Ghidra

Reverse Engineering Toolkit

Who I am

https://nstarke.github.io/

https://soundcloud.com/nicholas-starke

@nstarke

Reverse Engineer / Threat Researcher for Hewlett Packard Enterprise / Aruba Networks

What is Ghidra?

- Reverse Engineering Toolkit for analyzing natively compiled executables
- Competitor to IDA Pro / Radare 2, amongst others
- Created by National Security Agency and open sourced in 2019

Where can I download ghidra?

Releases:

https://ghidra-sre.org/

Source Code:

https://github.com/NationalSecurityAgency/ghidra

Benefits of Using Ghidra

- Open source so it’s free.
- Supports many common Instruction Set Architectures (ARM, MIPS, Intel x86, etc).
- Developed by brilliant people so it works really well.
- Superior decompiler output.

Cons of using Ghidra

- No paid support
- No debugger in released branch yet.
- Doesn’t support obscure CPU architectures like IDA pro does.

What is Ghidra Typically Used for?

- Malware Analysis:
  - How does this malware binary work?
  - What does it do on infected systems?
- Vulnerability Research
  - Are there exploitable security issues in a given binary program?

Vulnerability Research

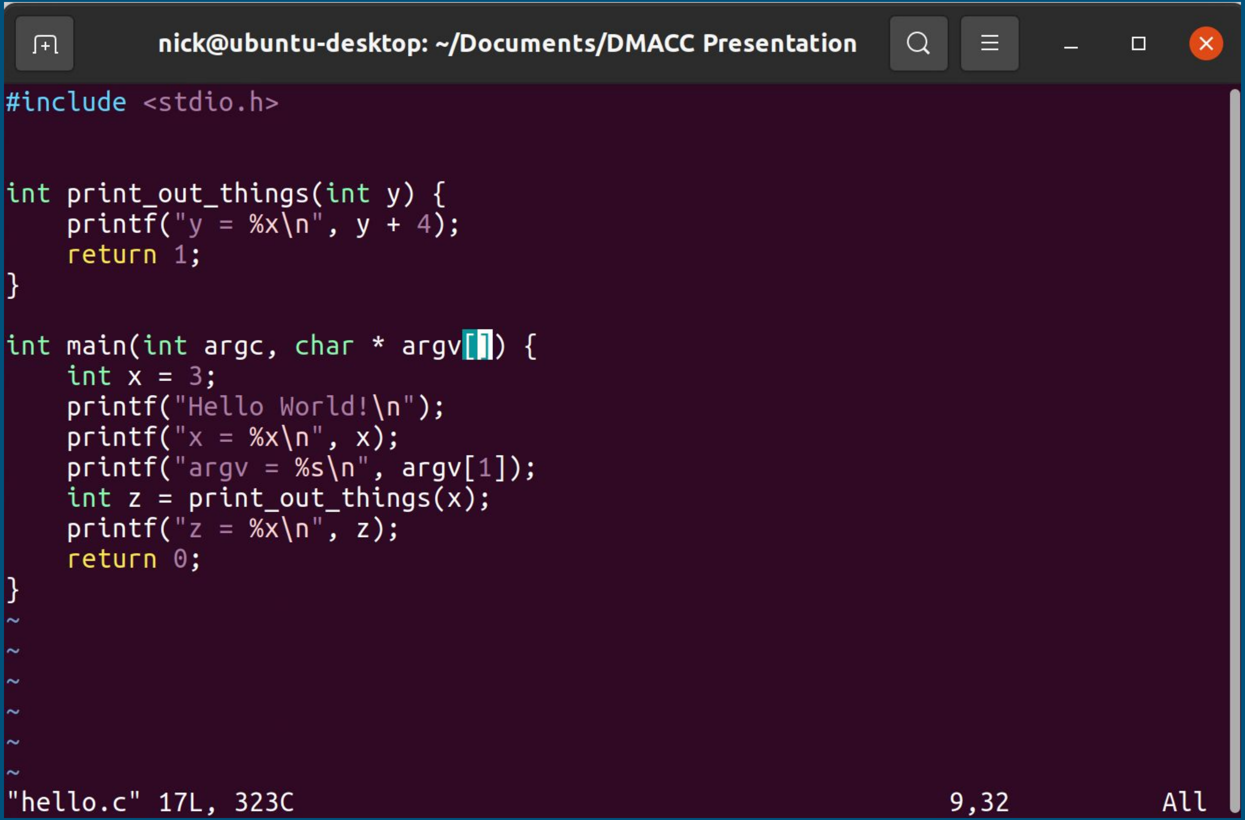
I use Ghidra primarily for Vulnerability Research. Two reasons:

1) High level decompiler output is best in class
2) It is free (IDA Pro licensing is very expensive)

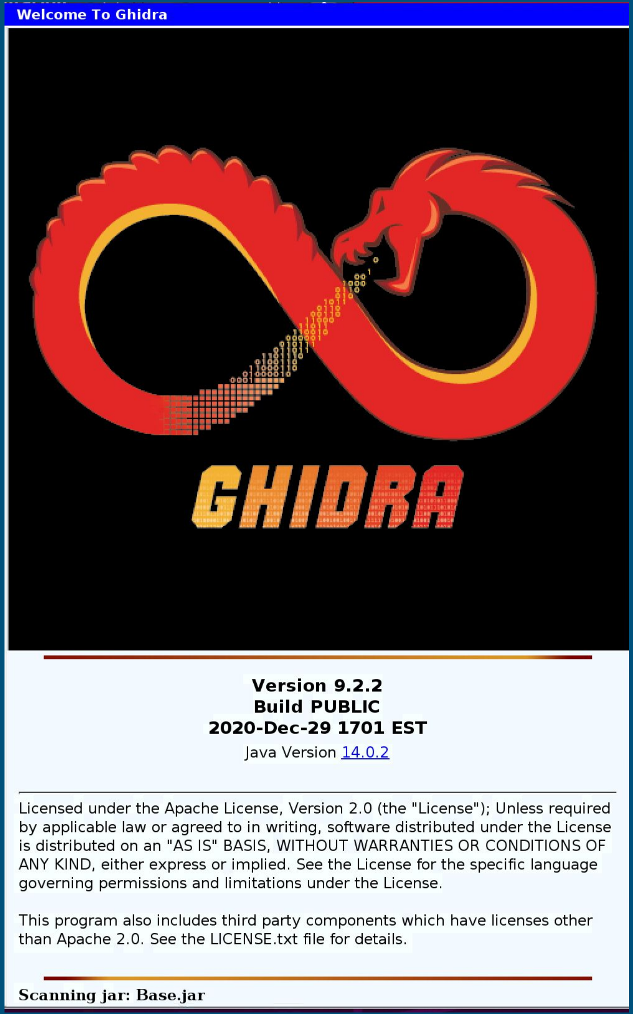
2 Phases of Reverse Engineering

1) Application level
    a) Concerned with over all code flow and capability

Sample C Program (Linux)



```
#include <stdio.h>


int print_out_things(int y) {
    printf("y = %x\n", y + 4);
    return 1;
}

int main(int argc, char * argv[]) {
    int x = 3;
    printf("Hello World!\n");
    printf("x = %x\n", x);
    printf("argv = %s\n", argv[1]);
    int z = print_out_things(x);
    printf("z = %x\n", z);
    return 0;
}
~
~
~
~
~
~
"hello.c" 17L, 323C                                         9,32          All
```

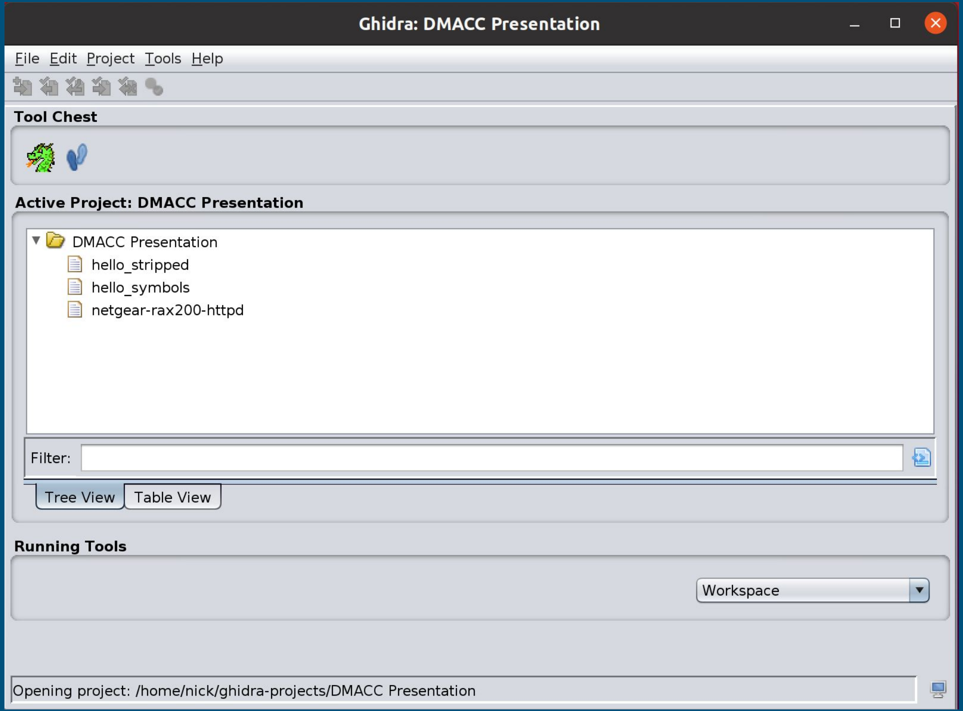


Welcome To Ghidra

Version 9.2.2
Build PUBLIC
2020-Dec-29 1701 EST
Java Version 14.0.2

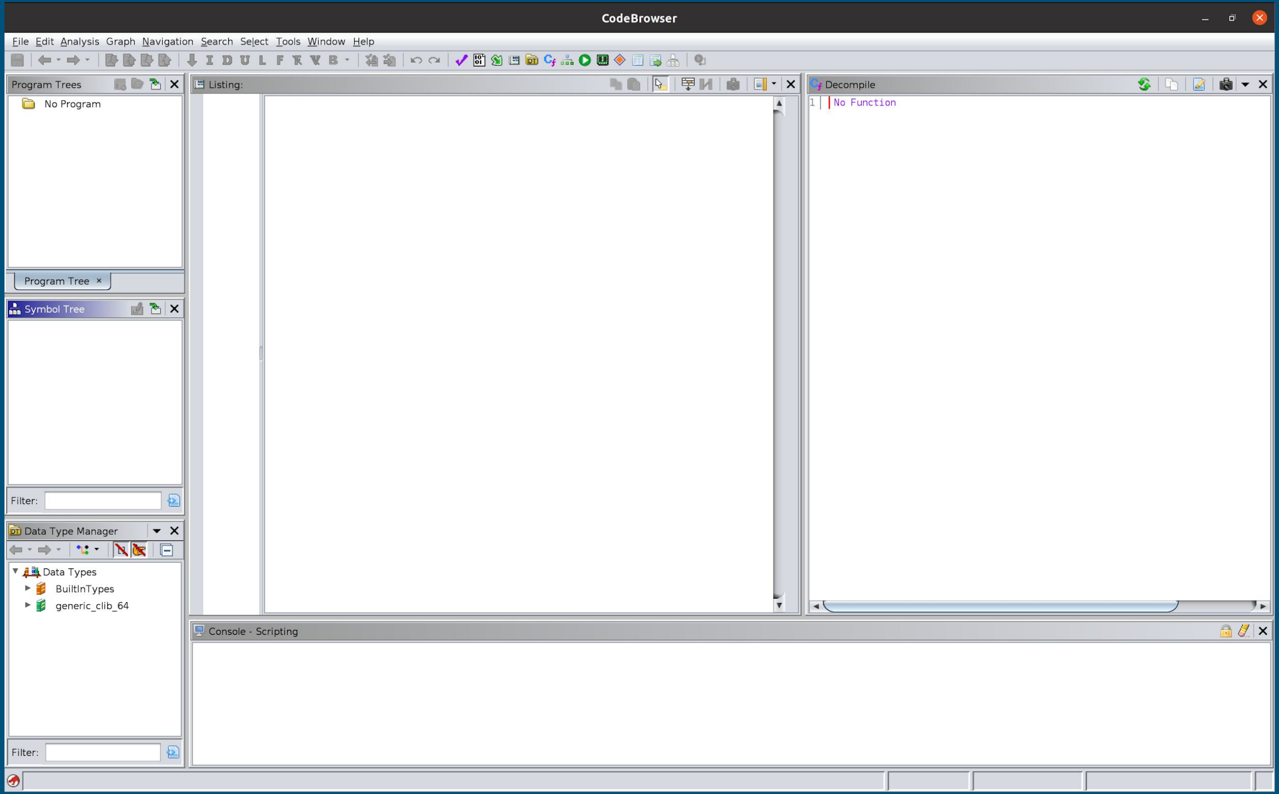
Scanning jar: Base.jar

Ghidra Project Menu

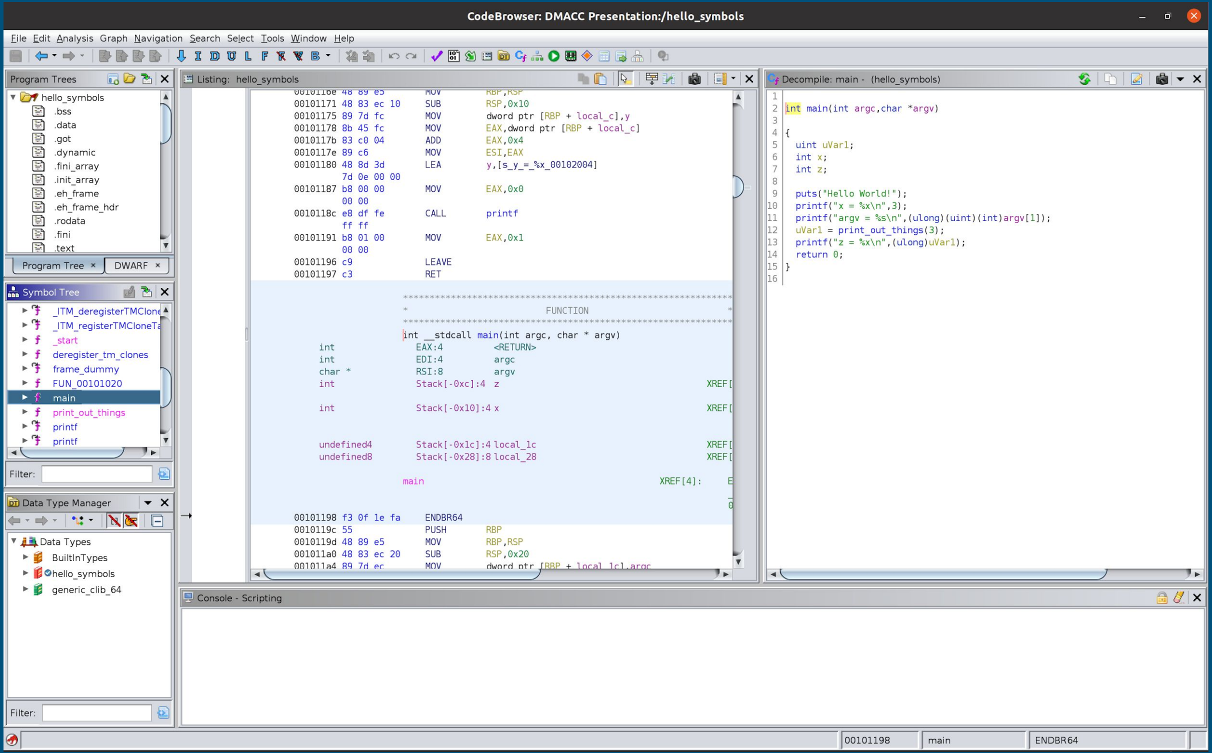


Opening project: /home/nick/ghidra-projects/DMACC Presentation

Blank Ghidra Code Browser



Ghidra with open project



CodeBrowser: DMACC Presentation:/hello_symbols
File Edit Analysis Graph Navigation Search Select Tools Window Help
Program Trees
Program Tree
DWARF
Symbol Tree
Filter:
Data Type Manager
Listing: hello_symbols
Decompile: main - (hello_symbols)
Console - Scripting

```
hello_symbols
    .bss
    .data
    .got
    .dynamic
    .fini_array
    .init_array
    .eh_frame
    .eh_frame_hdr
    .rodata
    .fini
    .text
```

```
_ITM_deregisterTMClone
_ITM_registerTMCloneTa
_start
deregister_tm_clones
frame_dummy
FUN_00101020
main
print_out_things
printf
printf
```

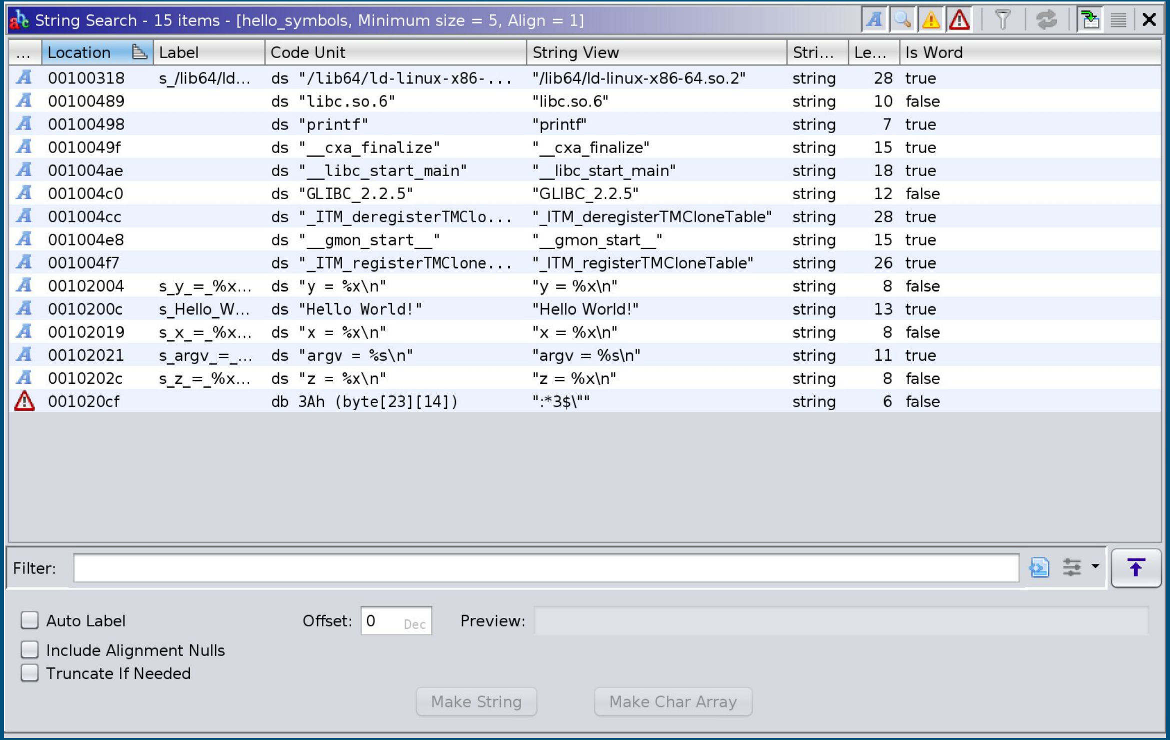
```
Data Types
    BuiltInTypes
    hello_symbols
    generic_clib_64
```

```
0010116e 48 89 e5      MOV      RBP,RSP
00101171 48 83 ec 10   SUB      RSP,0x10
00101175 89 7d fc      MOV      dword ptr [RBP + local_c],y
00101178 8b 45 fc      MOV      EAX,dword ptr [RBP + local_c]
0010117b 83 c0 04      ADD      EAX,0x4
0010117e 89 c6         MOV      ESI,EAX
00101180 48 8d 3d      LEA      y,[s_y_=_%x_00102004]
         7d 0e 00 00
00101187 b8 00 00      MOV      EAX,0x0
         00 00
0010118c e8 df fe      CALL     printf
         ff ff
00101191 b8 01 00      MOV      EAX,0x1
         00 00
00101196 c9            LEAVE
00101197 c3            RET

                       ******************************************************
                       *                       FUNCTION                     *
                       ******************************************************
                       int __stdcall main(int argc, char * argv)
     int                 EAX:4          <RETURN>
     int                 EDI:4          argc
     char *              RSI:8          argv
     int                 Stack[-0xc]:4  z                          XREF[
     
     int                 Stack[-0x10]:4 x                          XREF[


     undefined4          Stack[-0x1c]:4 local_1c                   XREF[
     undefined8          Stack[-0x28]:8 local_28                   XREF[

                       main                                XREF[4]:     E

                                                                        0
00101198 f3 0f 1e fa   ENDBR64
0010119c 55            PUSH     RBP
0010119d 48 89 e5      MOV      RBP,RSP
001011a0 48 83 ec 20   SUB      RSP,0x20
001011a4 89 7d ec      MOV      dword ptr [RBP + local_1c],argc
```

```c

int main(int argc,char *argv)

{
  uint uVar1;
  int x;
  int z;

  puts("Hello World!");
  printf("x = %x\n",3);
  printf("argv = %s\n",(ulong)(uint)(int)argv[1]);
  uVar1 = print_out_things(3);
  printf("z = %x\n",(ulong)uVar1);
  return 0;
}

```

```
00101198    main    ENDBR64
```

Ghidra Strings Menu



String Search - 15 items - [hello_symbols, Minimum size = 5, Align = 1]

...	Location	Label	Code Unit	String View	Stri...	Le...	Is Word
A	00100318	s_/lib64/ld...	ds "/lib64/ld-linux-x86-...	"/lib64/ld-linux-x86-64.so.2"	string	28	true
A	00100489		ds "libc.so.6"	"libc.so.6"	string	10	false
A	00100498		ds "printf"	"printf"	string	7	true
A	0010049f		ds "__cxa_finalize"	"__cxa_finalize"	string	15	true
A	001004ae		ds "__libc_start_main"	"__libc_start_main"	string	18	true
A	001004c0		ds "GLIBC_2.2.5"	"GLIBC_2.2.5"	string	12	false
A	001004cc		ds "_ITM_deregisterTMClo...	"_ITM_deregisterTMCloneTable"	string	28	true
A	001004e8		ds "__gmon_start__"	"__gmon_start__"	string	15	true
A	001004f7		ds "_ITM_registerTMClone...	"_ITM_registerTMCloneTable"	string	26	true
A	00102004	s_y_=_%x...	ds "y = %x\n"	"y = %x\n"	string	8	false
A	0010200c	s_Hello_W...	ds "Hello World!"	"Hello World!"	string	13	true
A	00102019	s_x_=_%x...	ds "x = %x\n"	"x = %x\n"	string	8	false
A	00102021	s_argv_=_...	ds "argv = %s\n"	"argv = %s\n"	string	11	true
A	0010202c	s_z_=_%x...	ds "z = %x\n"	"z = %x\n"	string	8	false
⚠	001020cf		db 3Ah (byte[23][14])	":*3$\""	string	6	false

Filter:

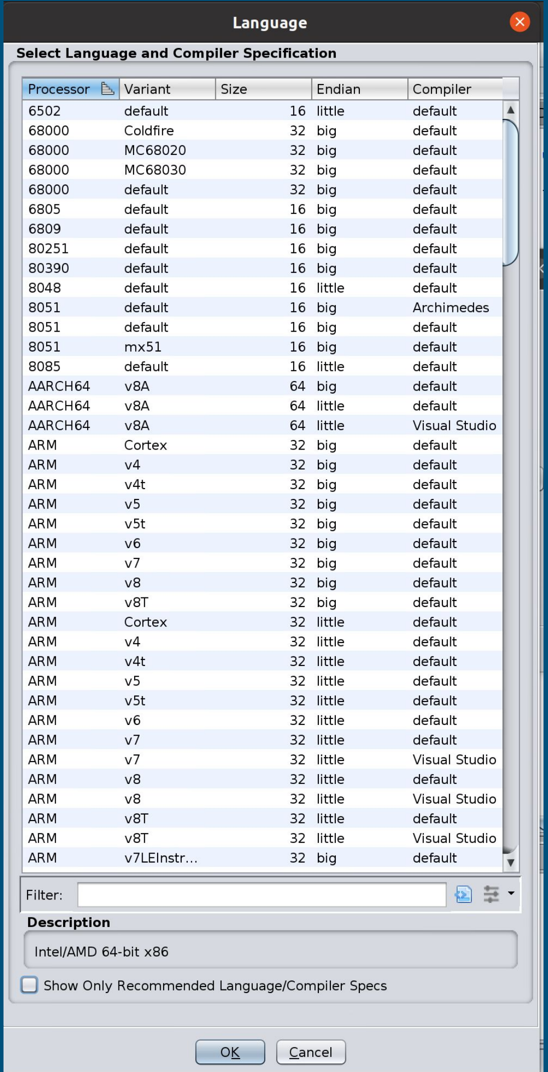
☐ Auto Label

☐ Include Alignment Nulls

☐ Truncate If Needed

Offset: 0 Dec

Preview:



Language

Select Language and Compiler Specification

Processor	Variant	Size	Endian	Compiler
6502	default	16	little	default
68000	Coldfire	32	big	default
68000	MC68020	32	big	default
68000	MC68030	32	big	default
68000	default	32	big	default
6805	default	16	big	default
6809	default	16	big	default
80251	default	16	big	default
80390	default	16	big	default
8048	default	16	little	default
8051	default	16	big	Archimedes
8051	default	16	big	default
8051	mx51	16	big	default
8085	default	16	little	default
AARCH64	v8A	64	big	default
AARCH64	v8A	64	little	default
AARCH64	v8A	64	little	Visual Studio
ARM	Cortex	32	big	default
ARM	v4	32	big	default
ARM	v4t	32	big	default
ARM	v5	32	big	default
ARM	v5t	32	big	default
ARM	v6	32	big	default
ARM	v7	32	big	default
ARM	v8	32	big	default
ARM	v8T	32	big	default
ARM	Cortex	32	little	default
ARM	v4	32	little	default
ARM	v4t	32	little	default
ARM	v5	32	little	default
ARM	v5t	32	little	default
ARM	v6	32	little	default
ARM	v7	32	little	default
ARM	v7	32	little	Visual Studio
ARM	v8	32	little	default
ARM	v8	32	little	Visual Studio
ARM	v8T	32	little	default
ARM	v8T	32	little	Visual Studio
ARM	v7LEInstr...	32	big	default

Filter:

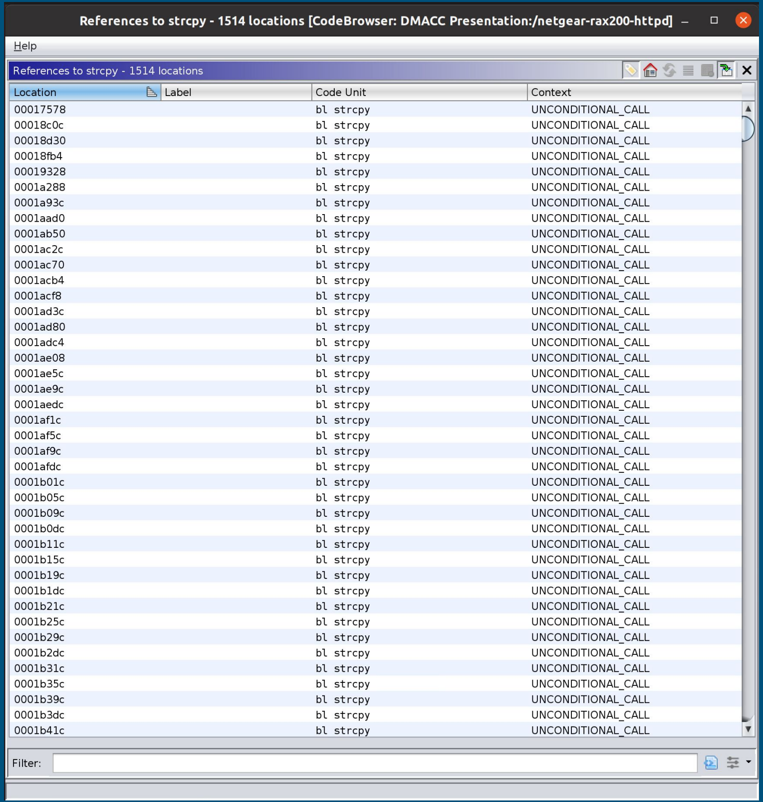
Description

Intel/AMD 64-bit x86

☐ Show Only Recommended Language/Compiler Specs

OK Cancel

Ghidra References



References to strcpy - 1514 locations [CodeBrowser: DMACC Presentation:/netgear-rax200-httpd]

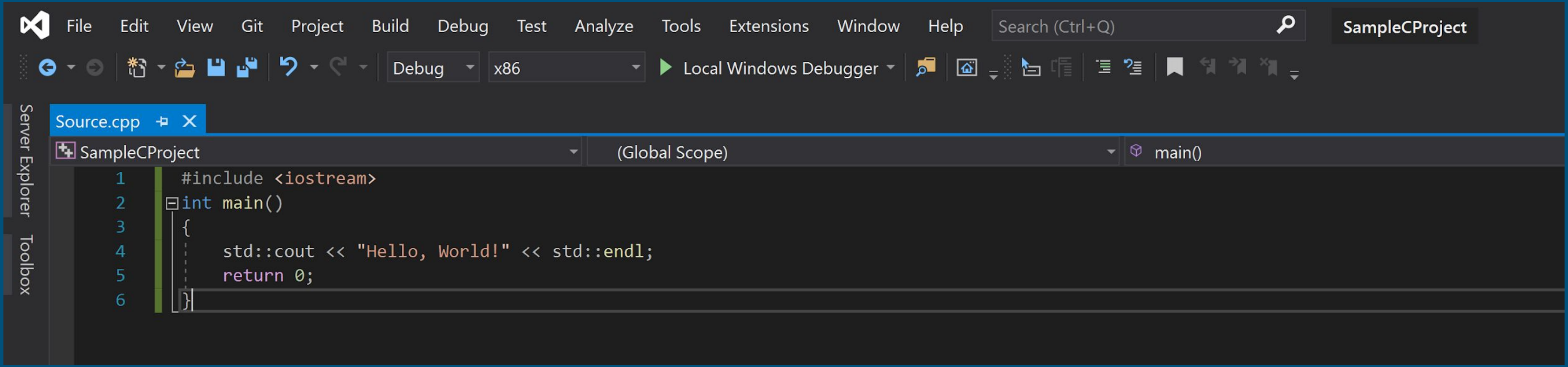
Help

References to strcpy - 1514 locations

Location	Label	Code Unit	Context
00017578		bl strcpy	UNCONDITIONAL_CALL
00018c0c		bl strcpy	UNCONDITIONAL_CALL
00018d30		bl strcpy	UNCONDITIONAL_CALL
00018fb4		bl strcpy	UNCONDITIONAL_CALL
00019328		bl strcpy	UNCONDITIONAL_CALL
0001a288		bl strcpy	UNCONDITIONAL_CALL
0001a93c		bl strcpy	UNCONDITIONAL_CALL
0001aad0		bl strcpy	UNCONDITIONAL_CALL
0001ab50		bl strcpy	UNCONDITIONAL_CALL
0001ac2c		bl strcpy	UNCONDITIONAL_CALL
0001ac70		bl strcpy	UNCONDITIONAL_CALL
0001acb4		bl strcpy	UNCONDITIONAL_CALL
0001acf8		bl strcpy	UNCONDITIONAL_CALL
0001ad3c		bl strcpy	UNCONDITIONAL_CALL
0001ad80		bl strcpy	UNCONDITIONAL_CALL
0001adc4		bl strcpy	UNCONDITIONAL_CALL
0001ae08		bl strcpy	UNCONDITIONAL_CALL
0001ae5c		bl strcpy	UNCONDITIONAL_CALL
0001ae9c		bl strcpy	UNCONDITIONAL_CALL
0001aedc		bl strcpy	UNCONDITIONAL_CALL
0001af1c		bl strcpy	UNCONDITIONAL_CALL
0001af5c		bl strcpy	UNCONDITIONAL_CALL
0001af9c		bl strcpy	UNCONDITIONAL_CALL
0001afdc		bl strcpy	UNCONDITIONAL_CALL
0001b01c		bl strcpy	UNCONDITIONAL_CALL
0001b05c		bl strcpy	UNCONDITIONAL_CALL
0001b09c		bl strcpy	UNCONDITIONAL_CALL
0001b0dc		bl strcpy	UNCONDITIONAL_CALL
0001b11c		bl strcpy	UNCONDITIONAL_CALL
0001b15c		bl strcpy	UNCONDITIONAL_CALL
0001b19c		bl strcpy	UNCONDITIONAL_CALL
0001b1dc		bl strcpy	UNCONDITIONAL_CALL
0001b21c		bl strcpy	UNCONDITIONAL_CALL
0001b25c		bl strcpy	UNCONDITIONAL_CALL
0001b29c		bl strcpy	UNCONDITIONAL_CALL
0001b2dc		bl strcpy	UNCONDITIONAL_CALL
0001b31c		bl strcpy	UNCONDITIONAL_CALL
0001b35c		bl strcpy	UNCONDITIONAL_CALL
0001b39c		bl strcpy	UNCONDITIONAL_CALL
0001b3dc		bl strcpy	UNCONDITIONAL_CALL
0001b41c		bl strcpy	UNCONDITIONAL_CALL

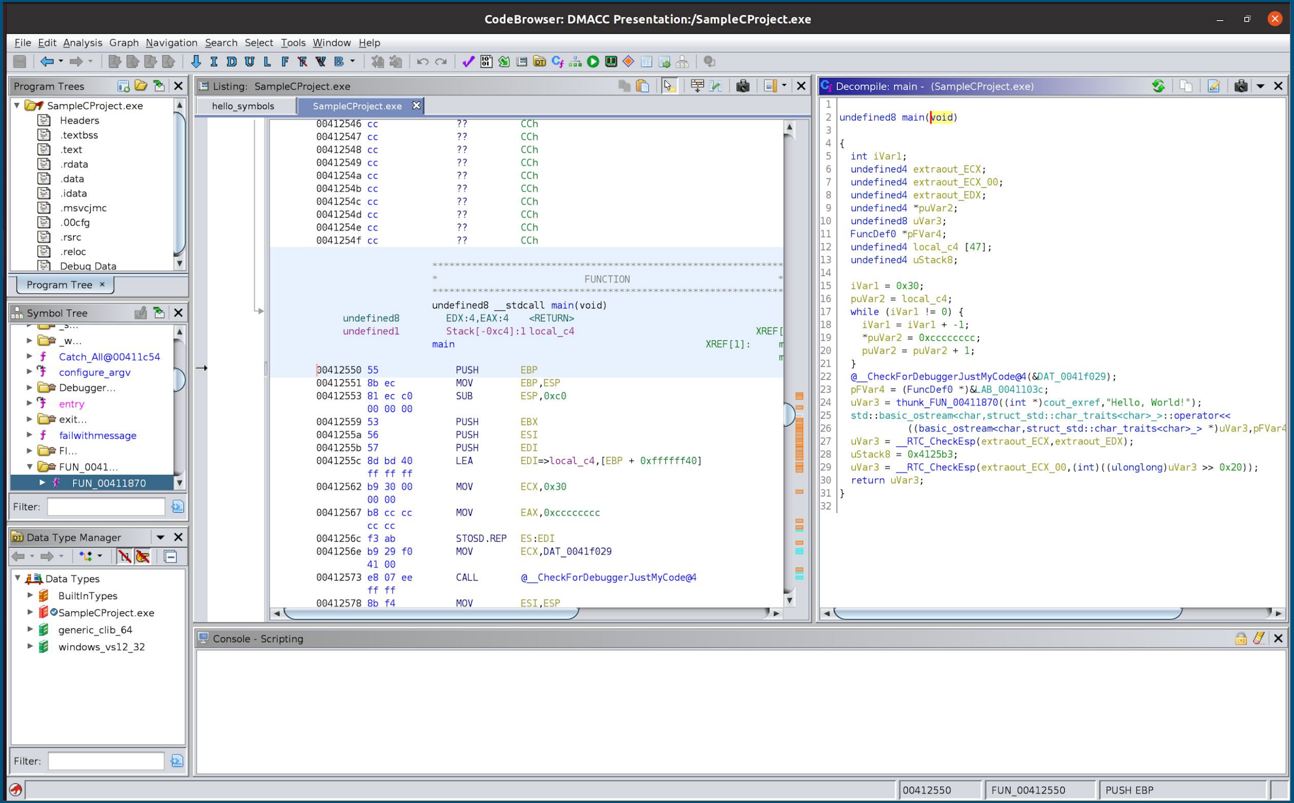
Filter:

Sample C++ Program (Windows)



```
#include <iostream>
int main()
{
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

Sample C Program (Windows) - in Ghidra



```
00412546 cc          ??         CCh
00412547 cc          ??         CCh
00412548 cc          ??         CCh
00412549 cc          ??         CCh
0041254a cc          ??         CCh
0041254b cc          ??         CCh
0041254c cc          ??         CCh
0041254d cc          ??         CCh
0041254e cc          ??         CCh
0041254f cc          ??         CCh

                    ******************************************************************
                    *                            FUNCTION                            *
                    ******************************************************************
                    undefined8 __stdcall main(void)
     undefined8       EDX:4,EAX:4    <RETURN>
     undefined1       Stack[-0xc4]:1 local_c4                              XREF[
                    main                                           XREF[1]:     m
                                                                                m
00412550 55             PUSH       EBP
00412551 8b ec          MOV        EBP,ESP
00412553 81 ec c0       SUB        ESP,0xc0
         00 00 00
00412559 53             PUSH       EBX
0041255a 56             PUSH       ESI
0041255b 57             PUSH       EDI
0041255c 8d bd 40       LEA        EDI=>local_c4,[EBP + 0xffffff40]
         ff ff ff
00412562 b9 30 00       MOV        ECX,0x30
         00 00
00412567 b8 cc cc       MOV        EAX,0xcccccccc
         cc cc
0041256c f3 ab          STOSD.REP  ES:EDI
0041256e b9 29 f0       MOV        ECX,DAT_0041f029
         41 00
00412573 e8 07 ee       CALL       @__CheckForDebuggerJustMyCode@4
         ff ff
00412578 8b f4          MOV        ESI,ESP
```

```c

undefined8 main(void)

{
  int iVar1;
  undefined4 extraout_ECX;
  undefined4 extraout_ECX_00;
  undefined4 extraout_EDX;
  undefined4 *puVar2;
  undefined8 uVar3;
  FuncDef0 *pFVar4;
  undefined4 local_c4 [47];
  undefined4 uStack8;

  iVar1 = 0x30;
  puVar2 = local_c4;
  while (iVar1 != 0) {
    iVar1 = iVar1 + -1;
    *puVar2 = 0xcccccccc;
    puVar2 = puVar2 + 1;
  }
  @__CheckForDebuggerJustMyCode@4(&DAT_0041f029);
  pFVar4 = (FuncDef0 *)&LAB_0041103c;
  uVar3 = thunk_FUN_00411870((int *)cout_exref,"Hello, World!");
  std::basic_ostream<char,struct_std::char_traits<char>_>::operator<<
            ((basic_ostream<char,struct_std::char_traits<char>_> *)uVar3,pFVar4
  uVar3 = __RTC_CheckEsp(extraout_ECX,extraout_EDX);
  uStack8 = 0x4125b3;
  uVar3 = __RTC_CheckEsp(extraout_ECX_00,(int)((ulonglong)uVar3 >> 0x20));
  return uVar3;
}

```

Reverse Engineering Challenges

https://crackmes.one/

A free collection of reverse engineering challenges, like HackTheBox / TryHackMe.

These challenges are great for practicing reverse engineering.

Additional Learning Resources

Documentation: https://ghidra.re/ghidra_docs/api/index.html

Book: https://nostarch.com/GhidraBook

Questions?

Thank you!