclass pointer is converted to

When the parent pointer is used, the memory address saved by the pointer is unchanged. When the subclass is initialized

Will overwrite the virtual table address in the parent class with its own virtual table address, but this must happen in

After the parent class is initialized, it is necessary to ensure that the parent class can be initialized normally.

C++ The member variables in the class are also initialized in the constructor, and can also be used in the constructor

Their constructors (if any) can also be found in numbers, but there is no experience at all.

Programs optimized during compilation time will not appear before the class initializes the virtual table.

(Eight)C++function hook technology

A way to modify the machine code at the beginning of the function at runtime or the function address in the function table

Realize the operation of transferring all calls to the function to the target function, which is plug-in interception BDS

The key way of original function call to realize custom function.

If there is a three-function C++Program fragment:

```
int a(int v) {return v + 4 ;}  
  
int b(int v) {return v * 3 ;}  
  
int main() {return a(5);}
```

C++The normal execution sequence of the program is:

```
↓ functionmain Begin execution;  
  
↓ carried out a(5);  
  
↓ carried out 5 +4 And return the result 9 ;  
  
↓ functionmain return 9 .
```

But use the function Hook Technology, modify function a The machine code at the beginning makes it turn

transfer b, So the revised program flow becomes:

```
↓ functionmain Begin execution;  
  
↓ carried out a(5);  
  
↓ Execution function a Transfer instruction at the beginning to the function b;  
  
↓ carried out b(5);  
  
↓ carried out 5 * 3 And return the result 15 ;  
  
↓ functionmain return 15 .
```

This tutorial uses the official Microsoft plug-in C/C++function hook Library Detours,Features

Mature, the objective function is hook The machine code before modification will be preserved, and it will be saved losslessly.

The original function, and allows to call at any time.

For in C++For the function in the code, its function name can be implicitly converted to the function

Number of virtual addresses, and for functions in other modules in the process memory address space, you need

To add through the module base address RVA Value to get its virtual address.

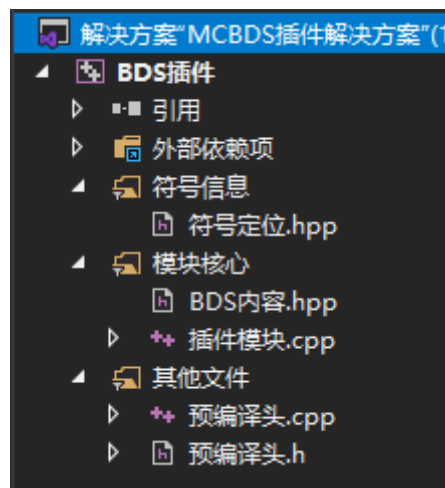
Six, the basic operation of the development process

Before officially writing a plug-in, you need to understand one of the processes of using plug-in solutions.

These necessary operations, and learn to use them.

(1) Understand the files in the plug-in solution

use VS2019 Open the plug-in solution and look at the Explorer window:



Here, the "symbol information" and "module core" filter directories are frequently used to develop plug-ins

For the files used, the contents of the files under "Other Files" do not need to be changed, but do not delete either.

The purpose of the main file here is:

- Symbol positioning.hpp: This file should be generated by the development assistant, please do not modify it manually

document content. Development Assistant fromBDS Attached to the main program PDB Extract all symbols from the file

of RVA Value and export the result to this file. Each time the operation, the content of the file is completed

Full coverage once. The generated file contains "MSSYM_"Integer type changes

Quantity whose value is symbolic RVA value.

- BDS content.hpp: Specially used to store pairs BDS Reverse analysis results of internal processes

file. due to Windows Version BDS There is a certain optimization during compilation, and there are functions that are optimized

The situation disappeared. If you must use these missing functions, please add them and store them here

in.

- Plug-in module.cpp: The main module where the plug-in code is stored, please replace the key of the plug-in

Place the code here. There are two functions inside: void init() versus void exit(), Respectively in the module

Called when starting or exiting. If there is some data that needs to be loaded when the module is just loaded

Initialization, you can put the initialization code in init Inside the function; what needs to be done when the module exits

Emotions, such as saving a file, you can put it in exit Function.

(2) Use the development assistant to obtain symbols RVA variable

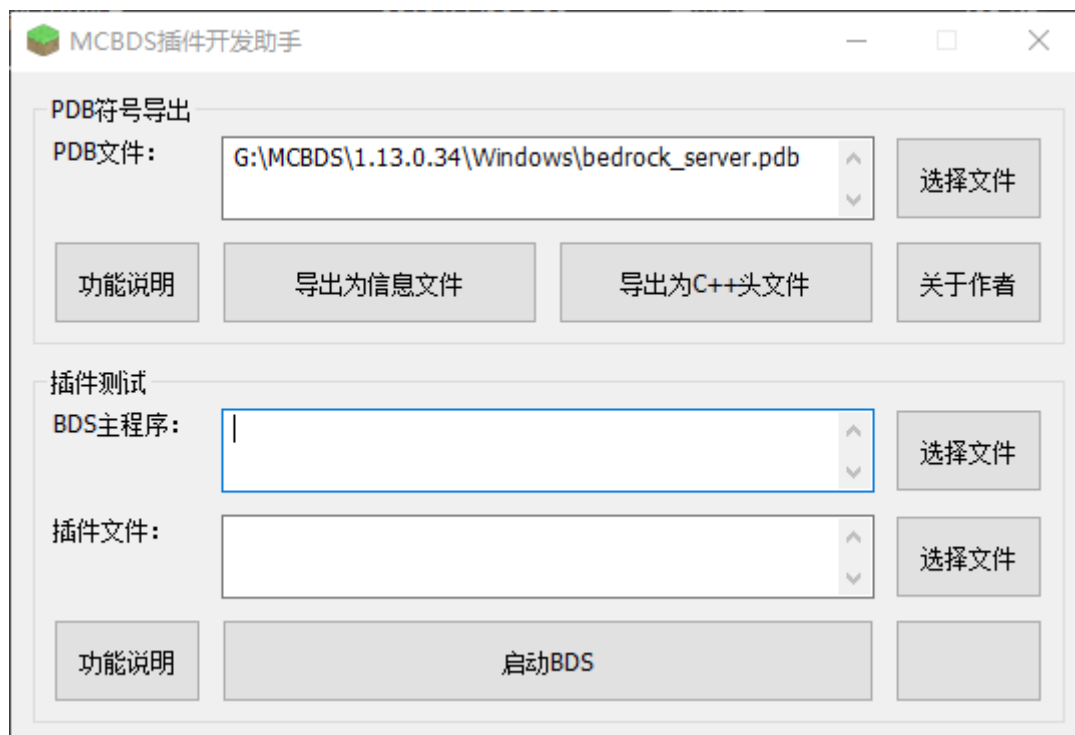
PDB All symbols and corresponding RVA Value, but there is no guide

Before we went out, we couldn't directly get what we wanted RVA value. In this step we use tools

Extract the internal symbol information and save it in a text format that is convenient for analysis:

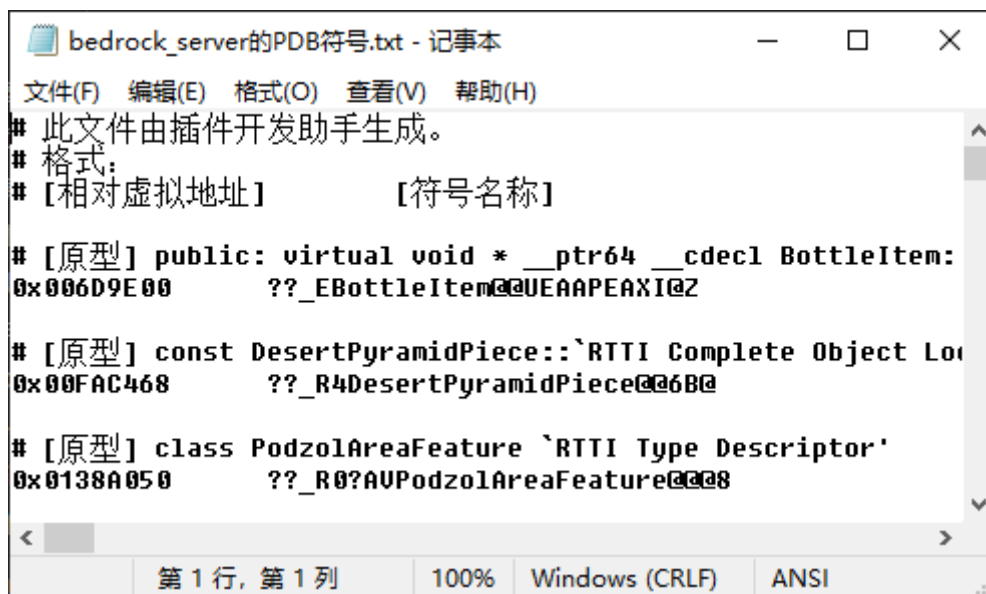
turn on "MCBDS Plug-in Development Assistant", click "PDB File" edit box on the right

"Select File" button to select the corresponding PDB file:



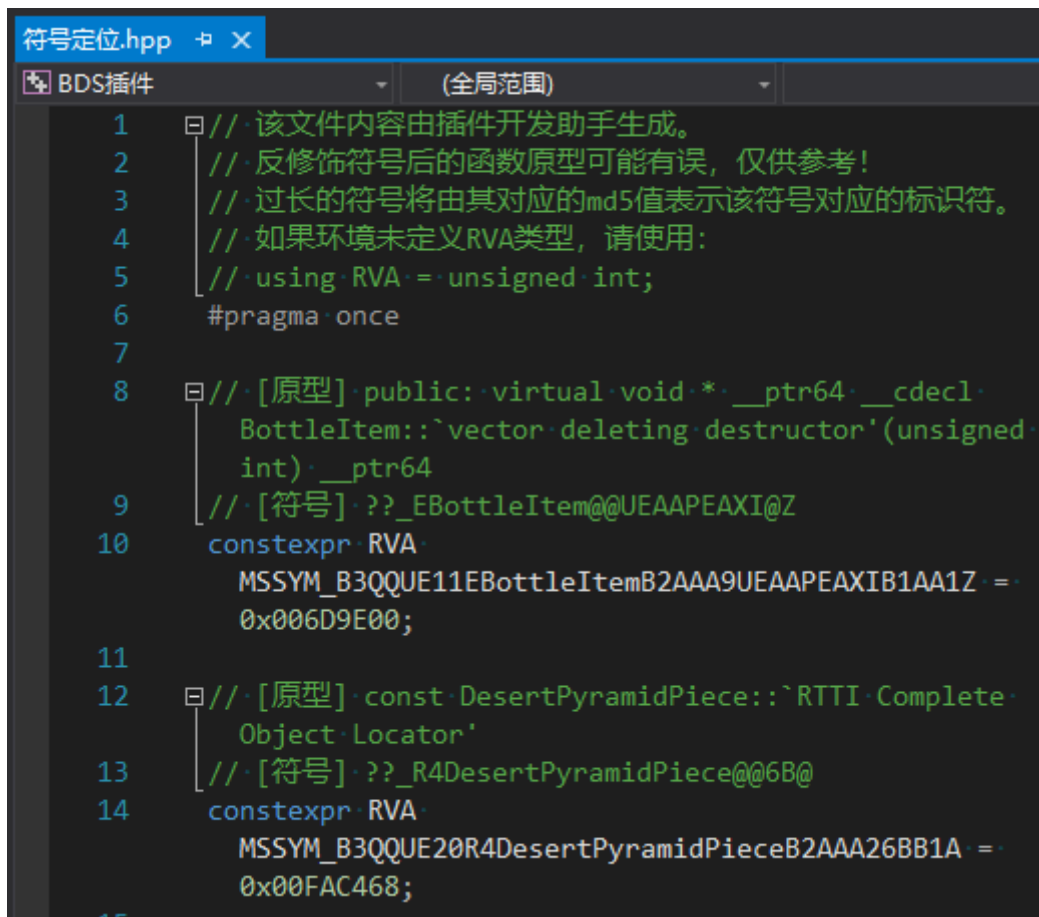
Then click the "Export Information File" button, it will export to an information file, format

Is a text, which records the information of all symbols:



If you click "Export as C++Header file" button, it will export C++Header file, pass

Normally, save it to the "Symbol Positioning." in the plug-in solution.hpp"file:



```
符号定位.hpp
BDS插件 (全局范围)
1 // 该文件内容由插件开发助手生成。
2 // 反修饰符号后的函数原型可能有误，仅供参考！
3 // 过长的符号将由其对应的md5值表示该符号对应的标识符。
4 // 如果环境未定义RVA类型，请使用：
5 // using RVA = unsigned int;
6 #pragma once
7
8 // [原型] public: virtual void* __ptr64 __cdecl
9 // [符号] ??_EBottleItem@@UEAAPEAXI@Z
10 constexpr RVA
11     MSSYM_B3QQUE11EBottleItemB2AAA9UEAAPEAXIB1AA1Z =
12     0x006D9E00;
13
14 // [原型] const DesertPyramidPiece::`RTTI`Complete
15 // [符号] ??_R4DesertPyramidPiece@@6B@
16 constexpr RVA
17     MSSYM_B3QQUE20R4DesertPyramidPieceB2AAA26BB1A =
18     0x00FAC468;
```

Unlike the previously exported information file, this header file will symbolize RVA Value make

Save it with a constant variable, and attach its corresponding prototype and symbol respectively.

In addition, with IDA Pro The analysis result file is the same, as long as BDS The version has not changed,

The exported files are all valid and do not need to be re-exported every time they are used.

(Three) use THook Series macro writing Hook Code

THook Series macros are a type of Hook Function macro, it provides a simple

Way to achieve complexHookRequirements and processes, Can make developers write an ordinaryC++

Function as easy to implementHook effect. It is fromMCMrARMTransplanted there, this

Here is the official tutorial:

<https://github.com/minecraft-linux/server-modloader/wiki/Hooking-API>

Used in this plug-in solution THook The series of macros is slightly different: the use of symbols

The name is changed to the corresponding symbol RVA value. Others retain the style of the original tutorial, if

Developed Ubuntu Version BDS Plug-in, you should be familiar with its use.

here THook The way to use the macro is:

```
THook(Return value type, function RVA, Formal parameters...) {  
  
    Objective function code  
  
}
```

THook First use the function RVA Find the location of the original function, and then use the brackets

The given return value type and formal parameters, plus the code in the enclosed curly braces, construct a

Objective function, and finally use Hook The original function of the operation is transferred to this objective function.

In this way, all calls to the original function will be transferred to the target function code here.

Return after processing by the objective function.

Sometimes we have a functionHook The operation is just to add a little bit to its original function

Change, such as printing the value of a certain variable to determine if there is a programming error, not

Is to directly deprecate its function, so we need to be in this function Hook After that

The ability to continue to use the original function. THook The series macro was designed with this in mind

In this case, it has a built-in function pointer variable original. Call the function it points to,

You can use the functions of the original function.

in case THook The return value type and formal parameters set in the series of macros and the original function

Can be exactly the same, it can ensure that the objective function is perfectly called and the same as the original function.

Return without any errors, but the reality is always not so ideal. Once you encounter the type letter

If the information is incomplete, it will not be consistent. In this case, the objective function we provide

To stay at C++Compatibility in function calling conventions, otherwise CPU When the program is executed, it will

An unpredictable value disorder has caused the program to crash. This is a very serious error.

It must be corrected if found. Here are some suggestions:

- If you encounter a built-in type (char,int,bool,double Etc.) or composed of them
Pointer type (char*Etc.), it is enough to use the same type in the objective function.

- If you encounter a pointer of an unknown structure, you can use any type of pointer instead, including
void*, If you just pass this actual parameter, use void*Will suffice.

- Floating point type (float,double Etc.) can' t be replaced by other types, because this

This type of parameter is passed to the called function through a dedicated way. But in passing by

The structure of floating-point number type will not be passed through a special way.

THookSeries of macros use the constructor of global structure variables to doHookwork,

So it's Hook The operation is to precede the plug-in init Function execution. apart fromTHook,also

There are three other macros that can be used. allTHook series:

- THook: The target is a normal function.

- TStaticHook: The goal is to have a static class method in a complete class definition.

- TInstanceHook: The target is a class method with a complete class definition.

- TClasslessInstanceHook: The target is a class method without a complete class definition.

If you want Hook More than twice in the same position, then please use the 2 The macro,
such as:TStaticHook2.

Here before we use it again Block::hasState Function to demonstrate THook Macro compilation

write. The function corresponding to the symbol we have obtainedRVA Information, the function prototype has also been

confirm. For the symbolTHook The code can be written like this:

```
THook(bool, //Return value type from the original function
```

```

MSSYM_B1QA8hasStateB1AA5BlockB2AAA4QEBAB1UE14NAEBVI
temStateB3AAAA1Z, //Primitive RVA variable

void* class_this, void* pItemState //Formal parameters (compatible)
) {

return original(class_this, pItemState);

}

```

This code adds void* class_this Implied this Pointer while using void*Type replaces the original Block*versus ItemState*Types of.

This code behaves only in Hook function Block::hasState After intercepted Its two parameters (this pointer,ItemState Pointer) and forwarded to the original function for execution, Return the value it returns (bool Types of). No other actions were taken. If needed What action does it have, in THook Just write the code in the provided code block.

apart from THook Series of macros, used in functions Hook versus UnHook Function can also Achieve the same effect. The parameters and return value of the objective function of your choice still have to be the same as the original function Are largely the same. Two realizationsHook The way of function can coexist, but the operation is the same This function may bring unpredictable errors, please try to avoid this situation.

(4) Use SYMCALL Macro call symbol

During the development process, there are often need to use BDS Internal functions to do some work In the case of using SYMCALL Series of macros can be realized.

If there is a return value:

```
Return value=SYMCALL(Return value type, function RVA, Actual parameters...);
```

If there is no return value:

```
SYMCALL(void, function RVA, Actual parameters...)
```

Pay attention to THook Different, the parameter list here is the actual parameter list, which is about to

The list of parameters passed to the function, and THook What needs to be filled in is the list of formal parameters.

(5) Use other macros

In addition to the plug-in solution THook versus SYMCALL, And also prepared other improvement development

Efficiency macro:

1 ,**POINTER** versus **POINTER_ADD_OFFSET** Macro

```
POINTER(Pointer type, pointer amount)
```

```
POINTER_ADD_OFFSET(Pointer type, pointer amount, offset)
```

POINTER Used to convert any type of value to bit pointer type;

POINTER_ADD_OFFSET Add the pointer to the given memory address offset (byte

Is the unit), and then convert the type.

such as `POINTER_ADD_OFFSET(long*, 0 x1000, 0 x2)`, It will give a

Type is `long*Pointer`, the address saved in the pointer is `0 x1002`. But if you use

Pointer addition, such as `POINTER(long*, 0 x1000) + 2`, Its result is `0 x1008`,

because `sizeof(long)`for 4 Bytes, "+ 2 "Will shift the pointer by two long Type length.

2 ,**CLASS_OBJECT** in **CLASS_VTABLE_OBJECT** Macro

```
CLASS_OBJECT(Object type, This Pointer, offset)
```

```
CLASS_VTABLE_OBJECT(Object type, This Pointer, offset)
```

CLASS_OBJECT For access C++Objects inside a class or structure;

CLASS_VTABLE_OBJECT For access C++Objects in the virtual table in the class;

If a class is in memory `0 x1000` In the position, there is a member variable inside it for

int Type, offset from the head of the class 0 x40, Then you can access this variable like this:

```
CLASS_OBJECT(int, 0 x1000, 0 x40)
```

versus CLASS_OBJECT similar, CLASS_VTABLE_OBJECT The access is the virtual

The object in the pseudo-table is usually used to obtain the address of the virtual function of the class.

3 .SYM_OBJECT Macro

SYM_OBJECT Macros are used to use symbols that are not functions:

```
SYM_OBJECT(Object type, symbol RVA value)
```

The object type is the data type behind the symbol.

If in the main module of the process 0 x4600 There is a int Type of global variable, then

Write code that can access it:

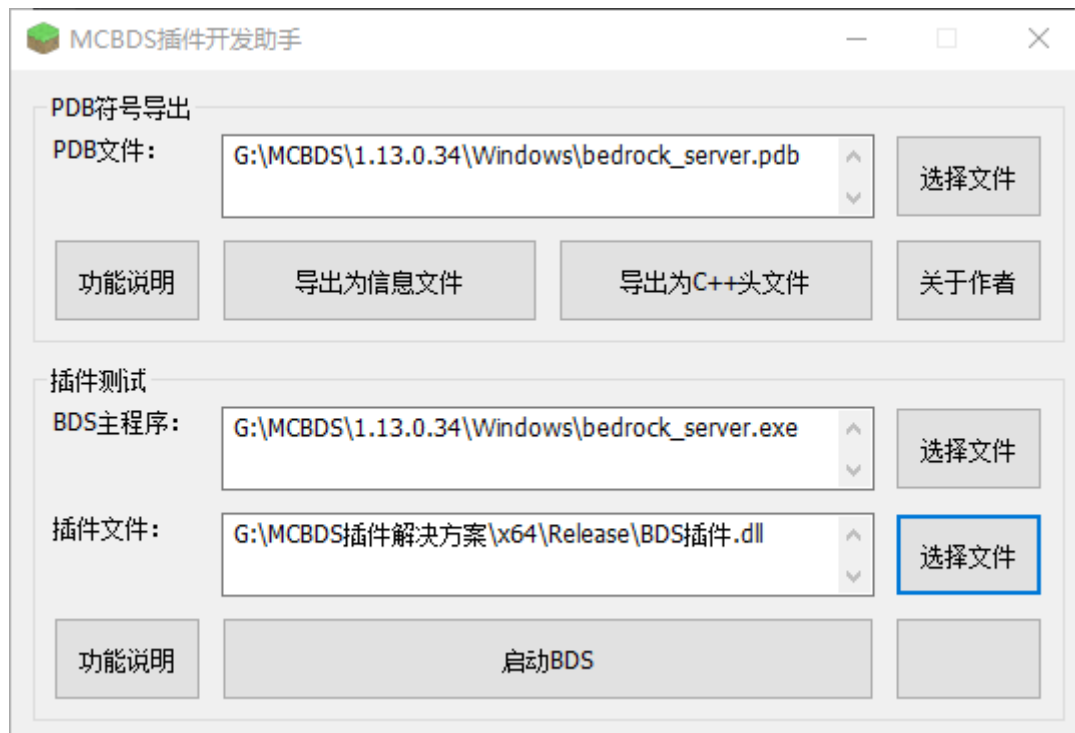
```
SYM_OBJECT(int, 0 x4600)
```

(6) Start BDS And test the plugin

After the plug-in is written, use VS2019 Generate a solution, it will generate "BDS Plugins.dll"

file. At this time, you need to test the plug-in.

will BDS Both the main program and the plug-in file are selected into their respective edit boxes:



Then click "Start BDS" Button to start BDS And load the corresponding plug-in.

After the test is complete, be sure to enter "stop" Instruction to BDS Come close BDS, other

Method close BDS Make the launcher mistakenly think BDS Is closed because of a crash, it will be automatically reset

Newly pulled up BDS process.

Seven, preliminary understanding BDS Internal composition

(1) Basic understanding

BDS Is to use C++ Language written and compiled, and other C++ The procedure is the same,

It uses standard input and output, using C++ Standard library:

```

271 v269 = __readfsqword(0x28u);
272 _this = (AppPlatform **)a1;
273 DedicatedServer::initializeLogging(a1);
274 FeatureToggles::initialize(a1);
275 v214 = 1LL;
276 LODWORD(v29) = 210;
277 BedrockLog::log(0, 1, 0, 0xCu, 2LL, (__int64)"start", v29, (__int64)"Starting Server");
278 v213 = 1LL;
279 Common::getServerVersionString[abi:cxx11](&v268);
280 v2 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::c_str(&v268);
281 LODWORD(v30) = 211;
282 BedrockLog::log(0, 1, 0, 0xCu, 2LL, (__int64)"start", v30, (__int64)"Version %s", v2);
283 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v268);
284 v212 = 1LL;
285 v3 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::c_str(a2);
286 LODWORD(v31) = 212;
287 BedrockLog::log(0, 1, 0, 0xCu, 2LL, (__int64)"start", v31, (__int64)"Session ID %s", v3);
288 std::make_unique<ConsoleInputReader>(&v211);
289 std::unique_ptr<ConsoleInputReader,std::default_delete<ConsoleInputReader>>::operator=((char *)a1 +

```

It's easy to see that there are uses such as `std::string`, `std::unique_ptr`. Such

Standard library content. BDS A large part of the internal calls are related to C++ Related to the standard library.

To understand BDS What is done in such code, not just to be familiar with C++ Language, still have to do

To be familiar C++ Standard library of languages. Online here C++ The standard library learning website is available.

address: <https://zh.cppreference.com/w/cpp>

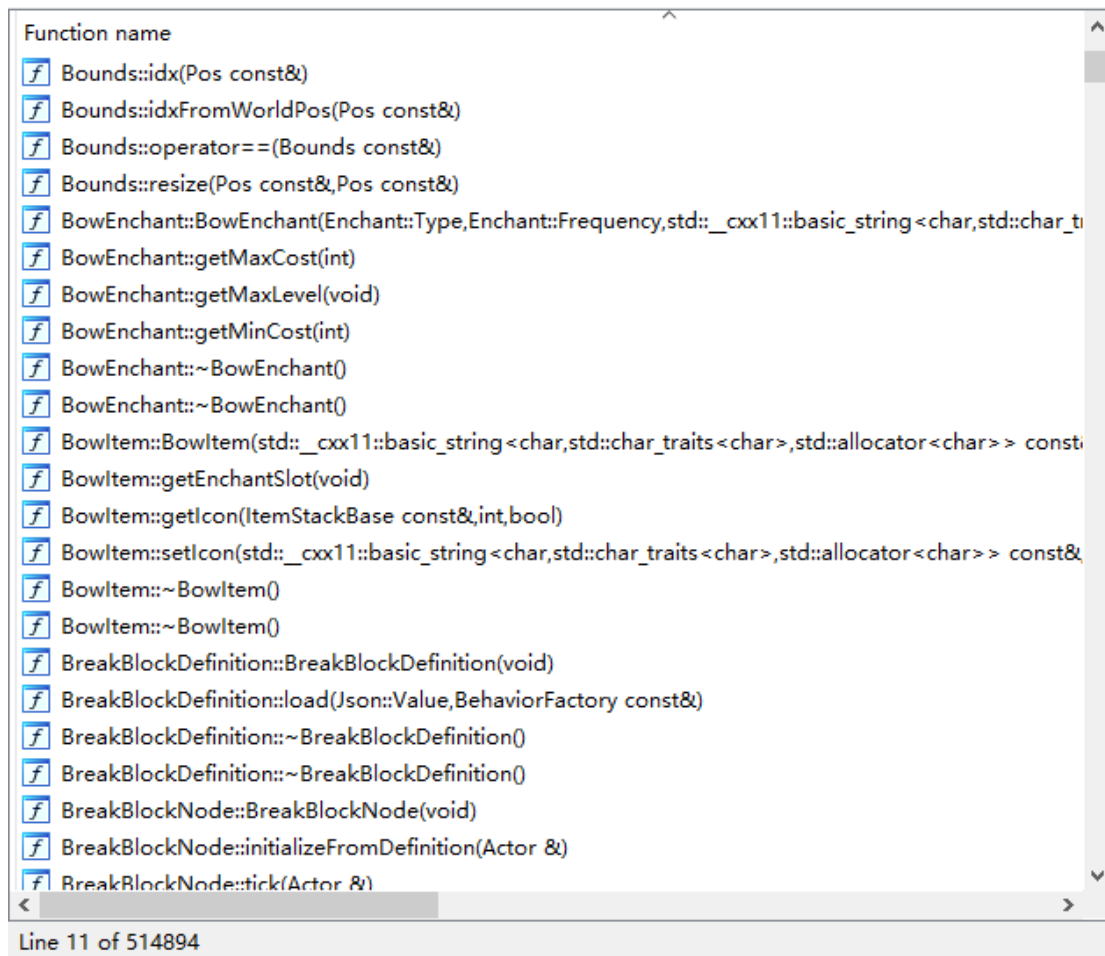
When you encounter something unclear, you can check on it.

(2) Overall composition

Any program is an algorithm plus data structure, in IDA Pro Inside, we regard the algorithm as

It is a skeleton, and the data structure is attached to the skeleton. 5 1 Thousands of functions together constitute

Up BDS Programs, many of which are class methods before compilation:



Can be considered numerous C++Classes make up all of this, and these classes are used throughout

The various functions of the server, such as:

Box related:Block,DirtBlock,BedBlock,IceBlock Wait;

Biological related:Zombie,Villager,Silverfish,Wolf Wait;

World related:Level,LevelChunk,BlockSource,Dimension Wait.

In addition to dividing by purpose, there is also an inheritance relationship between these classes, such as the funnel party

Piece(HopperBlock), leaf block (LeafBlock), sand cube (SandBlock)

Wait, all cube classes have a common father or grandfather:BlockLegacy class. Decide

The most basic functions and appearances of these block classes (subclasses) are described.

These classes are like cells, forming organizations according to various intricate relationships.

And eventually form a called BDS The "biology". In addition to the classes mentioned above, there are

Various in-game play methods (redstone, projectile, manufacturing synthesis, etc.), commands, event queues, land

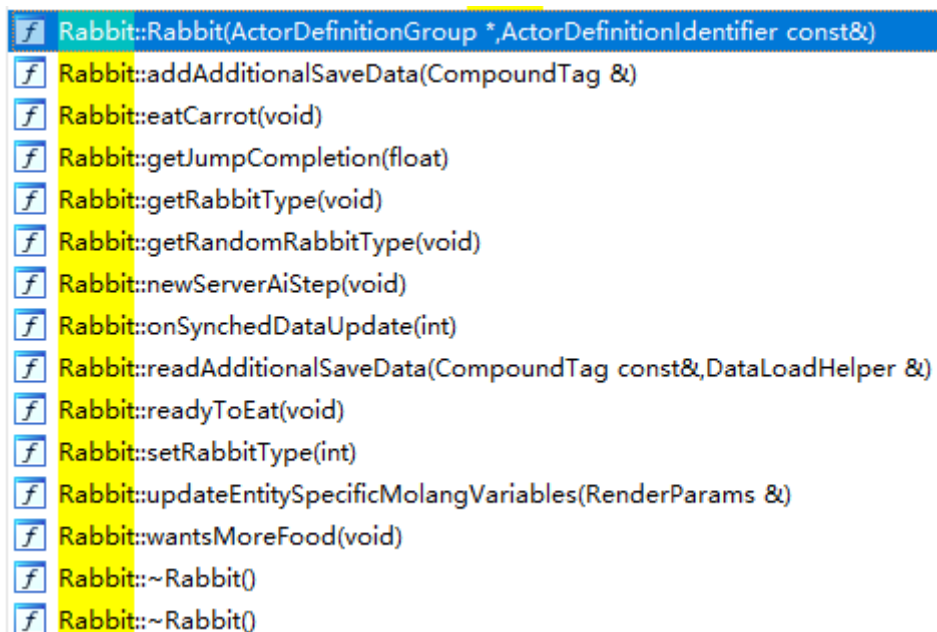
Picture archives and so on, so when you want to know something, first try to search for it

English name, and then find one of the key words, as mentioned before, in the function column

Press in the table Ctrl-F Open the search box, fill in the string and filter out related methods. according to

The name of the class method can make a basic inference about the function of the function. For example, we search

Rabbit(rabbit):

A screenshot of a C++ class method list for the 'Rabbit' class. The list is displayed in a window with a blue header bar. The methods are listed in a column, each preceded by a small icon of a function symbol 'f' inside a square. The methods are: Rabbit::Rabbit(ActorDefinitionGroup *,ActorDefinitionIdentifier const&), Rabbit::addAdditionalSaveData(CompoundTag &), Rabbit::eatCarrot(void), Rabbit::getJumpCompletion(float), Rabbit::getRabbitType(void), Rabbit::getRandomRabbitType(void), Rabbit::newServerAiStep(void), Rabbit::onSynchedDataUpdate(int), Rabbit::readAdditionalSaveData(CompoundTag const&,DataLoadHelper &), Rabbit::readyToEat(void), Rabbit::setRabbitType(int), Rabbit::updateEntitySpecificMolangVariables(RenderParams &), Rabbit::wantsMoreFood(void), Rabbit::~Rabbit(), and Rabbit::~Rabbit(). The first method is highlighted in blue, and the last two are highlighted in yellow.

```
f Rabbit::Rabbit(ActorDefinitionGroup *,ActorDefinitionIdentifier const&)
f Rabbit::addAdditionalSaveData(CompoundTag &)
f Rabbit::eatCarrot(void)
f Rabbit::getJumpCompletion(float)
f Rabbit::getRabbitType(void)
f Rabbit::getRandomRabbitType(void)
f Rabbit::newServerAiStep(void)
f Rabbit::onSynchedDataUpdate(int)
f Rabbit::readAdditionalSaveData(CompoundTag const&,DataLoadHelper &)
f Rabbit::readyToEat(void)
f Rabbit::setRabbitType(int)
f Rabbit::updateEntitySpecificMolangVariables(RenderParams &)
f Rabbit::wantsMoreFood(void)
f Rabbit::~Rabbit()
f Rabbit::~Rabbit()
```

You can see that there are several interesting class methods:

eatCarrot: It is guessed that it is related to the behavior of rabbit eating carrots;

readyToEat: The guess is to judge whether the rabbit is ready to eat food;

setRabbitType: Guess is to set the type (color) of the rabbit.

We will not study and analyze whether the guess here is correct. In short, this is a ten

An efficient way to find the location of the function.

(3) Start execution

Each C++The program has a program calledmain The function as the start of the program's execution

mouth, BDS is no exception, it is also from main Started. Need to pay attention to within the key points

Include:

- main Executed in the main thread, it will initialize a call DedicatedServer Of the class, of course

After calling its class method DedicatedServer::start Start the server;

- DedicatedServer::start Will do a lot of work before the server runs, such as loading and configuring

Set files, resource packs, start worker threads, etc., and eventually enter an event loop,

Until the server is ready to shut down, it will exit the loop and start from start Function returns;

- start Will generate a lot of internal BDS Working thread, there are I/O Read and write or network access

Ask the thread. There is a special thread called "MC_SERVER", the main thread will only

Start an instance of this thread and use it to complete most of the background work of the game. It is

From ServerInstance::_update Start execution, if it is found in the crash log

If this function is used, it means that the crash is caused by the thread, not the main thread;

- BDS In the case of too many players, there will be a freeze, if the server's network

The width is not full, then CPU There must be a core fully loaded, this fully loaded core

What is being executed is started by the main thread "MC_SERVER" Working thread, pay attention to

It's not that there are too many creatures (including monsters) in the server, because this part of the calculation is very intensive CPU

Resources, and are processed within the thread.

(4) The difference between the versions

Since it is compiled from the same set of codes, from a functional point of view, the two versions BDS

The behavior is the same, but the difference in optimization level causes the two versions to look slightly at the internal behavior

Are different, such as SleepGoal::tick, Look first Ubuntu Version of:

```

1 __int64 __fastcall SleepGoal::tick(SleepGoal *this)
2 {
3     __int64 result; // rax
4
5     MoveToPOIGoal::tick(this);
6     if ( !(__int64 (__fastcall **)(_QWORD **))this + 18) + 696LL)(__int64 *)this + 18) & 1 )
7         return SleepGoal::lockPosToBedPos(this);
8     result = (__int64 (__fastcall **)(SleepGoal *))(__int64 *)this + 88LL)(this);
9     if ( result & 1 )
10         result = SleepGoal::setSleepVariables(this);
11     return result;
12 }

```

Look again Windows Version of:

```

1 void __fastcall SleepGoal::tick(SleepGoal *this)
2 {
3     SleepGoal *v1; // rbx
4     __int64 v2; // rdx
5     __int64 v3; // rax
6     _DWORD *v4; // rax
7     _DWORD *v5; // rax
8     __int64 v6; // rax
9     __int64 v7; // rcx
10
11     v1 = this;
12     MoveToPOIGoal::tick(this);
13     if ( !((__int8 (__fastcall **)(_QWORD **))v1 + 19) + 688i64)(__int8 *)v1 + 19) )
14     {
15         if ( !((__int8 (__fastcall **)(SleepGoal *))(__int64 *)v1 + 80i64))(v1 )
16             return;
17         LOBYTE(v2) = 1;
18         (*(void (__fastcall **)(_QWORD, __int64))(__int64 *)v1 + 19) + 2024i64)(__int64 *)v1 + 19, v2);
19         v3 = (__int64 *)v1 + 19;
20         *(_DWORD *)v3 + 2272 = 0;
21         *(_BYTE *)v3 + 2276 = 0;
22         (*(void (__fastcall **)(_QWORD **))(__int64 *)v1 + 19) + 1712i64)(__int64 *)v1 + 19);
23         v4 = (_DWORD *)v1 + 19;
24         v4[283] = Vec3::ZERO;
25         v4[284] = *(&Vec3::ZERO + 1);
26         v4[285] = dword_1413ECB48;
27         v5 = (_DWORD *)v1 + 19;
28         v5[86] = Vec3::ZERO;
29         v5[87] = *(&Vec3::ZERO + 1);
30         v5[88] = dword_1413ECB48;
31         v6 = Actor::tryGetComponent<NavigationComponent>((__int64 *)v1 + 19);
32         if ( v6 )
33         {
34             v7 = (__int64 *)v6 + 56;
35             if ( v7 )
36                 (*(void (__fastcall **)(__int64, __int64, _QWORD))(__int64 *)v7 + 72i64))(v7, v6, (__int64 *)v1 + 19);
37         }
38     }
39     SleepGoal::lockPosToBedPos(v1);
40 }

```

The difference is already very obvious, Ubuntu The analysis results of the version are very readable and more suitable

Good for analysis, but Windows Version has smaller requirements on computer performance, and in the same configuration

under Windows The version allows more players to play at the same time without stuttering.

Windows Version BDS When compiling, the compiler will choose to optimize some functions.

Cause Ubuntu The functions that exist in the version are in Windows If you can't find it in the version, generally come

Say the following type functions are in Windows Version BDS The risk of being optimized and disappearing here is very high:

- Empty function: It does nothing inside.

- Too short function: Because it is too short, the compiler will integrate it into the function that calls it as a whole,

Combine them into a complete function and compile machine code. This operation is called "inlining".

Some of the optimized functions are class construction and destruction methods.

- Duplicate functions, two or more functions with exactly the same behavior, the compiler will

The numbers "merge" into one.

For the convenience of explanation, I will show later IDA Pro The screenshot is not clearly stated

Under the circumstances, they all default from Ubuntu Version BDS of.

(5) What to pay attention to

1 . Functions beginning with "__" or "___"

This type of function is usually a function generated by the compiler or by IDA Pro Reverse analysis generation

of. If you can't open it by double-clicking it, it's IDA Pro Self-generated, it

There is no actual function body. Usually this kind of function has nothing to do with the reverse analysis work we have to do.

Off, only has a very simple function, its existence can almost be ignored.

function "__readfsqword" belong IDA Pro Self-generated, you can only

Ubuntu Version BDS Analyzing F5 See it in the result of the function, its only role is

Judgment detection x86-64 System stack pointer register during program execution RSP Is the location pointed to?

There is a misalignment. No help to our reverse analysis work.

2 . With "sub_" Function

The names of these functions are chosen because the compiler removes their names during the compilation process.

Or they have no names at all. IDA Pro In the analysis gave it an assignment this

The only name. The most common is to initialize `astd::function` Object.

8 . Develop the first plug-in

Now that everything is ready, to the core part of this tutorial: build me

Our plugin. In order to give developers a deep understanding of the development process, the first plug-in

We designed a simple monitoring function for player behavior.

(1) Design goal: record player's aggressive behavior

As a yuba, sometimes you need to know whether the players in the server will be malicious

For, for example, attacking other players or the player's biological property. This requires the server to remember

Record some behaviors for verification. due toBDS Use standard output, for easy viewing, here

We also output the recorded information to standard output. Therefore, you need to do the following

One thing:

- Intercept the execution of the player's attack, and then execute the program we set;
- When intercepted, collect information about the player, such as the player's name;
- Organize the collected information into a log record and output it to standard output.

(2) Basic analysis: find and analyze BDS Internal key function

turn on IDA Pro And load the previously saved bedrock_server.i64(Ubuntu Version),

Use the previously mentioned method to speculate on the function we need to find:

"The function we are looking for is the player's aggressive behavior. On the English name, 'player' is

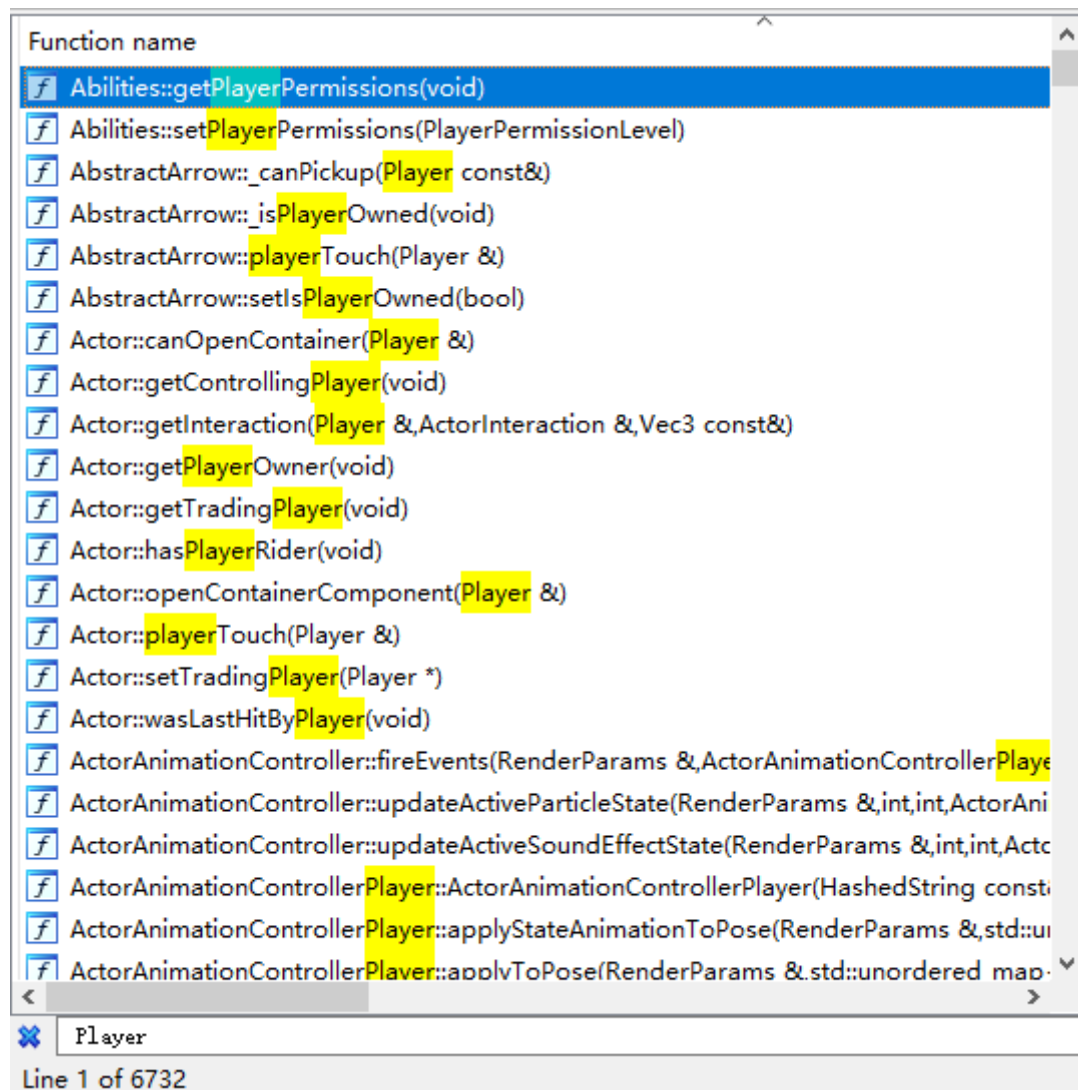
'Player', and 'attack' is 'Attack'"

The name of the key function may be related to its name, here we choose to search

"Player"String. Here in order to centrally view the operations of the same class and to facilitate the

To search for the first letter of the name, we click on the "Function name"

To sort this column:



Here, we see a lot of "Player"-Related functions, at this time the function list

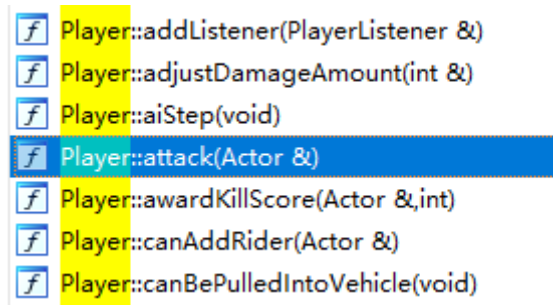
Has been screened and left 6 7 3 2 There are so many functions, but there are still too many. Think about it carefully BDS

The internal composition of the server: a large number of classes constitute all of this, the use of these classes throughout the server opener

Various functions. Then we have reason to guess: there is a "Player" type,

It is in charge of all operations of the player. In alphabetical order, we quickly discovered this called

do "Player", and found the class method (function) we want:



function:Player::attack, The name perfectly matches the previous guess, it looks like this

It's a key function of the player's aggressive behavior. It's totally unexpected to call this name but the function is not the same.

The reason. For the sake of safety, we have to test this function to verify it

Is it triggered when the player attacks?

Please note that since we are developing Windows Version BDS Plug-in, writing plug-in

Before, please open Windows Version BDS Reverse analysis result file confirms this letter

Whether the number has been optimized and disappeared, it has been confirmed here that this function exists, so that we

You can proceed to the next step.

If the objective function is Windows The version is optimized and disappeared, then it cannot be used

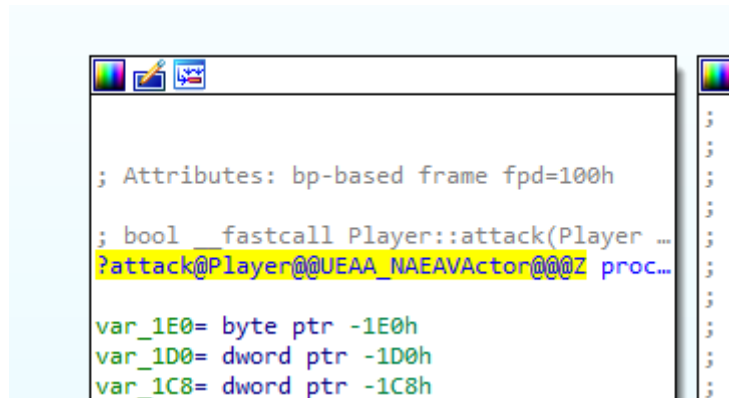
This function does something. Maybe all the analysis is in vain, maybe the result of the analysis

Can help find the next useful function, or another way of thinking to achieve the desired function.

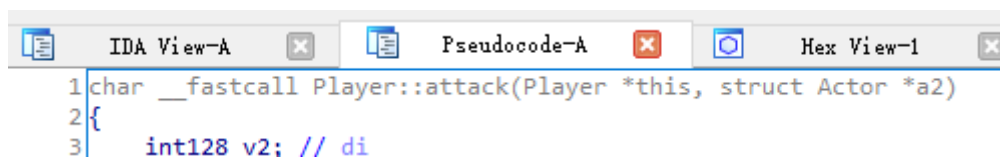
(3) Test symbol: simple functional verification

turn on Windows Version BDS Reverse analysis of the result file, find Player::Attack letter

Count, then copy its symbol:



Press again at this time F5 Key to open the pseudo-code interface and confirm its function prototype:



We have collected its symbols and prototypes, and now, we begin to write work.

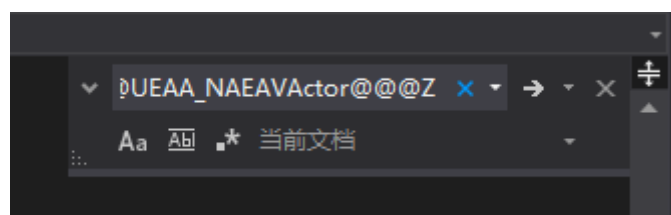
First, open the plug-in solution in the development kit, as the first use, here I

We use the development assistant to export C++Header file to "Symbol Location.hpp"File, then type

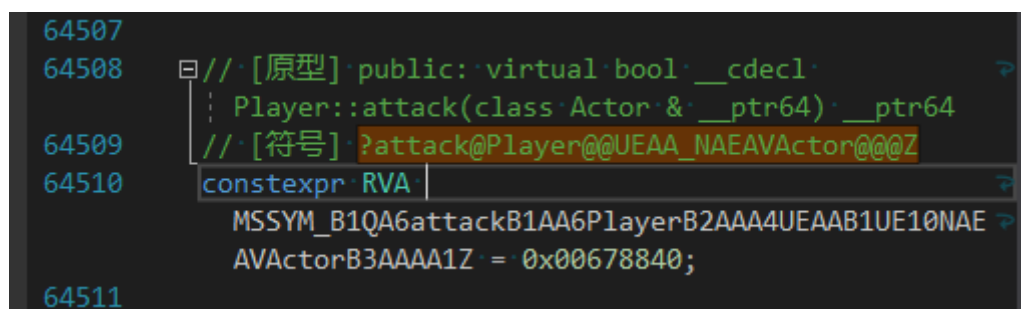
Open this file. ExportedC++The header file is very large, there are 1 0 0 MB Left and right size, open

It may take a long time. After opening, due to the huge number of internal variables, we use the search function

Can quickly find the symbol of our choice. Press downCtrl-F Combination keys, and paste into the symbol:



Then quickly found the target:



Then we started writing the plug-in, using THook The macro intercepts the call of this function as

Keep and BDS The output code of its own is the same, here is used UTF-8 String:

```
THook(bool,  
→ MSSYM_B1QA6attackB1AA6PlayerB2AAA4UEAAB1UE10NAEAVA  
→ void* class_this, void* pactor)  
{  
→ std::cout << u8"Player::attack执行" << std::endl;  
→ return original(class_this, pactor);  
}  
  
void init() {  
→ std::cout << u8"插件已启动! " << std::endl;  
}  
void exit() {}
```

Finally, compile the plug-in and start it with the development assistant BDS:

```
插件已启动!  
NO LOG FILE! - setting up server logging...  
[2019-12-07 19:56:31 INFO] Starting Server  
[2019-12-07 19:56:31 INFO] Version 1.13.0.34  
[2019-12-07 19:56:31 INFO] Session ID 20191e5a-b9d7-4b8b-  
[2019-12-07 19:56:31 INFO] Level Name: World  
[2019-12-07 19:56:31 INFO] Game mode: 0 Survival  
[2019-12-07 19:56:31 INFO] Difficulty: 1 EASY  
[2019-12-07 19:56:33 INFO] IPv4 supported, port: 19132  
[2019-12-07 19:56:33 INFO] IPv6 supported, port: 19133  
[2019-12-07 19:56:33 INFO] IPv4 supported, port: 53157  
[2019-12-07 19:56:33 INFO] IPv6 supported, port: 53158  
[2019-12-07 19:56:33 INFO] Server started.
```

Select the address port of the server, here is the default port of the machine:



Open the game client and enter the game server to attack a creature:



Then view BDS Output window:

```
[2019-12-07 19:59:57 INFO] Server started.  
[2019-12-07 20:00:11 INFO] Player connected:  
Player::attack执行
```

As we imagined, the behavior of attacking the sheep in the game was detected. We are sure

Up Player::attack You can get the player's aggressive behavior.

Strictly speaking, this is the first plug-in we have developed, although it is used for functionality verified. The subsequent development work will continue on this plug-in solution.

(4) In-depth analysis: looking for ways to obtain other necessary information

As a monitoring function, it is far from enough to only remind players of the operation in the server. Yes, we should at least know the name of the player, the location and the name of the attacker, Only such output content has the significance of recording. Then the total information we need to obtain is:

● Player: Name (Name), dimensions (Dimension), coordinate(Position)

Now we follow the above method to find the function needed here.

1 . In Player Find related functions in the class

contain"Name"of:

```
f Player::getAlwaysShowNameTag(void)
f Player::getFormattedNameTag(void)
f Player::setName(std::__cxx11::basic_string<
```

contain"Dimension"of:

```
f Player::_fireDimensionChanged(void)
f Player::_fireWillChangeDimension(void)
f Player::changeDimensionWithCredits(AutomaticID<Dimension,int>)
f Player::fireDimensionChangedEvent(AutomaticID<Dimension,int>)
f Player::getRespawnDimensionId(void)
f Player::isPositionRelevant(AutomaticID<Dimension,int>,BlockPos const&)
f Player::setRespawnDimension(AutomaticID<Dimension,int>)
f Player::setRespawnDimensionId(AutomaticID<Dimension,int>)
```

contain"Position"or"Pos"of:

```

[f] Player::_chooseSpawnPositionWithinArea(void)
[f] Player::_ensureSafeSpawnPosition(Vec3 &)
[f] Player::_findFallbackSpawnPosition(Vec3 &,std::vector<Vec3> &)
[f] Player::_fixup4JBedSpawnPosition(Vec3 &)
[f] Player::_validateSpawnPositionAvailability(Vec3 const&,int &)
[f] Player::checkAndfixSpawnPosition(Vec3 &,std::vector<Vec3> &)
[f] Player::clearRespawnPosition(void)
[f] Player::getSpawnPosition(void)
[f] Player::getStandingPositionOnBlock(BlockPos const&,int &)
[f] Player::handleMovePlayerPacket(Player::PositionMovementPacket &)
[f] Player::hasRespawnPosition(void)
[f] Player::isPositionRelevant(AutomaticID<Dimension>,int &)
[f] Player::recheckSpawnPosition(void)
[f] Player::setBedRespawnPosition(BlockPos const&)
[f] Player::setRespawnPosition(BlockPos const&,bool)
[f] PathNavigation::getTempMobPos(Mob const&)
[f] Player::getAttachPos(ActorLocation,float)
[f] Player::getCapePos(float)
[f] Player::resetPos(bool)
[f] Player::updateTeleportDestPos(void)
[f] PlayerCommandOrigin::getCursorHitBlockPos(void)

```

This result is generally not satisfactory because:

- Name only setName, There is no corresponding getName, And some are formatted

The name tag (understood literally).

- Dimension only RespawnDimension(Literally: rebirth dimension) related letters

There is no information about the dimensions of the player's current location.

- Position or Pos There are only functions related to birth, rebirth, or player objects. only

One that seems to be a bit related is that getStandingPositionOnBlockAnd after

Comes with aBlockPos parameter. HerePos YesPositionOmission,BlockPos

Indicates the position of the square. In addition, the function that sets the player's current position is also not seen.

2 . Analysis Player Suspicious function in the class

Based on the search results above, here we look at the following functions:

- getAlwaysShowNameTag
- getFormattedNameTag
- getStandingPositionOnBlock

first of all getAlwaysShowNameTag:

```
1 __int64 __fastcall Player::getAlwaysShowNameTag(Player *this)
2 {
3     return 1LL;
4 }
```

After that getFormattedNameTag:

```
1 __int64 __fastcall Player::getFormattedNameTag[abi:cxx11]
2 {
3     Actor::getFormattedNameTag[abi:cxx11](a1, a2);
4     return a1;
5 }
```

Finally getStandingPositionOnBlock:

```
15 v13 = __readfsqword(0x28u);
16 BlockPos::above((BlockPos *)v9, (int *)a2);
17 BlockPos::above((BlockPos *)v10, v9);
18 v2 = *((float *)this + 133);
19 Vec3::Vec3((Vec3 *)v8, 0.0, v2, 0.0);
20 v7 = LODWORD(v2);
21 _mm_storel_ps((double *)&v6, (__m128)operator+(v10, v8));
22 v12 = v7;
23 v11 = v6;
24 v5 = v7;
25 v4 = v6;
26 return _mm_loadl_epi64((const __m128i *)&v4);
27 }
```

Can see the function getAlwaysShowNameTag Only return one bool value true.

function getFormattedNameTag It's packed Actor The class method of the class. Final letter

number getStandingPositionOnBlock Just take the whole process PlayerA value in the class is used

Calculation, and the coordinates must have at least X,Y,Z Three float Value, obviously not fully obtained

Take the coordinates. only left with Actor::getFormattedNameTag No analysis, due to this letter

Number does not belong Player Class, so leave it to later Actor Analyze it when class. Present

Already determined Player There is nothing we want in the class.

3 . Expand the search scope to Player Class-related class

The above results did not allow us to successfully find the function we want. This also gave me

We have more room for thinking: a normal player is playing on the server, without any other

The subject will set his dimensions and coordinates, which is obviously unreasonable. After all, hell

Sending the door still has to be sent across dimensions. Then since Player Without these in the class, will it be

What about its parent class or member variable? Or it's not stored at all Player class

Inside? Or is this kind of operation not written as a class method at all?

Here we first place the guess that is not written as a class method, from BDS Code style of the program

Look up, develop BDS Of programmers are obviously writing code in compliance with a certain specification, the above figure

Can already be seen in Player There are many class methods in a class, although the function of these class methods

Very single, but the developer still encapsulates them with class methods, even if this function only

Return a value. The possibility that it has not been written as a class method is too small. When all other possible

This speculation is only considered when sex has been rejected.

Then the next step is to find Player The parent class or member variables of the class to search for some

clue. Please briefly review the above mentioned C++ The memory layout of the class. Just find

Player Class constructor Player::Player, You can know which classes are in it,

Whether it is a parent class or a member variable. We choose to look first Player The inheritance relationship of the class.

turn on Player Class constructor, found at the beginning Mob::Mob Function, where

used IDA Pro Automatically named a1 The variables are treated as Mob Category this Pointer, and follow

IDA Pro Naming rules, a1 Yes Player::Player The first parameter, which is Player

Category this Pointer, and the operation of covering the virtual table in the following figure is in the next row immediately, this

So sure Mob Class is Player The parent class of the class:

```

67 char v76; // [rsp+618h] [rbp-88h]
68 char v77; // [rsp+638h] [rbp-68h]
69 char v78; // [rsp+658h] [rbp-48h]
70 unsigned __int64 v79; // [rsp+678h] [rbp-28h]
71
72 v79 = __readfsqword(0x28u);
73 v45 = a3;
74 v44 = a4;
75 src = a5;
76 v42 = a6;
77 Mob::Mob((Mob *)a1, a2);
78 *(_QWORD *)a1 = &vtable for 'Player' + 2;
79 *(_DWORD *)a1 + 2520 = 0;
80 *(_BYTE *)a1 + 2524 = 0;

```

Then open Mob::Mob and found Actor::Actor Function, this time, IDA Pro

Name this variable directly this, determine Mob The parent class is Actor:

```

1 Mob * __fastcall Mob::Mob(Mob *this, Level *a2)
2 {
3     Mob *result; // rax
4
5     Actor::Actor(this, a2);
6     *(_QWORD *)this = &vtable for 'Mob' + 2;
7     *(_DWORD *)this + 530 = 0;
8     *(_DWORD *)this + 531 = 0;

```

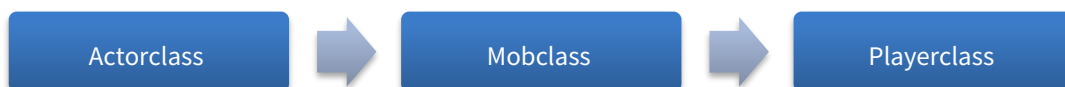
Look again Actor::Actor function:

```

18 unsigned __int64 v19; // [rsp+1D8h] [rbp-8h]
19
20 v19 = __readfsqword(0x28u);
21 v11 = a3;
22 *(_QWORD *)this = &vtable for 'Actor' + 2;
23 v10 = this;

```

Okay, no more. Then Player The derivation relationship of the class is:



Now we need to find out if there is anything we want in these two classes. First

Mob class:

```

[f] MinecraftEventing::fireEventPortalUsed(Player *,A
[f] MineshaftFeature::createStructureStart(Dimensio
[f] Mob::onPlayerDimensionChanged(Player *,Autor
[f] NetEventCallback::handle(NetworkIdentifier con:
[f] NetherDimension::NetherDimension(Level &,Sch

```

```

[f] MinecraftEventing::fireEventItemNamed(Player *,Iter
[f] MinecraftEventing::firePersonaMissingCapeTexture(
[f] MobArmorEquipmentPacket::getName(void)
[f] MobEffect::getByName(std::__cxx11::basic_string<ch

```

The result: nothing. Not even as good as Player class. There is only Actor class

While Player::attack There are also Actor Class, which means that the player is in the game

Those "creatures" that attack are actually the same as the player, and the player is also the same Actor,

The basic part is BDS The interior is the same. this is Actor Search situation of the class:

include "name" of:

```

[f] Actor::canShowNameTag(void)
[f] Actor::filterFormattedNameTag(UIProfa
[f] Actor::getAlwaysShowNameTag(void)
[f] Actor::getFormattedNameTag(void)
[f] Actor::getNameTag(void)
[f] Actor::getNameTagAsHash(void)
[f] Actor::setNameTag(std::__cxx11::basic_s
[f] Actor::setNameTagVisible(bool)

```

include "Dimension" of:

```

[f] Actor::_setDimension(Dimension &)
[f] Actor::_setNetherPortalData(AutomaticID<Dimension,i
[f] Actor::canChangeDimensions(void)
[f] Actor::changeDimension(AutomaticID<Dimension,int>,
[f] Actor::changeDimension(ChangeDimensionPacket con
[f] Actor::getDimension(void)
[f] Actor::getDimensionConst(void)
[f] Actor::getDimensionId(void)

```

contain "Pos" of:

```

[f] Actor::_getBlockOnPos(void)
[f] Actor::_playFlySound(BlockPos const&,Block const&)
[f] Actor::_playStepSound(BlockPos const&,Block const&)
[f] Actor::_randomHeartPos(void)
[f] Actor::_setPos(Vec3 const&)
[f] Actor::_setPosPrev(Vec3 const&)
[f] Actor::calcCenterPos(void)
[f] Actor::checkEntityOnewayCollision(BlockSource &,BlockPos const&)
[f] Actor::distanceSqrToBlockPosCenter(BlockPos const&)
[f] Actor::enableAutoSendPosRot(bool)
[f] Actor::getAttachPos(ActorLocation,float)
[f] Actor::getFiringPos(void)
[f] Actor::getFirstAvailableSeatPos(Actor&,Vec3 &)
[f] Actor::getInterpolatedPosition(float)
[f] Actor::getInterpolatedRidingPosition(float)
[f] Actor::getPos(void)
[f] Actor::getPosExtrapolated(float)
[f] Actor::getPosOld(void)
[f] Actor::handleInsidePortal(BlockPos const&)
[f] Actor::isWithinRestriction(BlockPos const&)
[f] Actor::onBounceStarted(BlockPos const&,Block const&)
[f] Actor::onOnewayCollision(BlockPos)
[f] Actor::positionAllRiders(void)
[f] Actor::positionRider(Actor&,float)
[f] Actor::restrictTo(BlockPos const&,float)
[f] Actor::setBlockTarget(BlockPos const&)
[f] Actor::setPos(Vec3 const&)
[f] Actor::setPosDirectLegacy(Vec3 const&)
[f] Actor::setPosPrev(Vec3 const&)
[f] Actor::setPreviousPosRot(Vec3 const&,Vec2 const&)

```

Now the results are very satisfactory, there are various methods we want to find.

4 . Analysis Actor Suspicious function in the class

because Actor Class is Player Grandfather of the class, we can directly take Player Class means

Needle to call Actor Functions in the class.Actor Inside the class Actor::getNameTag,

Actor::getDimensionId with Actor::getPos, So choose to place the previous

getFormattedNameTag Function unless getNameTag Does not meet the requirements.

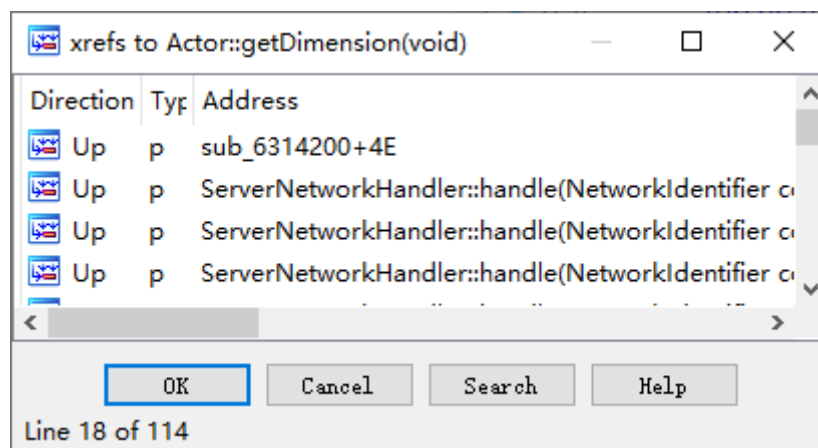
Maybe someone here may ask why I chose getDimensionId Instead of

getDimension? Doesn't this look more reasonable?

Because it was introduced earlier, Dimension is a class, but DimensionId is the dimension of ID value. Even if you getDimension Class, you have to look at it to judge the dimension ID worth it. Take directly ID is to avoid getting Dimension Get it after the class ID, This It's superfluous. analysis getDimension The return type process is like this:

First use IDA Pro In the right-click menu "Jump to xref" Function lookup referenced

getDimension Each position of:



Just find a function, just the first one sub_6314200 Just do it and find

getDimension Location:

```
27 v14 = v4;  
28 v5 = (Dimension *)Actor::getDimension(v4);  
29 v6 = Dimension::getVillageManager(v5);  
30 v13 = (const BlockPos *)std::unique_ptr<Vill
```

Then view Dimension::getVillageManager Class method:

```
1 Dimension *__fastcall Dimension::getVillageManager(Dimension *this)  
2 {  
3     return (Dimension *)((char *)this + 1168);  
4 }
```

In the end, it was very straightforward to judge Actor::getDimension Returned a Dimension pointer, And this pointer is then used as Dimension::getVillageManager Class method this pointer.

Before supplementing the content of the test plug-in, there is one very important thing that needs to be done, and

And don't forget about this: verificationWindows Version BDS Is the corresponding function still

exist. It has been verified here that these three functions still exist, and you can rest assured to proceed to the next step.

As I said before,MSVC The symbol will be more thanGCC The symbol stores more information, especially

Is the return value type information. The more information, the more beneficial to our analysis. right now

Windows Version BDS These three functions happen to be in, so let's take a look at their prototypes:

```
public: virtual class std::basic_string<char, struct
std::char_traits<char>, class std::allocator<char>>
const & Actor::getNameTag(void)const
public: virtual class AutomaticID<class Dimension,
int> Actor::getDimensionId(void)const
const struct Vec3 * __fastcall Actor::getPos(Actor
* __hidden this)
```

First look getNameTag Function, its return value is std::string Type, here is

Show that its parameters are void, But it is a non-static class method, so there must be an implicit

of this Pointer parameters. To be safe, let's verifyIDA Pro Participate in him

Is the number analysis consistent with our prediction (the figure below is fromWindows Version BDS):

```

1 void __fastcall Actor::getNameTag(__int64 a1)
2 {
3     __int64 v1; // rdx
4     __int64 v2; // rax
5     void **result; // rax
6
7     v1 = *(_QWORD *)(a1 + 304);
8     if ( (unsigned __int16)((*(_QWORD *))(a1 + 312)
9         && (v2 = *(_QWORD *))(v1 + 32)) != 0
10        && *(_BYTE *)(v2 + 8) == 4 )
11     {
12         result = (void **)(v2 + 16);
13     }
14     else
15     {
16         result = &Util::EMPTY_STRING;
17     }
18     return result;
19 }

```

Very good, it does have a parameter a1, And was used. Our prediction is correct

of. After that getDimensionId Function (the image below comes from Windows Version BDS):

```

1 DWORD __fastcall Actor::getDimensionId(__int64 a1, _DWORD *a2)
2 {
3     *a2 = *(_DWORD *)(a1 + 188);
4     return a2;
5 }

```

It returned a DWORD Value, which is 32 Value in bit integer form, excluding

It is the memory address (64 Bit size) possibilities, the possible types are int or unsigned

int. But with its prototype AutomaticID The class is not the same. Just when we are ready to find

it is at Windows Version BDS Found that it no longer exists when the constructor in

Is instead looking for Ubuntu Version BDS Internal realization:

```

1 __int64 __fastcall AutomaticID<Dimension,int>::AutomaticID(_DWORD *a1)
2 {
3     __int64 result; // rax
4
5     result = AutomaticID<Dimension,int>::_makeRuntimeID();
6     *a1 = result;
7     return result;
8 }

```

AutomaticID Contains only one DWORD Variable, no vtable pointer.

It should be a pure structure, only this is left after optimization DWORD variable.

Then we use DWORDIsometricIntType to replace this AutomaticID.

Also, pay attention Windows Version BDS in `getDimensionId` There is a parameter `a2`, it is a pointer, used to retrieve the value, then we are using `getDimensionId` when

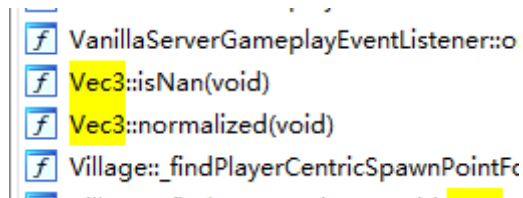
Pass the address of the variable we use to get the value to `a2` That's it.

Finally `getPos` Function (the image below comes from Windows Version BDS):

```
1 const struct Vec3 *__fastcall Actor::getPos(Actor *this)
2 {
3     return (Actor *)((char *)this + 1108);
4 }
```

Same as above, `Vec3::Vec3` Does not exist in Windows Version BDS:

(The picture below is from Windows Version BDS)



turn up Ubuntu Version BDS middle `Vec3::Vec3` function:

```
1 void __fastcall Vec3::Vec3(Vec3 *this, float a2)
2 {
3     *(float *)this = a2;
4     *((float *)this + 1) = a2;
5     *((float *)this + 2) = a2;
6 }
```

So the structure `Vec3` Only has three members float It's just an array of types. it should

What is saved is the coordinates of the player. For such a pure combination of a single data type

Structure, the compiler generally does not change its memory layout without authorization.

Here is a digression, float The effective figures of the type are very limited, and the current

1 .13.0.34 Version of BDS internal use float To save the player's precise coordinates, which leads to

It is strange that the player in the game moves to a place beyond one million grid coordinates on the map.

It is impossible to play normally at all due to the phenomenon of escape. Mojang Do not choose to use in the development of the game

More effective numbers double Types cause such problems.

Now, we have enough understanding of these three functions. Let's sort out me below

The information we obtained:

- function getNameTag, The return value type is std::string*, There is only one parameter, the class

Type is Actor*,for this pointer.

- function getDimensionId, The return value type is int*, There are two parameters, one type

Yes Actor*,for this pointer. The other is a pointer used to save the obtained value, the class

Type is int*.

- function getPos, The return value type is Vec3*, There is only one parameter, the type is Actor*,

for this pointer.

From the inside, they are all taken fromActor The member variables inside the class and return the pointer value,

We cannot intuitively judge whether the information they return is what we need. right now I am

We need to test these three functions.

5 . Test the function that has passed the analysis step

Based on the previous test plug-in, we use the above mentioned SYMCALL Macro

Use these three functions to get the information we want, and because Player::attack There is also

A Actor Parameters, we get its NameTag:

```

std::string* p_player_name = SYMCALL(std::string*,
→ MSSYM_MD5_7044ab83168b0fd345329e6566fd47fd,
→ class_this);
std::string* p_actor_name = SYMCALL(std::string*,
→ MSSYM_MD5_7044ab83168b0fd345329e6566fd47fd,
→ pactor);
int dimensionId;
SYMCALL(int,
→ MSSYM_B1QE14getDimensionIdB1AA5ActorB2AAA4UEBAB1QA2AVB2QD
→ class_this, &dimensionId);
const struct Vec3 {
→ float x;
→ float y;
→ float z;
}* p_position = SYMCALL(const Vec3*,
→ MSSYM_B1QA6getPosB1AA5ActorB2AAE12UEBAAEBWVec3B2AAA2XZ,
→ class_this);

```

Then output it in a very clear way:

```

std::cout
→ <<u8"Player::attack 执行"<<std::endl
→ <<u8"Player(this) getNameTag: "
→ <<p_player_name->c_str()<<std::endl
→ <<u8"Player(Actor) getNameTag: "
→ <<p_actor_name->c_str()<<std::endl
→ <<u8"getDemensionId: "<<dimensionId<<std::endl
→ <<u8"getPos: "<<std::endl
→ <<u8"x: "<<p_position->x<<std::endl
→ <<u8"y: "<<p_position->y<<std::endl
→ <<u8"z: "<<p_position->z<<std::endl;

```

Attack a sheep again, execution result:

```

Player::attack 执行
Player(this) getNameTag: VolutedBucket31
Player(Actor) getNameTag:
getDemensionId: 0
getPos
x: 188.632
y: 63.5625
z: -9.51421

```

Here we have perfectly obtained the player's name, player coordinates, and player dimensions ID.

In subsequent tests, we determined the dimension ID There are three values: 0 , 1 , 2 ,

They represent the main world, hell, and the end. Dimensions in the diagramID for 0 , Which means that the player is in the dominant world

World. The coordinate value is partially correct, which means Vec3 Internally is in accordance with X,Y,Z of

The order to store the coordinate values.

Although we have obtained the player's information, we have not succeeded in obtaining the attacked pair

The name of the elephant. So why? Because the target (sheep) didn't NameTag

? After careful consideration of game experience, I found that this is indeed the case. Sheep is a type of creature.

Rather than the creature's own name, all creatures that are sheep are called "sheep".

Minecraft There is an item in it called a name tag. If the named creature has a name,

Then creatures that are not named naturally have no names, which can explain this speculation.

6 . Continue to look for the function to get the name of the creature type

Unnamed creatures may not have names, but creatures always have types, and they can get classes.

Type names are also feasible. If we get its typeId, We can test the dimension

degree ID The same test for the value of each creature type, but unlike the dimension, the creature type is right or wrong

There are a lot of them. There are dozens of kinds of biological eggs in the creative mode backpack. The test will be one ten.

Divide time-consuming and laborious work. If the version update adds new creatures, the subsequent adaptation

It can also be very annoying. So is there any way to use BDS Own function output biological class

What about the type name? Here we first think about where we can see these types of creatures

Name, and then look at how these functions get the name:

First of all, when each player dies, the game will prompt us what kind of

Something killed, although we don't know if the name of the creature type in this message covers

All types, but at least most of the deaths caused by hostile or neutral creatures will be displayed.

Secondly, when the player tests a certain biological trap (such as an iron machine), they will use

command"/kill @e"To kill the creatures to ensure that the site is cleaned up after the test for the next test.

The command itself will prompt the result of the command execution, and attach the type names of many creatures.

Here for the sake of safety, we choose to look first Kill Command part. search for "kill",

Then I found the most relevant part of it:

```
f EndDragonFight::setDragonKilled(EnderDragon &)  
f KillCommand::KillCommand(void)  
f KillCommand::execute(CommandOrigin const&,CommandOutput &)  
f KillCommand::setup(CommandRegistry &)  
f KillCommand::~KillCommand()  
f KillCommand::~KillCommand()  
f LootItemKilledByActorCondition::LootItemKilledByActorCondition(Actc
```

"KillCommand" This name is in line with our guess here, it is the most suspect

Big class. In these items, except for the structure and destructor of the class, the only thing left is

execute with setup. Kill The command itself cleans up the object when the command is executed,

And feedback the message, then analyze execute Naturally is the best choice, let's take a look at this

A function, since this is a very large function, I don't waste space to show its full picture here

Now, we show which part is most likely to be needed:

```
CommandSelectorResults<Actor>::begin(&v17, &v18);  
CommandSelectorResults<Actor>::end(&v16, &v18);  
while ( SelectorIterator<Actor>::operator!=(&v17, &v16) & 1 )  
{  
    v10 = SelectorIterator<Actor>::operator*(&v17);  
    (*(void (__fastcall **)(__int64))(*(_QWORD *)v10 + 1816LL))(v10);  
    std::allocator<char>::allocator(&v15);  
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>(&v22,  
        "targetname",  
        &v15);  
    CommandOutput::addToResultList(v11, (__int64)&v22, v10);  
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~allocator(&v15);  
    std::allocator<char>::~allocator(&v15);  
    SelectorIterator<Actor>::operator++(&v17);  
}
```

This is a loop that uses iterators to traverse CommandSelectResult<Actor> class

Every discrete structure inside. Its template parameter is Actor, And appeared in the code

"targetname", judging from the literal meaning, this is a saved kill the goal

of Actor* A list of pointers of the type, and the purpose of iterating is to retrieve each item. Of course

And then send these items into CommandOutput::addToResultList, Which is to send it into

The so-called "command output" category in the "result list". So, obviously, this is

The function we are looking for. Here we click to enter addToResultList analysis:

```
v7 = __readfsqword(0x28u);
v5 = (ItemActor *)a3;
if ( *a1 == 4 )
{
    v4 = std::unique_ptr<CommandPropertyBag,std::default_de
    getEntityName[abi:cxx11]((__int64)&v6, v5);
    CommandPropertyBag::addToResultList(v4, a2, &v6);
    std::__cxx11::basic_string<char,std::char_traits<char>,
}
return __readfsqword(0x28u);
```

Here is more clear, a major suspect has appeared: getEntityName. with

Kind of here getEntityName After execution, the parameter &v6 Was given to the following

CommandPropertyBag::addToResultList in. Now, by this name, I

We have already had a reason to stop the analysis work at hand and directly deal with this getEntityName function

Expanded the analysis.

Let's take a look Windows Version BDS Is there this function? By the way, take a look at its parameters

The specific type of the number and return value. This is Ubuntu Version BDS It's a treatment that you can't enjoy,

because MSVC Symbols provide more information:

```
class std::basic_string<char, struct
```

```
std::char_traits<char>, class std::allocator<char>>
```

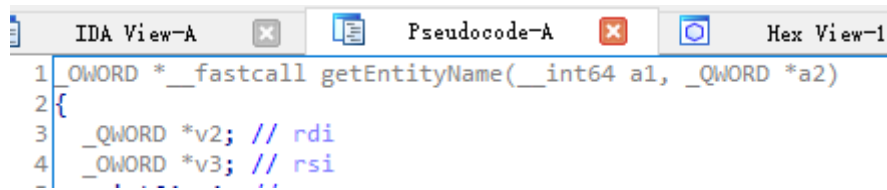
```
getEntityName(class Actor const &)
```

Here we confirm Windows Version BDS This function exists, and from the function prototype

Judging from the above, it should have a parameter Actor* But it tries to return the entire std::string

Structure, let's take a look IDA Pro Analysis result of its function prototype:

(The picture below is from Windows Version BDS)



Confirm that it has two parameters. Now we don't know what the extra parameters are

Meaning, so we need to focus on analyzing the parameters before and after, let's look at Ubuntu

Version BDS What is the origin of the two parameters of the above function? Look at the previous article again.

Screenshot:

```
v7 = __readfsqword(0x28u);
v5 = (ItemActor *)a3;
if ( *a1 == 4 )
{
    v4 = std::unique_ptr<CommandPropertyBag,std::default_de
    getEntityName[abi:cxx11]((__int64)&v6, v5);
    CommandPropertyBag::addToResultList(v4, a2, &v6);
    std::__cxx11::basic_string<char,std::char_traits<char>,
}
return __readfsqword(0x28u);
```

function getEntityName There are two actual parameters, &v6 with v5, In the figure, v6 At this

This function has not been initialized at all before, and has not even been touched, and v5 Yes

CommandOutput::addToResultList The third parameter a3. Continue to track it up

KillCommand::execute:

```
CommandSelectorResults<Actor>::begin(&v17, &v18);
CommandSelectorResults<Actor>::end(&v16, &v18);
while ( SelectorIterator<Actor>::operator!=(&v17, &v16) & 1 )
{
    v10 = SelectorIterator<Actor>::operator*(&v17);
    (*(void (__fastcall **)(__int64))(*(_QWORD *)v10 + 1816LL))(&v10);
    std::allocator<char>::allocator(&v15);
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator
    &v22,
    "targetname",
    &v15);
    CommandOutput::addToResultList(v11, (__int64)&v22, v10);
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator
    std::allocator<char>::~~allocator(&v15);
    SelectorIterator<Actor>::operator++(&v17);
}
```

The third parameter is v10, and v10 Originally taken from the iterator Actor*Types of.

On the other hand, we look at CommandOutput::addToResultList Function prototype,

Taken from Ubuntu Version BDS(inWindows Version BDS The above has disappeared):

```
CommandOutput::addToResultList(std::__cxx11::basic
_string<char, std::char_traits<char>,
std::allocator<char>> const&, Actor const&)
```

It is also confirmed here that the third parameter is Actor*Types of. The rest is the parameter &v6Up,

As before v6 Has not been initialized, we need to click on getEntityName analysis,

Then found something like this:

```
(_QWORD *)a2 + 432LL))(a2);
s<char>,std::allocator<char>>::basic_string(a1, v3);
traits<char>,std::allocator<char>>::empty(a1) & 1 )

its<char>,std::allocator<char>>::basic_string(&v13);
\*(/ 00000000 *v8 + 1240111)(v8) -- 256 \
```

because&v6 Yes getEntityName The first parameter, so IDA Pro Name it for a1. Then we confirmed again &v6 for std::string*Types of.

Now let' s summarize the function getEntityName Related information: The return value is std::string*Type, there are two parameters, the first one is std::string*Type, used to get Name, the second is Actor*Types of. Next, add the test code in the plug-in:

```
std::string actor_typename;
SYMCALL(void*,
→ MSSYM_MD5_01064f7d893d9f9ef50acf1f931d1d79,
→ &actor_typename, pactor);
```

And output it together with the previous information:

```

Player::attack 执行
Player(this) getNameTag: VolutedBucket31
Player(Actor) getNameTag:
getEntityName: Sheep
getDemensionId: 0
getPos
x: 1323.2
y: 70.62
z: 96.2709

```

Here we have successfully obtained its type name as "Sheep", because in the game

The target of internal attack is sheep.

(5) Improved functions: output behavior records in a readable format

In the above process, we have obtained and printed out the required parameters one by one

Come. From the developer's point of view, know the meaning and purpose of each parameter, but as a

With the yuba with plug-ins, they probably don't understand the meaning of these parameters, nor can they understand

The level of these parameters, then we need a concise and concise display of this behavior record

Specific meaning format, such as:

"The player John123 In the main world (2 5 4 , 8 0 , 7 5) Location attacked Sheep"

Now let's write the code, the first is the dimension part, because what we get is the dimension

degree ID, So you need to ID Converted to dimension name:

```

std::string dimension;
switch (dimensionId) {
case 0: dimension = u8"主世界"; break;
case 1: dimension = u8"地狱"; break;
case 2: dimension = u8"末地"; break;
default:
    dimension = u8"未知维度"; break;
};

```

Then we readjust the format of the output information:

```
std::cout
→ <<·u8"玩家"·<<·*p_player_name
→ <<·u8"在"·<<·dimension·<<·u8"中"
→ <<·u8" ("·<<·static_cast<int>(p_position->x)
→ <<·u8", "·<<·static_cast<int>(p_position->y)
→ <<·u8", "·<<·static_cast<int>(p_position->z)
→ <<·u8") "
→ <<·u8"位置攻击"·<< actor_typeofname << u8"。 "
→ <<·std::endl;
```

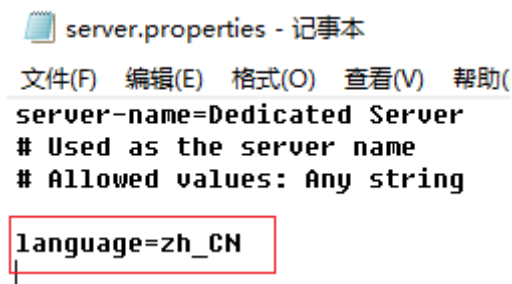
Finally, compile the plug-in, and use the development assistant to test the plug-in:

```
[2019-11-28 10:28:39 INFO] Player connected: VolutedBucket31, x
玩家VolutedBucket31在主世界中 (1335, 69, 67) 位置攻击Pig。
玩家VolutedBucket31在主世界中 (1365, 67, 72) 位置攻击Sheep。
玩家VolutedBucket31在主世界中 (1370, 64, 82) 位置攻击Creeper。
玩家VolutedBucket31在主世界中 (1341, 49, 44) 位置攻击Drowned。
玩家VolutedBucket31在主世界中 (1326, 67, 37) 位置攻击Skeleton。
玩家VolutedBucket31在主世界中 (1258, 60, 92) 位置攻击Squid。
```

In the picture, players attacked pigs, sheep, creepers, water ghosts, skeletons, and squids.

If you don't want to see the type name in English, please follow the method mentioned before

BDS Change the default language to Chinese:



```
server.properties - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
server-name=Dedicated Server
# Used as the server name
# Allowed values: Any string
language=zh_CN
```

Pay attention to UTF-8 Encode and save the file. Then startBDS You can see that the name is

Adjusted to Chinese:

```

[2019-11-28 10:36:08 INFO] Server started.
list
目前有 0/10 个玩家在线:

[2019-11-28 10:38:20 INFO] Player connected: VolutedBucket31, xu
玩家VolutedBucket31在主世界中 (1249, 59, 87) 位置攻击鲑鱼。
玩家VolutedBucket31在主世界中 (1240, 61, 99) 位置攻击鱿鱼。
玩家VolutedBucket31在主世界中 (1270, 50, 99) 位置攻击水鬼。
玩家VolutedBucket31在主世界中 (1318, 68, 97) 位置攻击猪。
玩家VolutedBucket31在主世界中 (1319, 64, 118) 位置攻击蜘蛛。
[2019-11-28 10:40:30 INFO] Player disconnected: VolutedBucket31,

```

In this way, our first plug-in is completed, which is in line with the design goals.

If you put it on the cloud server to open the server, it is recommended to use the development kit nanolauncher

Launcher to cooperate with plug-in startup BDS.

Nine, sum up development experience

Our first plug-in has been developed, now we review and summarize this long opening

Experience in the development process. It will also mention very important content that was not mentioned in the previous article.

Can be used in subsequent development.

(1) The overall process

- 1 . Design the plug-in functions that need to be implemented;
- 2 . Reverse analysis BDS Internal behavior, find out the function that needs to be intercepted;
- 3 . Write the plug-in function code and use hook Mechanism operation objective function;
- 4 . Use the launcher that can load the plug-in to start the server opener in the form of "loading the plug-in";
- 5 . Open the game client, enter the started plug-in server, and add functions to the plug-in

carry out testing.

(2) Reverse analysis process

1 . Search for related functions

BDS Many function symbols are provided internally. For the function prototypes resolved by symbols

Analysis is a very basic and important link. Learn to use hereIDA Pro Built-in search function

It is very necessary. In addition to using search, the names that may be obtained by a reasonable guess are also very

It is important and has a multiplier effect.

2 . Class-related analysis

because BDS Is composed of many classes and their class methods, to figure out the class and the class

The previous relationship is very important:

- Look at the constructor of the class to determine which classes are inherited and which classes belong to

Staff variable.

- use IDA Pro which provided "Jump to xref" or "Jump to xref to operand"

Function to view the class members in which other classes are referenced.

3 . The source of the function prototype

Use the following to confirm the function prototype:

- Two versions BDS Symbol in.

- Two versions BDS middle F5 The result of the decompilation function.

When confirming the function prototype, you must first F5 The result of decompilation is a template, but

The information provided by the post-analysis symbols is used to comprehensively analyze the number and types of the real parameters of the function.

(3) The process of writing code

First of all, you must be familiar with the use of related macros, including THook, SYMCALL with

POINTER_ADD_OFFSET, With the help of these macros, we can more quickly and accurately

Write the code.

Secondly, due to Windows Version BDS Many functions will be optimized. After determining the need to use

Before using the function, be sure to check whether the function still exists in Windows Version BDS

Internally, if it disappears, developers need to find other ways to implement functions.

Finally, in order to keep up with BDS The output code is the same, and the word should be kept during the process of writing the plug-in

The string encoding is always UTF-8. Development Assistant is starting BDS The console will automatically

The code is changed to UTF-8, No plug-in is needed for processing.

(4) Plug-in generation and use

1 . Encountered an internal compiler error when compiling the plug-in

If before VS2019 The version has been updated, then please try to use "Regenerate Solution", this problem can be solved in most cases.

2 ,BDS Plug-in adaptation work after the version update

After the plug-in is successfully generated, it can only be used in conjunction with the last export to the plug-in "symbol setting

Bit.hpp" Of the file BDS Version used. Because after the version is updated, the new version BDS Internal

symbol RVA Will be reassigned, at this time, loading the old version of the plug-in will incorrectly use the old version BDS

The position of the internal symbols to modify the new version BDS The same position in the BDS

collapse. To adapt to the new version, use the development assistant to set the "symbol definition" in the plug-in solution

Bit.hpp" Reuse the new version BDS attached PDB The file is generated, and then re

Newly generated plug-in. If some symbols appear in the process RVA The lack of causes a compilation error

Error, then re-analyze BDS, There should be a function at the missing symbol to fill up, and finally

Make a suitable new version BDS Mechanism plugin.

10. Pitfalls in reverse analysis

The functions compiled by the compiler will more or less not be consistent with the source code. in

In reverse analysis, various traps are often encountered. The following lists the possible

Several traps encountered:

(One)F5 The number of parameters analyzed for the first time by the function is incorrect

due toIDA ProComes withF5The function does not refer to the symbol provided in the analysis process at all

Information, the calling convention stipulates that the first few parameters are passed through registers,F5 Plug-in

Only based on the analysis of the use of parameters by the function itself, the number of analyzed parameters will be different.

The right situation.

The following function was initially used F5 Only one parameter is analyzed during function analysis:

```
__int64 __fastcall RakNetServerLocator::setHostGUID(__int64 a1)
{
    return std::function<RakNet::RakNetGUID ()(void)>::operator=(a1 + 176);
}
```

And look at the information provided by its symbols:

```
f RakNetServerLocator::checkCanConnectToCustomServerAsync(std::__cxx11::basic_string<char> const& a1, unsigned int a2)
f RakNetServerLocator::setHostGUID(std::function<RakNet::RakNetGUID ()(void)>)
f ReadOnlyBinaryStream::getVectorList<CommandOutputMessage>(std::function<C...
```

It should be a class method with two parameters, one of this class this pointer,

The other is pointing std::function<RakNet::RakNetGUID ()(void)>Pointer to the class. but

This is the same as F5 The results of the analysis are different. At this time, click on the only function called inside it:

```
std::function<RakNet::RakNetGUID ()(void)>::operator=(__int64 a1, __int64 a2)
-28h] [rbp-28h]
v4; // [rsp+48h] [rbp-8h]
```

Found here operator=The function does have two parameters, but the above F5 analysis

Only the first parameter is analyzed in, and the previous function is returned again:

```
__int64 __fastcall RakNetServerLocator::setHostGUID(__int64 a1, __int64 a2)
{
    return std::function<RakNet::RakNetGUID ()(void)>::operator=(a1 + 176, a2);
}
```

Here F5 The function has been based on the function called to this function "operator="Analysis

As a result, the parameters of this function were revised, from the original one parameter to two.

However, if the analyst does not manually click to decompile the internally called functions,F5

The function will not automatically correct the erroneous results it analyzes.

This example gives the analyst a warning: to determine the actual formal parameters of a function

Quantity, must combine the symbol and F5 Comprehensive analysis and judgment of the results of the function analysis.F5 Work

Can mark such parameters that cannot be analyzed correctly as orange, and be very careful if you encounter them!

(2) The number of parameters provided by the symbol is incorrect

The above example illustrates F5 The analysis result of the function is sometimes incorrect, then

There are also cases where the number of parameters provided by the symbol is different from the actual number of parameters:

Under normal circumstances, when there is no match, the actual parameters of the function will be more than the symbol provided

One more. This is because the return value type of the function prototype is the structure type, and according to the adjustment

By convention, the return value is usually passed through the register, and the register is also fixed

The size of (6 4 Bit), the size of the structure is usually larger than the size of the register, resulting in

The memory cannot store huge structures. So the compiler is compiling the function with this kind of return value

When counting, a parameter will be appended to the head or tail of the original parameter list, and then let

All callers of this function construct a local change of the structure type in their own function

And pass the address of this variable to the function through this parameter for the function to save and return

value. At the same time, when the function call is completed and returns, the register used to save the return value

A pointer to this structure variable will be stored. AboveActor::getDimensionId Can do

It is a typical case, so I won't give another example here.

(3) The compiler does not generate functions according to the standard calling convention

In rare cases, the compiler will generate functions that do not conform to the calling convention. It has been found that

There is only one situation, the performance is the same as that described above F5 Function analysis shows that the number of parameters is incorrect

The situation is similar, but the difference is that the compiler does not arrange the parameter passing party according to the calling convention

Formula, will lead to F5 In the analysis, an incorrect function parameter list is given, such as a certain

Use of unsigned functions F5 Prototype analyzed:

```
char __fastcall sub_7144EA0(std::_Function_base * a1,  
__int64 a2, __int64 a3, __int64 a4, __int64 a5, __int64  
a6, char a7)
```

There are seven parameters, namely a1 to a7, The return value type is char.F5 function display

The process of getting it out is as follows:

```
1 char __fastcall sub_7144EA0(std::_Function_base *a1,  
2 {  
3     __int64 (__fastcall *v7)(); // rax  
4     __int64 v8; // rax  
5  
6     std::_Function_base::_Function_base(a1);  
7     LOBYTE(v7) = sub_71496A0(&a7);  
8     if ( (unsigned __int8)v7 & 1 )  
9     {  
10        v8 = sub_7149700(&a7);  
11        sub_71496C0(a1, v8);  
12        v7 = sub_7149740;  
13        *((_QWORD *)a1 + 3) = sub_7149710;  
14        *((_QWORD *)a1 + 2) = sub_7149740;  
15    }  
16    return (char)v7;  
17 }
```

In the analysis, we can find that this function actually only uses two parameters, namely

a1 versus a7, And other parameters (a2 to a6) Has not been touched. In order to facilitate the identification of the cause,

Let's list its parameters a1 versus a7 The first use of:

● transfer_Function_base The function uses parameters a1;

● transfer sub_71496A0 The function uses parameters a7;

According to the calling convention, Ubuntu Version BDS Register rdi,rsi,rdx,rcx,

r8,r9 The first six parameters are passed, and the subsequent parameters are passed using the system stack. Sort it out now

The specific way to take out the parameters:

```
; __unwind { // __gxx_personality_v0
push    rbp
mov     rbp, rsp
sub     rsp, 30h
lea     rax, [rbp+arg_0]
mov     [rbp+var_8], rdi
mov     rdi, [rbp+var_8] ; this
mov     [rbp+var_20], rdi
mov     [rbp+var_28], rax
call    _ZNSt14_Function_baseC2Ev ; std::_Fun
mov     rdi, [rbp+var_28]
call    sub_71496A0
mov     [rbp+var_29], al
jmp     $+5
```

It can be seen that the parameters a1 Save in register rdi on,rdi At the beginning of execution

The value has not been changed, and the value of this register is completely saved to _Function_base letter

When the count starts to execute. parameter a7 Then from arg_0 Remove and save to rax, Subsequently insured

Save to var_28 On, and finally in the call sub_71496A0 Was removed and saved to rdi on.

Due to parameters a1 The first parameter is used (rdi) The delivery method, it

Be F5 The function is recognized as the first parameter, namely "a1", and the parameter a7 Is made up of the system stack (arg_0)

Passed, so it is recognized as the seventh parameter, which is "a7". Used to pass the second to the

The six parameter registers are not used to pass parameters. just because F5 Function on the system stack

The passed parameter is recognized as the seventh parameter, so F5 The function considers the second to the first

Six parameters exist, and added to the analysis results, this kind of trap appeared.F5

The function usually marks some of the second to sixth parameters in orange to indicate the parameter

The number has not been correctly analyzed.

(four) this The position of the structure pointed to by the pointer is incorrect

C++ When a class method calls other class methods of the same class instance, it will get itself

Got this The pointer is passed directly without any modification, because the memory of the same type of instance

The addresses are the same. The class method (function) in the figure below is calling other class methods of the same kind

Time, will give this The pointer plus an offset. This is an abnormal behavior. (The following figure

From Windows Version BDS)

```
void __fastcall ChestBlockActor::stopOpen(ChestBlockActor *this, struct  
{  
    ChestBlockActor::_closeChest((ChestBlockActor *)((char *)this - 248),  
}
```

At this point, if you want to use this class method (function), you need to figure out the this

Whether the class instance pointed to by the pointer is correct, this function needs to be called and there is a call relationship with it

The function does the analysis work.

Currently this trap is only in Windows Version BDS Appeared in.

1 1 . Debugging BDS process

In the process of reverse analysis, we often encounter the need to analyze certain variables under specific conditions

What kind of value will be obtained, or need to analyze the process of calling the function pointer

Which situation is the objective function, this time only depends on IDA Pro Correct BDS Static analysis

Or it is very troublesome to test the plug-in several times to get the result. Need to master BDS Process is running

The situation in the line.

In this section we will show on the alternative debugging methods, IDA Pro Allow the use of this

Local debugging, remote debugging is also allowed. In analysis BDS From the perspective, we do not recommend far

Because it has several serious drawbacks, one is relying on a stable network, and the other is

Compared with local debugging, it cannot be fully utilized IDA Pro The results have been automatically analyzed.

Local debugging has a very comfortable debugging experience.

The most ideal situation is to use IDA Pro Attach gdb Debugger debugging Ubuntu Version BDS,

But due to Linux Version IDA Pro The number of users is relatively small, and there are only 6.4 Version of

Linux Version IDA Pro, This version cannot attach local Linux Debugger. So it's a pity

We can't Linux Debugging in the environment BDS.

In this example we use Windows 10 Operating system to demonstrate how to use IDA Pro

Attach local Windbg Debugger to debug BDS process. The screenshots shown are all from

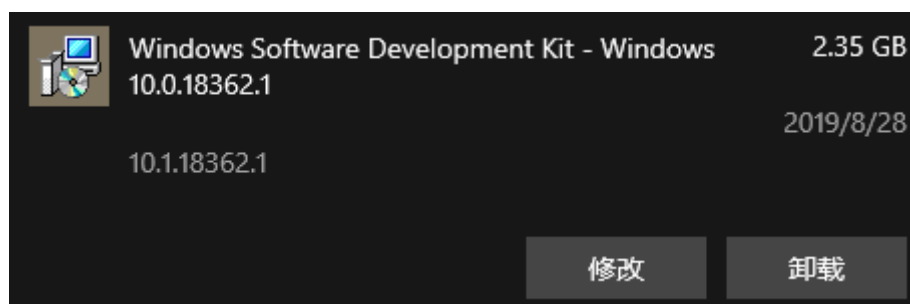
Windows Version BDS The debugging environment you are in.

(1) Download and install Windows SDK

Windows SDK Is for development Windows Application development kit, which provides

Tools for development and debugging. Before installation, please open the application list of the system to view Windows

SDK Whether it has been installed on the system, if it has been installed, please skip this step:



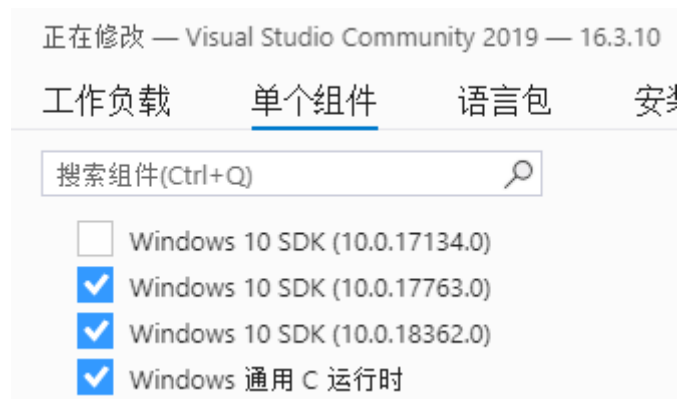
If it is not installed, click to open VS2019 Install the program, find the installed

VS2019 Entry, select the "modify" option, then select "single component", and finally find

To "Windows 10 SDK" Option. Because Windows 10 Because of the rolling update, will

Multiple Windows 10 SDK The version is in the list, here you only need to select the latest one

You don't need to install all of them.



(Two) choose to install Windbg

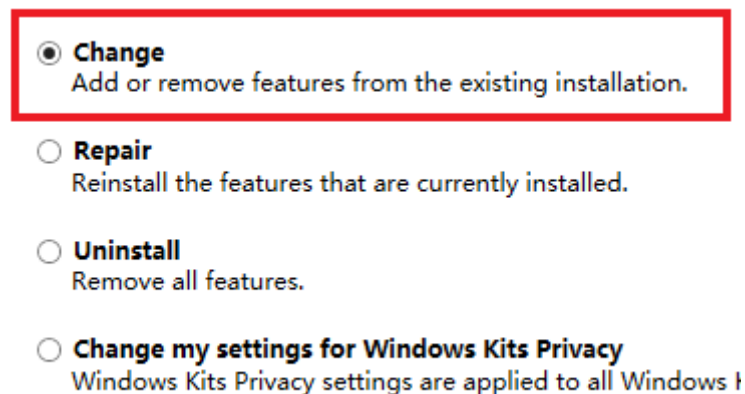
Windbg Yes Windows SDK Dedicated to debugging Windows The debugger of the process.

IDA Pro It can be used to debug the process.

First open the application list and find the installed Windows SDK, Click "Edit",

Will appear at this time Windows SDK Maintenance window, select in the single-selection list at this time

"Change" item, and then click "OK":



Then select "Debugging Tools for Windows" item:

Select the features you want to change

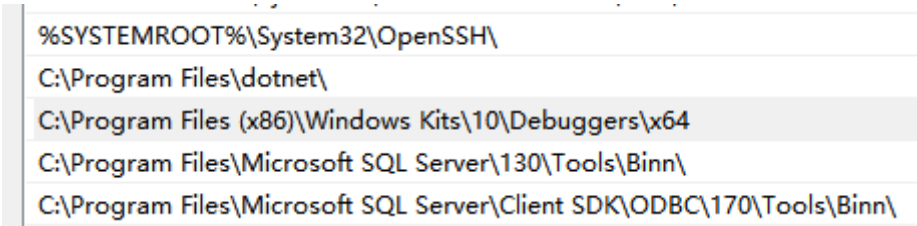
Click a feature name for more information.

<input type="checkbox"/> Windows Performance Toolkit	Windows Tools to rec for Window: in a graphic: Includes:
<input checked="" type="checkbox"/> Debugging Tools for Windows	
<input checked="" type="checkbox"/> Application Verifier For Windows	
<input type="checkbox"/> .NET Framework 4.7.2 Software Development Kit	
<input checked="" type="checkbox"/> Windows App Certification Kit	

Then click "Change" Button to save the modified options. At this time, the maintenance tool will automatically
installationWindbg To the computer.

Finally, in order to be used by other applicationsWindbg Debugger, we need to add

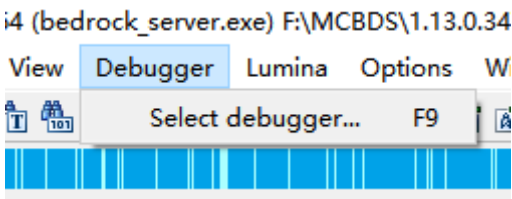
After installationWindbg The debugger program directory is added to the system PATH From the environment variables:



The path selected in the above figure is based on the system disk C plate,Windows SDK Installation path default
In the case of recognitionWindbg The location where it is installed. If you choose to customize during the installation process
Please change to a customized installation path.

(Three) in IDA Pro Set up the matching debugger on

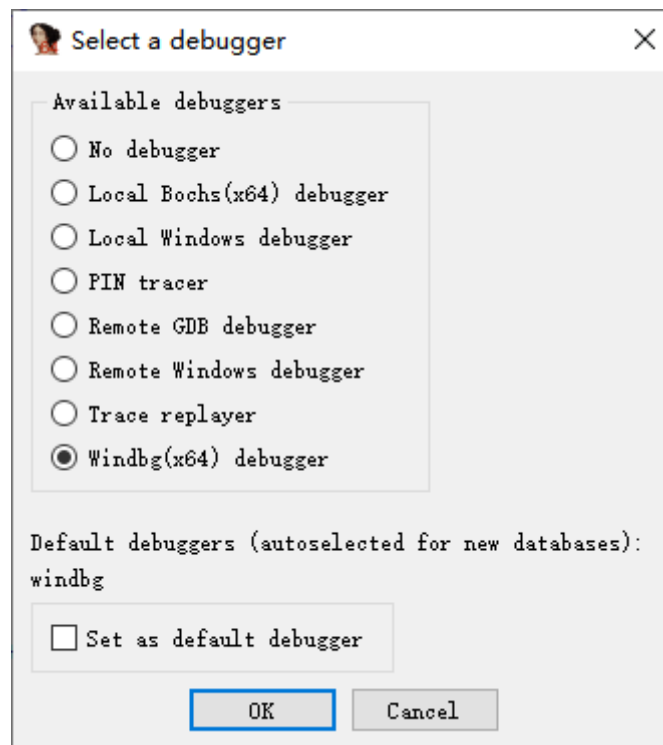
turn on IDA Pro And loadWindows Version BDS, And then click Menu on the menu bar
"Debugger":



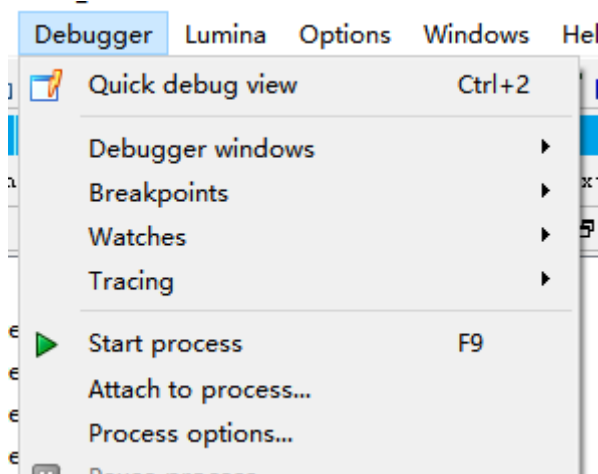
Click on the menu item "Select debugger", here will let the user choose to cooperate

Debugger type, here we select the single option "Windbg(x64) debugger", then

Click "OK" Button to save the selection results here.



After the selection is complete, click the menu again "Debugger":



Since we have already IDA Pro I have selected the debugger type, this time there are more menu bars

Many menu items dedicated to debugging.

(4) Preparation before debugging

By default, when the process being debugged is running, if the debugger is not

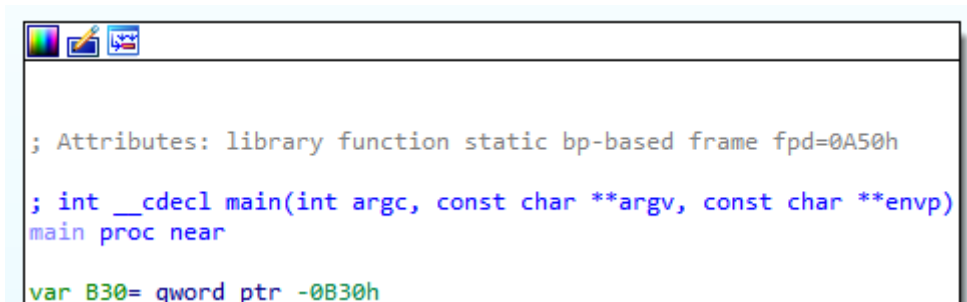
If you pause it automatically, it will only pause when it encounters a debug breakpoint or exception.

During the debugging process, the user needs to pause the process when it reaches a critical function.

Then start single-stepping to analyze the process of execution.

Here we choose BDS inside main Function to demonstrate the creation or cancellation of debugging breakpoints

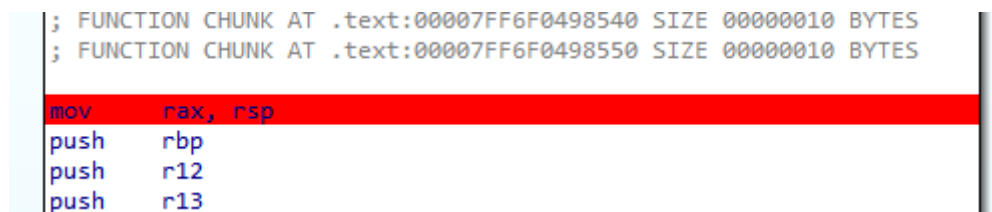
process. First find main The position of the function in the function list, open it in the process window:



```
; Attributes: library function static bp-based frame fpd=0A50h
; int __cdecl main(int argc, const char **argv, const char **envp)
main proc near
var_B30= qword ptr -0B30h
```

In order to be able to pause when this function is executed, we find main Function over

The first command in the program, and press F2 key:



```
; FUNCTION CHUNK AT .text:00007FF6F0498540 SIZE 00000010 BYTES
; FUNCTION CHUNK AT .text:00007FF6F0498550 SIZE 00000010 BYTES
mov     rax, rsp
push    rbp
push    r12
push    r13
```

It can be noticed that we press F2 When you press the key, the background of the instruction where the cursor is

It is marked in red, indicating that a debug breakpoint has been placed here. In the debugging state, the process is running

It will pause when the debug breakpoint is reached.

If you want to cancel the debugging breakpoint, place the cursor on the same line of the instruction marked in red, and then

Press down F2 Key to:



```
; FUNCTION CHUNK AT .text:00007FF6F0498550 SIZE 00000010 BYTES
mov     rax, rsp
push    rbp
push    r12
```

(5) Start the debugged process

Open the menu "Debugger", then click on the menu item "Start process", then

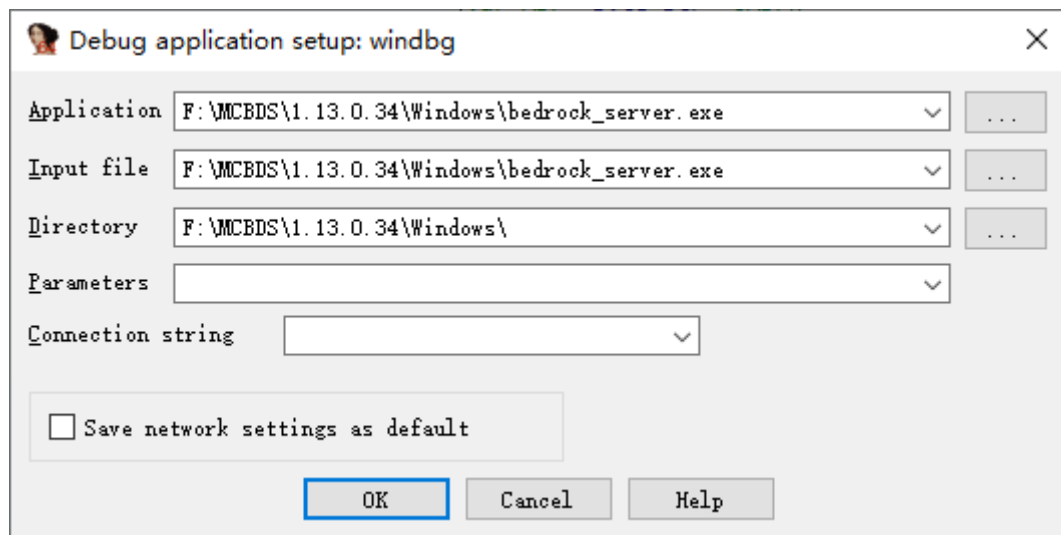
With IDA Pro Start attaching the debugger, ready to start the process being debugged. May have been

A warning box appears, prompting that this operation will be affected by the execution of the analysis

Ignore the warning here and click "OK" Button to continue to the next step.

During the first debugging, IDA Pro The following dialog box may pop up to make

The user configures the path of the executable file and its parameters and other information, here we will complete it:



It's important to note that this step is due to Windbg Does not support processing Chinese characters, such as

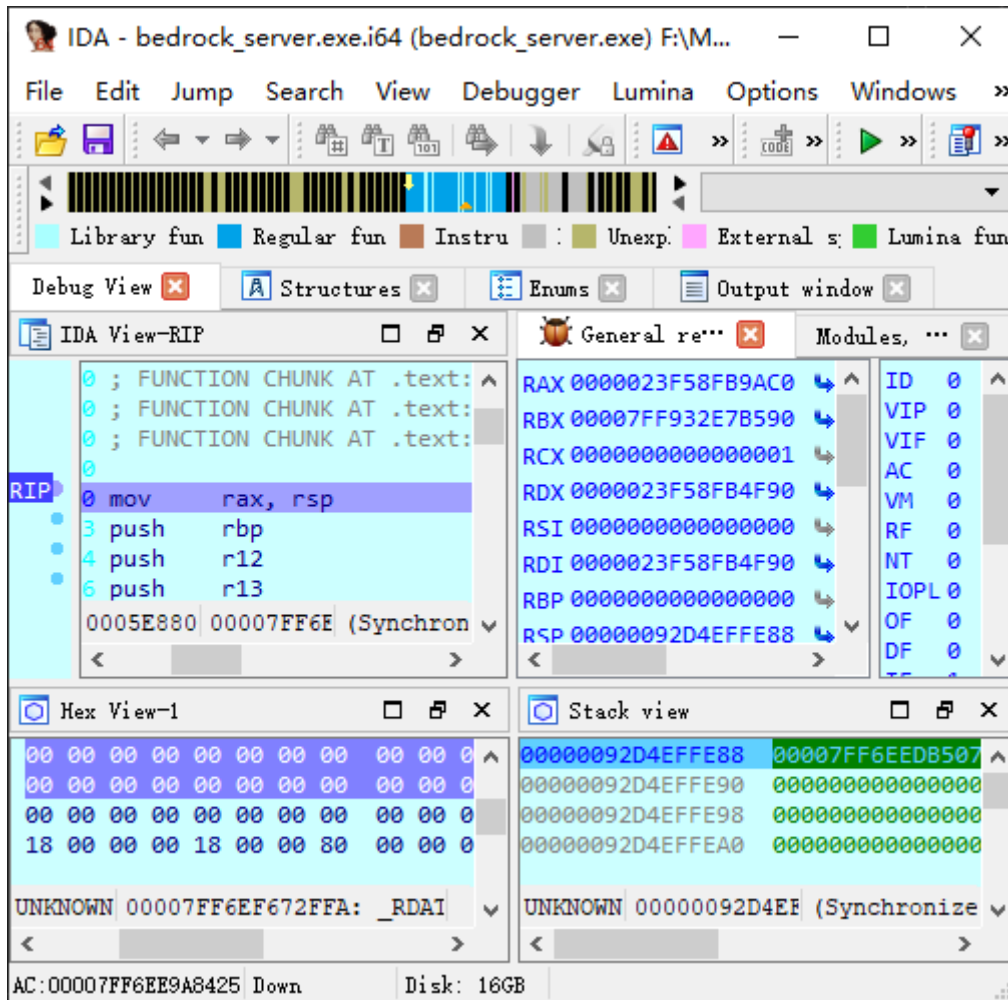
If debugged BDS The path of the main program file contains any Chinese characters, Then Windbg

Won't find BDS The main program cannot start debugging.

Finally click the button "OK", start the process being debugged.

(6) Monitoring the running status of the process

After the debugged process is started, IDA Pro Switch the layout in the window to the debug layout:



There are disassembly view, register view, memory view and stack view in this layout. this

Time, the process state has been in our previous main Pause on the debug breakpoint set on the function

Up, RIP Register instructions after the process resumes CPU The next instruction to be executed.

IDA Pro Some relatively convenient debugging shortcut keys are provided:

F7: Step into, execute one complete assembly instruction each time.

F8: Single step over, generally similar to single step into function, the difference is CALL

When ordering, it will wait for the execution flow to reach here CALL The next immediately after the instruction

Instructions.

F4: Execute to the cursor, before pressing this key, please set the line where the cursor is.

If it does not reach the cursor, the program will continue to execute.

(7) Analyze the operation status

Now let's try to analyze the current environment of the function:

First of all, our debug breakpoint is at the head of the function, at this time the function also

Not executed, we can see the uncontaminated parameters completely, which are argc, argv with envp, The types are int, char** with char**.

Then according to Microsoft x64 Calling convention, register RCX, RDX, R8, R9 Used to protect

Save the first four parameters passed to the called function, then because there are only 3 Parameters, only

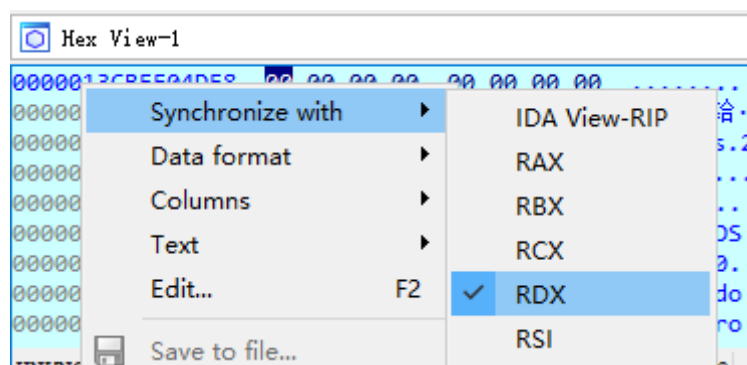
Have RCX, RDX, R8 They are used to save three parameters:

```
RAX 0000013CBEF09A10
RBX 00007FFB9223B590
RCX 0000000000000001
RDX 0000013CBEF04E10
RSI 0000000000000000
RDI 0000013CBEF04E10
RBP 0000000000000000
RSP 0000001CFF0FF728
RIP 00007FF75583F480
R8 0000013CBEF09A10
R9 0000013CBFA2400
```

RCX for 1, Which means there is only one program start parameter received. RDX with R8 respectively

for char**Pointer, we use the synchronization view option of the memory view right-click menu to track this

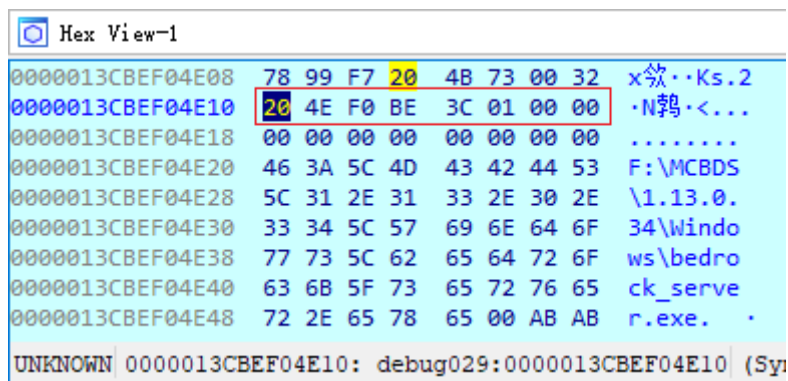
The position pointed to by the two pointers, first RDX:



due to RDX The saved value is char**, Then RDX The value of this memory pointed to should

Yes char*Type in 64-bit system, all types of pointers have the same length, which are 64-bit

Bit, that is 8 Bytes:



x86-64 The platform uses little endian (Little endian), the upper byte in the figure

The lower the position in the data, the higher the position of the lower byte in the data. Then here

char*The content of the type data is 0 x0000013CBEF04E20, Point to the one behind

"F:\MCBDS"At the beginning of the string. Obviously, this is BDS Own path character

string. Backenvp You can also use the same method to see what it can provide, here

Do not repeat the demonstration.

(8) End of debugging

After the analysis work is completed, you need to exit the debugging environment. There are two ways to turn off debugging:

use IDA Pro menu "Debugger" Menu item under "Terminate process" termination

Debug and close the debugged BDS process. Or let BDS Continue to execute, enter in the background

stop Order to let BDS Close by itself. BDS After closing, the debugging environment will automatically exit. usually,

Recommended Use stop Command to close the server opener, so that the server opener can count

It is kept well.

1 2 . Conclusion

At this point, the content of this tutorial is over. I hope that the beneficiaries of this tutorial can help

Minecraft The bedrock edition server plug-in makes its own contribution, after all, the bedrock edition plays

The home is gradually losing, among which there is an intention to use Windows Version BDS The yuba with plug-in is rare

The status quo is not optimistic. In addition, if you find an error in the tutorial or explain it poorly

Place, please give me feedback.

my contact information:

QQ: 2 4 4 6 8 3 8 7 6

mailbox:Player.MS@outlook.com