Advanced Topics in Malware Analysis

Welcome to GIDHRA

Brendan Saltaformaggio, PhD

Assistant Professor

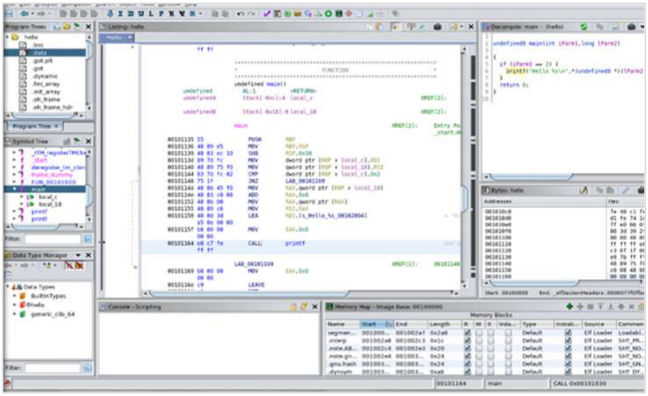
School of Electrical and Computer Engineering

GIDHRA

1

What is GHIDRA?

- Ghidra was designed and built by the United States NSA
- Ghidra combines an interactive, programmable, multi-processor disassembler with an in-house decompiler and is augmented by a complete plugin environment
- Although Ghidra was only recently released to the public, it has been used internally by the US government for years **and it is now free and open source**



```
undefined8 main(int iParm1,long lParm2)

{
  if (iParm1 == 2) {
    printf("Hello %s\n",*(undefined8 *)(lParm2
  }
  return 0;
}
```

2

How to Access GHIDRA

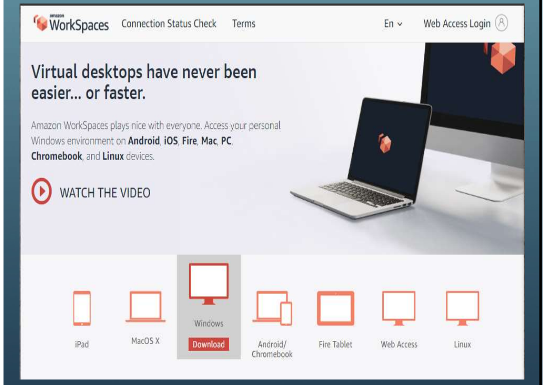
Georgia Tech

3

AWS Workspace

- Ghidra is installed in the AWS cloud servers
- To access GHIDRA:

1) Download AWS workspace from https://clients.amazonworkspaces.com/ for your OS.



Virtual desktops have never been easier... or faster.

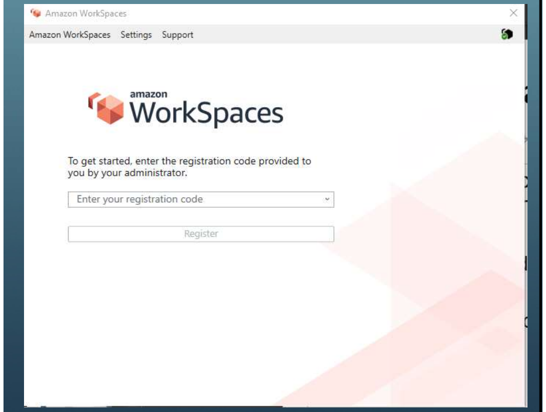
Amazon WorkSpaces plays nice with everyone. Access your personal Windows environment on **Android**, **iOS**, **Fire**, **Mac**, **PC**, **Chromebook**, and **Linux** devices.

Georgia Tech

4

AWS Workspace

- After you have successfully downloaded and installed AWS workspace,
  - You will be prompted to enter **registration code**
  - Use the registration code you received with this class.



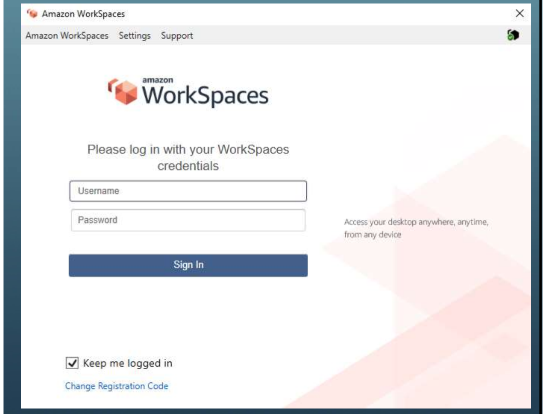
To get started, enter the registration code provided to you by your administrator.

Enter your registration code

5

AWS Workspace

- After registering you will be prompted to enter username and password.
  - Username: **gpbrudell123**
  - Password: Use password to access buzzport



Please log in with your WorkSpaces credentials

Username

Password

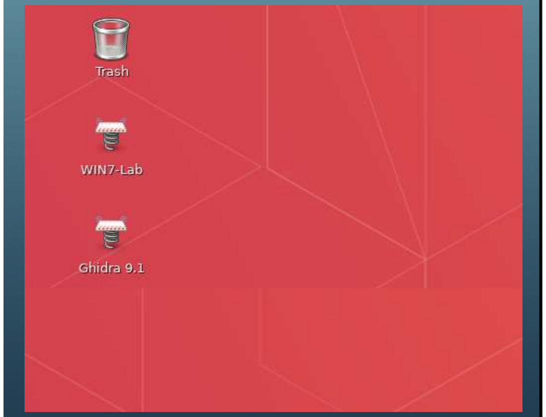
Access your desktop anywhere, anytime, from any device

☑ Keep me logged in

6

AWS Workspace

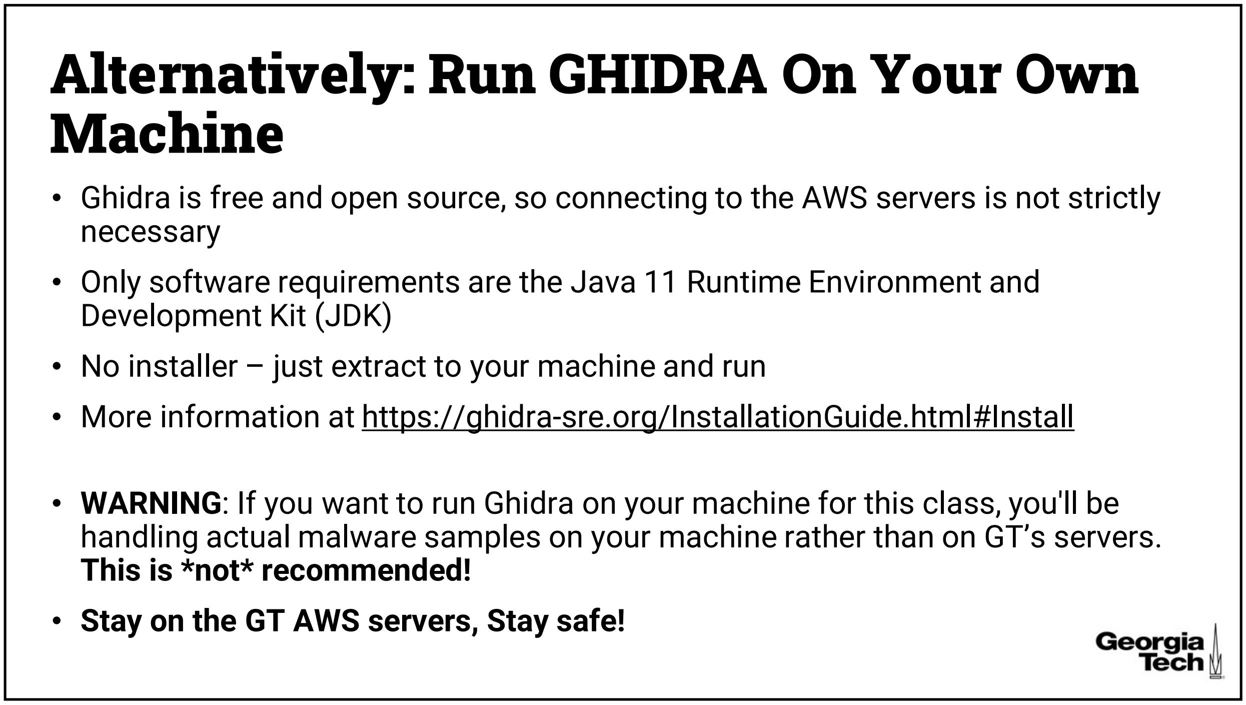
- After signing in you will see something similar to what is shown in the picture to start GHIDRA.
- Double click on to get started.



7

But, Professor, Can't I run Ghidra on my laptop??

8

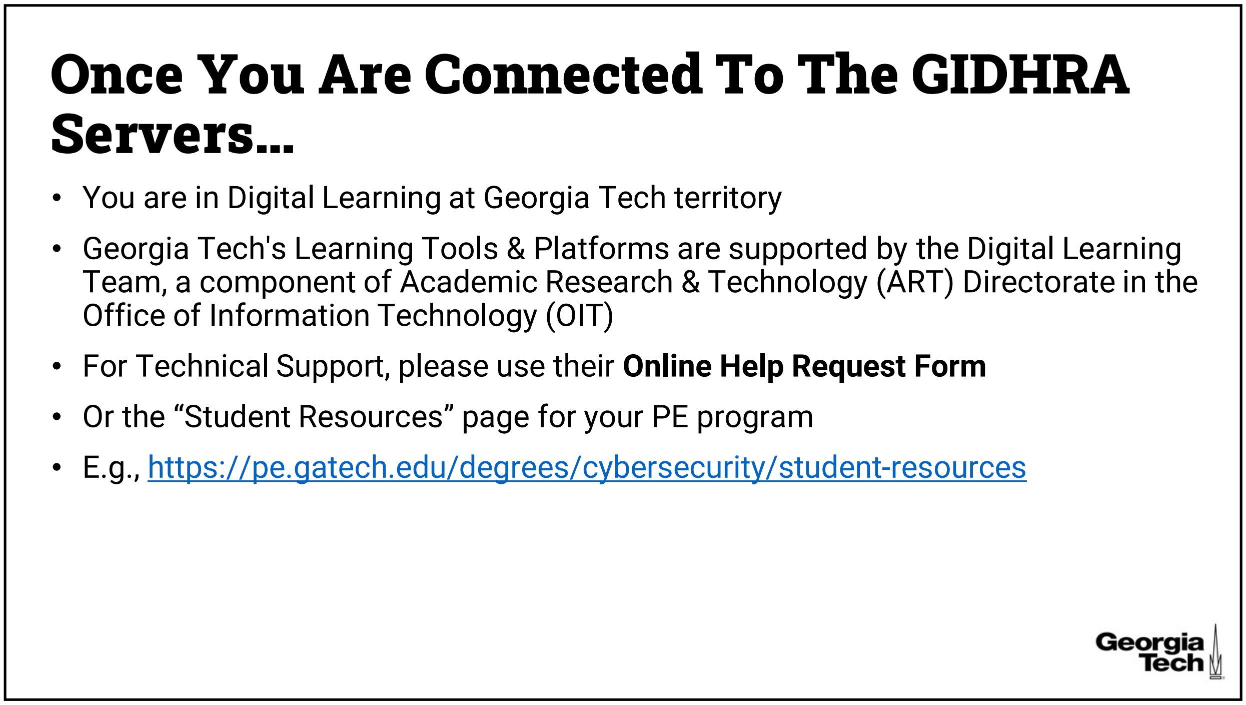


Alternatively: Run GHIDRA On Your Own Machine

- Ghidra is free and open source, so connecting to the AWS servers is not strictly necessary
- Only software requirements are the Java 11 Runtime Environment and Development Kit (JDK)
- No installer – just extract to your machine and run
- More information at https://ghidra-sre.org/InstallationGuide.html#Install

- **WARNING**: If you want to run Ghidra on your machine for this class, you'll be handling actual malware samples on your machine rather than on GT's servers. **This is *not* recommended!**
- **Stay on the GT AWS servers, Stay safe!**

9



Once You Are Connected To The GIDHRA Servers…

- You are in Digital Learning at Georgia Tech territory
- Georgia Tech's Learning Tools & Platforms are supported by the Digital Learning Team, a component of Academic Research & Technology (ART) Directorate in the Office of Information Technology (OIT)
- For Technical Support, please use their **Online Help Request Form**
- Or the "Student Resources" page for your PE program
- E.g., https://pe.gatech.edu/degrees/cybersecurity/student-resources

10

Advanced Topics in Malware Analysis

Welcome to GIDHRA

Brendan Saltaformaggio, PhD

Assistant Professor

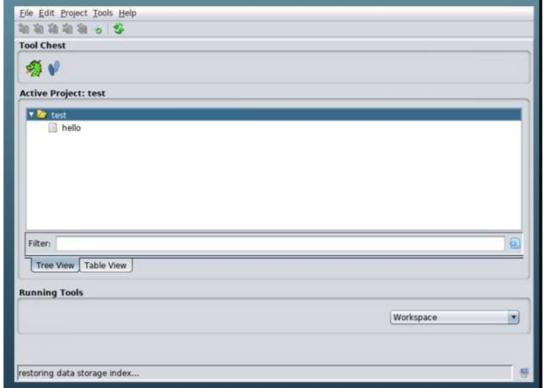
School of Electrical and Computer Engineering

GHIDRA Tips and Tricks

11

Select Your Analysis Target!

- Ghidra starts on a project window
  - Here, you can see previously opened projects or create a new one
- All Ghidra work is done within projects



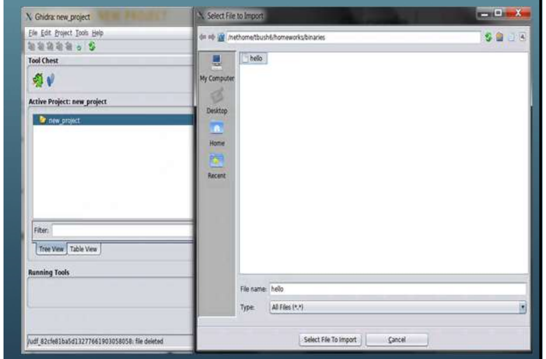
Active Project: test

restoring data storage index...

12

Starting A New Project

- File → New Project
- Ghidra has the option of creating a shared or individual project
- Once your project is created, File → Import File to bring in a binary



/nethome/tbush6/homeworks/binaries

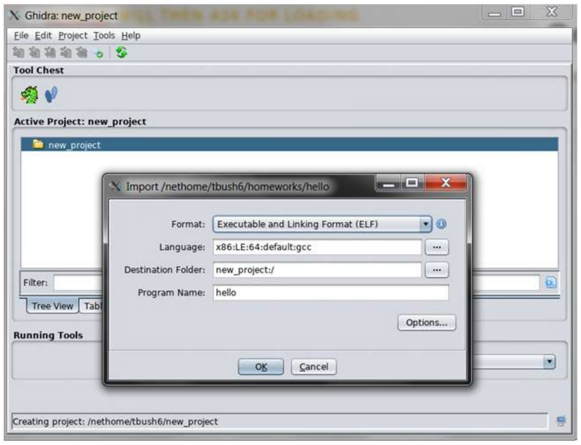
File name: hello

Type: All Files (*.*)

/udf_82cfe81ba5d13277661903058058: file deleted

GHIDRA Will Then Ask For Loading Instructions

- The defaults are almost always correct … unless you are dealing with nasty malware!



Import /nethome/tbush6/homeworks/hello

Format: Executable and Linking Format (ELF)

Language: x86:LE:64:default:gcc

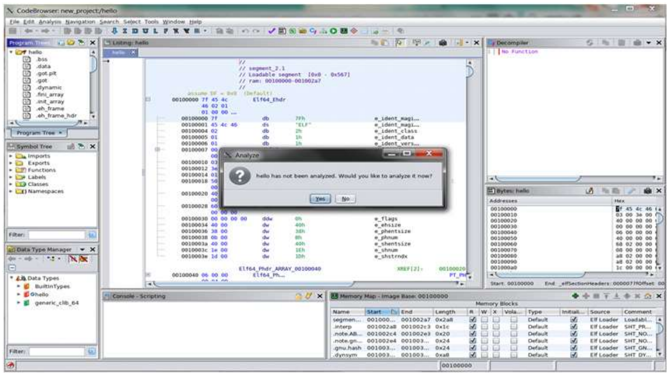
Destination Folder: new_project:/

Program Name: hello

Creating project: /nethome/tbush6/new_project

Viewing Your Binary

- After importing, double-click the binary name to open it up in CodeBrowser
- Ghidra will open in a disassembly view and offer to analyze the binary
  - Again, the defaults are all fine here

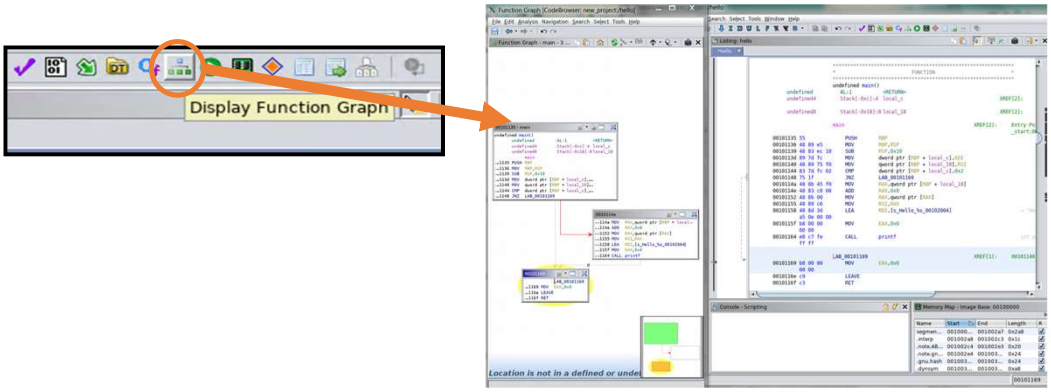


hello has not been analyzed. Would you like to analyze it now?

15

A Different Point of View

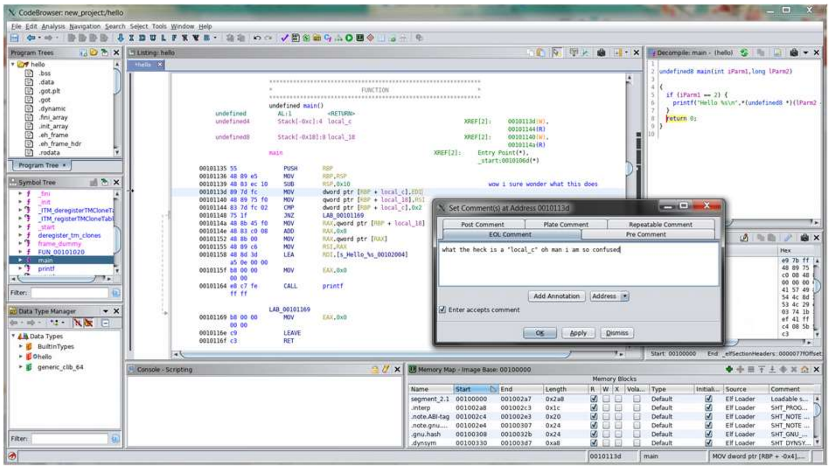
- It can be helpful to switch to "**Function Graph View**" to get a look at the control flow
- Click the graph symbol on the toolbar to open the control flow graph window


Display Function Graph
Function Graph [CodeBrowser: new_project:/hello]
Function Graph - main
Listing: hello
Console - Scripting
Memory Map - Image Base: 00100000

Location is not in a defined or unde

16

Comments: Right-click → Comments → Set eol Comment (or Press “;”)



```
                     **************************************************
                     *                    FUNCTION                    *
                     **************************************************
                     undefined main()
     undefined         AL:1           <RETURN>
     undefined4        Stack[-0xc]:4  local_c                 XREF[2]:     0010113d(W),
                                                                           00101144(R)
     undefined8        Stack[-0x18]:8 local_18                XREF[2]:     00101140(W),
                                                                           0010114a(R)
                     main                                 XREF[2]:     Entry Point(*),
                                                                       _start:0010106d(*)
00101135 55              PUSH       RBP
00101136 48 89 e5        MOV        RBP,RSP
00101139 48 83 ec 10     SUB        RSP,0x10                          wow i sure wonder what this does
0010113d 89 7d fc        MOV        dword ptr [RBP + local_c],EDI
00101140 48 89 75 f0     MOV        qword ptr [RBP + local_18],RSI
00101144 83 7d fc 02     CMP        dword ptr [RBP + local_c],0x2
00101148 75 1f           JNZ        LAB_00101169
0010114a 48 8b 45 f0     MOV        RAX,qword ptr [RBP + local_18]
0010114e 48 83 c0 08     ADD        RAX,0x8
00101152 48 8b 00        MOV        RAX,qword ptr [RAX]
00101155 48 89 c6        MOV        RSI,RAX
00101158 48 8d 3d        LEA        RDI,[s_Hello_%s_00102004]
         a5 0e 00 00
0010115f b8 00 00        MOV        EAX,0x0
         00 00
00101164 e8 c7 fe        CALL       printf
         ff ff

                     LAB_00101169
00101169 b8 00 00        MOV        EAX,0x0
         00 00
0010116e c9              LEAVE
0010116f c3              RET
```

```
undefined8 main(int iParm1,long lParm2)

{
  if (iParm1 == 2) {
    printf("Hello %s\n",*(undefined8 *)(lParm2
  }
  return 0;
}
```

Set Comment(s) at Address 0010113d

Post Comment | Plate Comment | Repeatable Comment | EOL Comment | Pre Comment

```
what the heck is a "local_c" oh man i am so confused
```

Add Annotation | Address

☑ Enter accepts comment

OK | Apply | Dismiss

Start: 00100000 End: _elfSectionHeaders::000077f0ffset

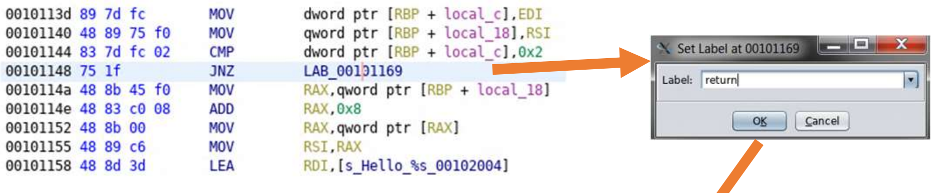
Name	Start	End	Length	R	W	X	Vola...	Type	Initiali...	Source	Comment
segment_2.1	00100000	001002a7	0x2a8	☑	□	□	□	Default	☑	Elf Loader	Loadable s...
.interp	001002a8	001002c3	0x1c	☑	□	□	□	Default	☑	Elf Loader	SHT_PROG...
.note.ABI-tag	001002c4	001002e3	0x20	☑	□	□	□	Default	☑	Elf Loader	SHT_NOTE ...
.note.gnu....	001002e4	00100307	0x24	☑	□	□	□	Default	☑	Elf Loader	SHT_NOTE ...
.gnu.hash	00100308	0010032b	0x24	☑	□	□	□	Default	☑	Elf Loader	SHT_GNU_...
.dynsym	00100330	001003d7	0xa8	☑	□	□	□	Default	☑	Elf Loader	SHT_DYNSY...

0010113d | main | MOV dword ptr [RBP + -0x4],...

Georgia Tech

17

Pro Tip: Rename Labels As You Go!



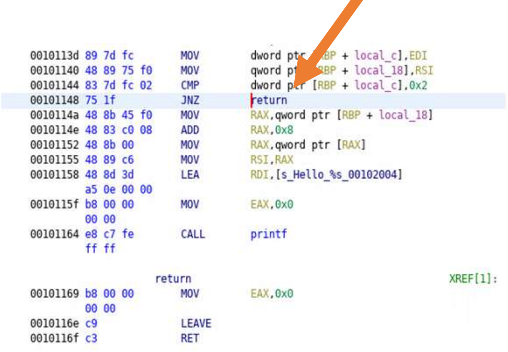
```
0010113d 89 7d fc        MOV        dword ptr [RBP + local_c],EDI
00101140 48 89 75 f0     MOV        qword ptr [RBP + local_18],RSI
00101144 83 7d fc 02     CMP        dword ptr [RBP + local_c],0x2
00101148 75 1f           JNZ        LAB_00101169
0010114a 48 8b 45 f0     MOV        RAX,qword ptr [RBP + local_18]
0010114e 48 83 c0 08     ADD        RAX,0x8
00101152 48 8b 00        MOV        RAX,qword ptr [RAX]
00101155 48 89 c6        MOV        RSI,RAX
00101158 48 8d 3d        LEA        RDI,[s_Hello_%s_00102004]
```

Set Label at 00101169

Label: return

OK | Cancel

1. Click on the element to rename
2. Press the "L" key
3. Enter name
4. Enjoy easier to read assembly!



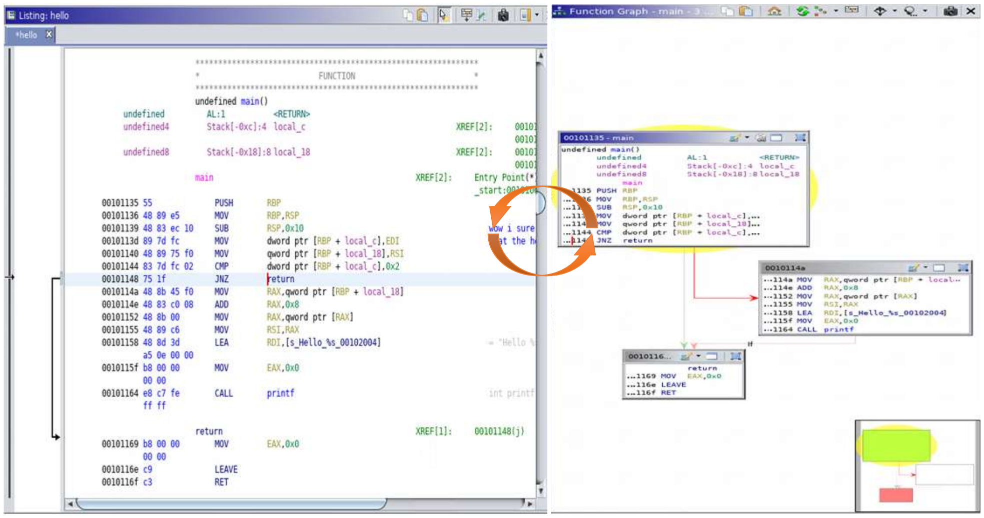
```
0010113d 89 7d fc        MOV        dword ptr [RBP + local_c],EDI
00101140 48 89 75 f0     MOV        qword ptr [RBP + local_18],RSI
00101144 83 7d fc 02     CMP        dword ptr [RBP + local_c],0x2
00101148 75 1f           JNZ        return
0010114a 48 8b 45 f0     MOV        RAX,qword ptr [RBP + local_18]
0010114e 48 83 c0 08     ADD        RAX,0x8
00101152 48 8b 00        MOV        RAX,qword ptr [RAX]
00101155 48 89 c6        MOV        RSI,RAX
00101158 48 8d 3d        LEA        RDI,[s_Hello_%s_00102004]
         a5 0e 00 00
0010115f b8 00 00        MOV        EAX,0x0
         00 00
00101164 e8 c7 fe        CALL       printf
         ff ff

                     return                                   XREF[1]:
00101169 b8 00 00        MOV        EAX,0x0
         00 00
0010116e c9              LEAVE
0010116f c3              RET
```

Georgia Tech

18

Pro Tip #2: Switch Between Views!

- Space bar can be used to switch between views.

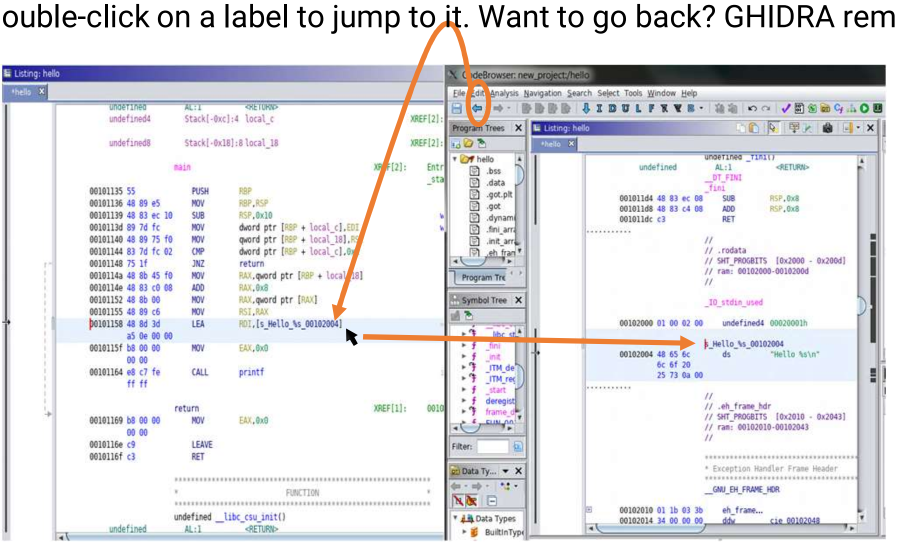


```
                 ****************************************************
                 *                     FUNCTION                     *
                 ****************************************************
                 undefined main()
     undefined         AL:1          <RETURN>
     undefined4        Stack[-0xc]:4  local_c                 XREF[2]:    0010
                                                                          0010
     undefined8        Stack[-0x18]:8 local_18                XREF[2]:    0010
                                                                          0010
                 main                                   XREF[2]:    Entry Point(*
                                                                    _start:00
00101135 55            PUSH       RBP
00101136 48 89 e5      MOV        RBP,RSP
00101139 48 83 ec 10   SUB        RSP,0x10                                  wow i sure
0010113d 89 7d fc      MOV        dword ptr [RBP + local_c],EDI             at the h
00101140 48 89 75 f0   MOV        qword ptr [RBP + local_18],RSI
00101144 83 7d fc 02   CMP        dword ptr [RBP + local_c],0x2
00101148 75 1f         JNZ        return
0010114a 48 8b 45 f0   MOV        RAX,qword ptr [RBP + local_18]
0010114e 48 83 c0 08   ADD        RAX,0x8
00101152 48 8b 00      MOV        RAX,qword ptr [RAX]
00101155 48 89 c6      MOV        RSI,RAX
00101158 48 8d 3d      LEA        RDI,[s_Hello_%s_00102004]                 = "Hello %
         a5 0e 00 00
0010115f b8 00 00      MOV        EAX,0x0
         00 00
00101164 e8 c7 fe      CALL       printf                                    int printf
         ff ff

                 return                                 XREF[1]:    00101148(j)
00101169 b8 00 00      MOV        EAX,0x0
         00 00
0010116e c9            LEAVE
0010116f c3            RET
```

```
undefined main()
     undefined      AL:1             <RETURN>
     undefined4     Stack[-0xc]:4    local_c
     undefined8     Stack[-0x18]:8   local_18
                main
...1135 PUSH  RBP
...1136 MOV   RBP,RSP
...11   SUB   RSP,0x10
...11   MOV   dword ptr [RBP + local_c],...
...114  MOV   qword ptr [RBP + local_18]...
...1144 CMP   dword ptr [RBP + local_c],...
...114  JNZ   return

...114a MOV   RAX,qword ptr [RBP + local...
...114e ADD   RAX,0x8
...1152 MOV   RAX,qword ptr [RAX]
...1155 MOV   RSI,RAX
...1158 LEA   RDI,[s_Hello_%s_00102004]
...115f MOV   EAX,0x0
...1164 CALL  printf

              return
...1169 MOV   EAX,0x0
...116e LEAVE
...116f RET
```

Pro Tip #3: Navigation Buttons!

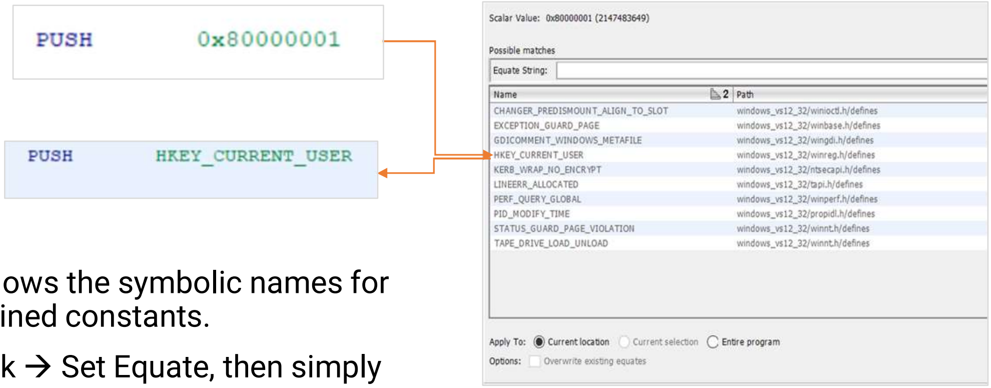
- Double-click on a label to jump to it. Want to go back? GHIDRA remembers!



```
     undefined         AL:1          <RETURN>
     undefined4        Stack[-0xc]:4  local_c                 XREF[2]:
     undefined8        Stack[-0x18]:8 local_18                XREF[2]:
                 main                                   XREF[2]:    Entr
                                                                    _sta
00101135 55            PUSH       RBP
00101136 48 89 e5      MOV        RBP,RSP
00101139 48 83 ec 10   SUB        RSP,0x10
0010113d 89 7d fc      MOV        dword ptr [RBP + local_c],EDI
00101140 48 89 75 f0   MOV        qword ptr [RBP + local_18],RS
00101144 83 7d fc 02   CMP        dword ptr [RBP + local_c],0x
00101148 75 1f         JNZ        return
0010114a 48 8b 45 f0   MOV        RAX,qword ptr [RBP + local_18]
0010114e 48 83 c0 08   ADD        RAX,0x8
00101152 48 8b 00      MOV        RAX,qword ptr [RAX]
00101155 48 89 c6      MOV        RSI,RAX
00101158 48 8d 3d      LEA        RDI,[s_Hello_%s_00102004]
         a5 0e 00 00
0010115f b8 00 00      MOV        EAX,0x0
         00 00
00101164 e8 c7 fe      CALL       printf
         ff ff

                 return                                 XREF[1]:    0010
00101169 b8 00 00      MOV        EAX,0x0
         00 00
0010116e c9            LEAVE
0010116f c3            RET

                 ****************************************************
                 *                     FUNCTION                     *
                 ****************************************************
                 undefined __libc_csu_init()
     undefined         AL:1          <RETURN>
```

```
                 undefined _fini()
     undefined         AL:1          <RETURN>
                 __DT_FINI
                 _fini
001011d4 48 83 ec 08   SUB        RSP,0x8
001011d8 48 83 c4 08   ADD        RSP,0x8
001011dc c3            RET
............
                 //
                 // .rodata
                 // SHT_PROGBITS  [0x2000 - 0x200d]
                 // ram: 00102000-0010200d
                 //

                 _IO_stdin_used

00102000 01 00 02 00   undefined4 00020001h

                 s_Hello_%s_00102004
00102004 48 65 6c      ds         "Hello %s\n"
         6c 6f 20
         25 73 0a 00
............
                 //
                 // .eh_frame_hdr
                 // SHT_PROGBITS  [0x2010 - 0x2043]
                 // ram: 00102010-00102043
                 //

                 ****************************************************
                 * Exception Handler Frame Header
                 ****************************************************
                 __GNU_EH_FRAME_HDR

00102010 01 1b 03 3b   eh_frame...
00102014 34 00 00 00   ddw        cie_00102048
```

Pro Tip #4: Rename Symbolic Constants



```
PUSH        0x80000001
```

```
PUSH        HKEY_CURRENT_USER
```

Scalar Value: 0x80000001 (2147483649)

Possible matches

Equate String:

Name	Path
CHANGER_PREDISMOUNT_ALIGN_TO_SLOT	windows_vs12_32/winioctl.h/defines
EXCEPTION_GUARD_PAGE	windows_vs12_32/winbase.h/defines
GDICOMMENT_WINDOWS_METAFILE	windows_vs12_32/wingdi.h/defines
HKEY_CURRENT_USER	windows_vs12_32/winreg.h/defines
KERB_WRAP_NO_ENCRYPT	windows_vs12_32/ntsecapi.h/defines
LINEERR_ALLOCATED	windows_vs12_32/tapi.h/defines
PERF_QUERY_GLOBAL	windows_vs12_32/winperf.h/defines
PID_MODIFY_TIME	windows_vs12_32/propidl.h/defines
STATUS_GUARD_PAGE_VIOLATION	windows_vs12_32/winnt.h/defines
TAPE_DRIVE_LOAD_UNLOAD	windows_vs12_32/winnt.h/defines

Apply To: ● Current location ○ Current selection ○ Entire program

Options: □ Overwrite existing equates

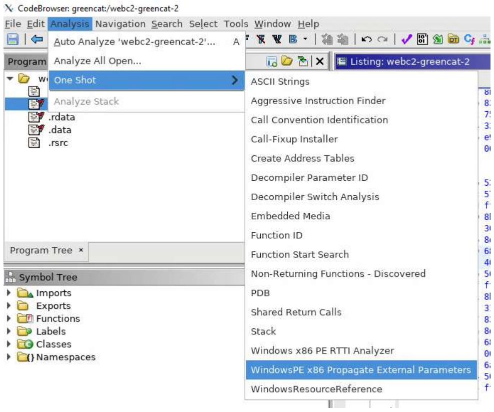
- Ghidra knows the symbolic names for many defined constants.
- Right click → Set Equate, then simply find the constant name you are looking for

Pro Tip #4: Rename Symbolic Constants

- If you can't find the symbol, you can always add it to Ghidra,
  - First, look up the symbol's definition in the header file
  - Then add new Enums/Structures/Constants in the Data Type Window
    - Right click on your project → New → …
    - More details here: https://youtu.be/u15-r5Erfnw

Pro Tip #5: Test Out GHIDRA’s Included Analyses!

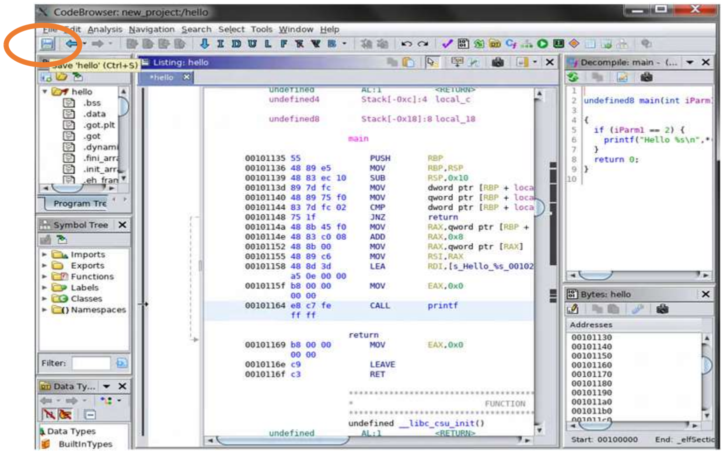
- From the menu bar: **Analysis → One Shot**
- For example: The “Propagate External Parameters” pass will try to auto-comment arguments to library function for you!
- You can even script your own Analysis passes … more on that later ;)


CodeBrowser: greencat:/webc2-greencat-2
File Edit Analysis Navigation Search Select Tools Window Help
Auto Analyze 'webc2-greencat-2'...
Analyze All Open...
One Shot
Analyze Stack
Program
.rdata
.data
.rsrc
Program Tree
Symbol Tree
Imports
Exports
Functions
Labels
Classes
Namespaces
Listing: webc2-greencat-2
ASCII Strings
Aggressive Instruction Finder
Call Convention Identification
Call-Fixup Installer
Create Address Tables
Decompiler Parameter ID
Decompiler Switch Analysis
Embedded Media
Function ID
Function Start Search
Non-Returning Functions - Discovered
PDB
Shared Return Calls
Stack
Windows x86 PE RTTI Analyzer
WindowsPE x86 Propagate External Parameters
WindowsResourceReference

Georgia Tech

Save Often!

- Malware kills computers. Losing all of your analysis kills reverse engineers.



```
                    undefined         AL:1            <RETURN>
                    undefined4        Stack[-0xc]:4  local_c
                    undefined8        Stack[-0x18]:8 local_18
                                main
         00101135 55              PUSH      RBP
         00101136 48 89 e5        MOV       RBP,RSP
         00101139 48 83 ec 10     SUB       RSP,0x10
         0010113d 89 7d fc        MOV       dword ptr [RBP + loca
         00101140 48 89 75 f0     MOV       qword ptr [RBP + loca
         00101144 83 7d fc 02     CMP       dword ptr [RBP + loca
         00101148 75 1f           JNZ       return
         0010114a 48 8b 45 f0     MOV       RAX,qword ptr [RBP +
         0010114e 48 83 c0 08     ADD       RAX,0x8
         00101152 48 8b 00        MOV       RAX,qword ptr [RAX]
         00101155 48 89 c6        MOV       RSI,RAX
         00101158 48 8d 3d        LEA       RDI,[s_Hello_%s_00102
                  a5 0e 00 00
         0010115f b8 00 00        MOV       EAX,0x0
                  00 00
         00101164 e8 c7 fe        CALL      printf
                  ff ff
                                return
         00101169 b8 00 00        MOV       EAX,0x0
                  00 00
         0010116e c9              LEAVE
         0010116f c3              RET
                                *************************************
                                *                           FUNCTION
                                *************************************
                                undefined __libc_csu_init()
                    undefined         AL:1            <RETURN>
```

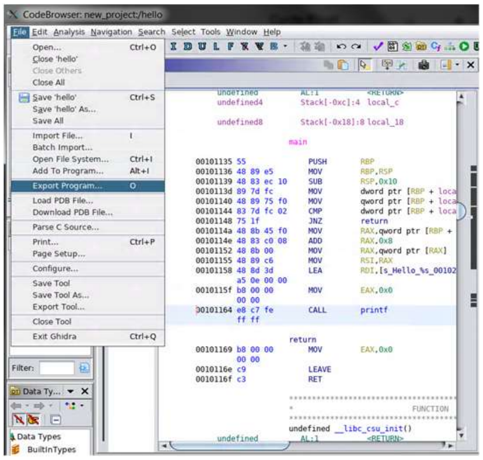
```
undefined8 main(int iParm

{
  if (iParm1 == 2) {
    printf("Hello %s\n",*
  }
  return 0;
}
```

```
00101130
00101140
00101150
00101160
00101170
00101180
00101190
001011a0
001011b0
001011c0
Start: 00100000     End: _elfSectic
```

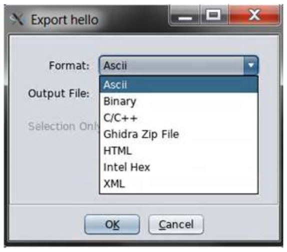
Georgia Tech

Export Your Assembly Listing

- GHIDRA offers several exporting options for your project
  - You will often turn in exported data for reverse engineering assignments



```
       undefined          AL:1           <RETURN>
       undefined4         Stack[-0xc]:4  local_c

       undefined8         Stack[-0x18]:8 local_18

                          main

00101135 55               PUSH      RBP
00101136 48 89 e5         MOV       RBP,RSP
00101139 48 83 ec 10      SUB       RSP,0x10
0010113d 89 7d fc         MOV       dword ptr [RBP + loca
00101140 48 89 75 f0      MOV       qword ptr [RBP + loca
00101144 83 7d fc 02      CMP       dword ptr [RBP + loca
00101148 75 1f            JNZ       return
0010114a 48 8b 45 f0      MOV       RAX,qword ptr [RBP +
0010114e 48 83 c0 08      ADD       RAX,0x8
00101152 48 8b 00         MOV       RAX,qword ptr [RAX]
00101155 48 89 c6         MOV       RSI,RAX
00101158 48 8d 3d         LEA       RDI,[s_Hello_%s_00102
         a5 0e 00 00
0010115f b8 00 00         MOV       EAX,0x0
         00 00
00101164 e8 c7 fe         CALL      printf
         ff ff

                          return
00101169 b8 00 00         MOV       EAX,0x0
         00 00
0010116e c9               LEAVE
0010116f c3               RET

                          ************************************
                          *                          FUNCTION
                          ************************************
                          undefined __libc_csu_init()
       undefined          AL:1           <RETURN>
```



Format: Ascii

Lesson Summary

- We looked at how to connect to GIDHRA
- Pro-tips will help while working on assignments