# ECE 6747 Advanced Topics in Malware Analysis

# MODULE 2 TRANSCRIPTS

## L1 GHIDRA

>> Hello everyone, and welcome back to advanced topics in malware analysis. Starting today, we're gonna start using the Ghidra tool to reverse engineer malware. But first, what is Ghidra? Ghidra is actually a reverse engineering framework that 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 United States government for many years. And thanks to them, it is now free and open source for anyone to use and improve however they can. For this class, we will be using Ghidra on a cloud service to allow everyone access to the Ghidra platform.

The first step to getting access to Ghidra is to install the AWS cloud workspace. To access Ghidra, first download AWS workspace from the URL shown on the screen. You can choose whatever operating system you're using. But a desktop or laptop environment will make reverse engineering a lot easier for you.

Once you've downloaded and installed AWS workspace, you will be prompted to enter a registration code. You should have received your registration code when you enrolled with this class. If you're having any problems with the registration code, feel free to email the TAs or I, and we can get this straightened out for you.

After entering your registration code, you'll be prompted for a username and password. This is your standard Georgia Tech username, the one with the numbers at the end, and the password that you use to access Buzzport. It's the typical Georgia Tech single sign-on that you use for everything else.

After signing in to the AWS workspace, you will be presented with the desktop of a workstation running in the AWS cloud. On there will be an icon for Ghidra. You can double-click that Ghidra launcher icon to get started. But I know what you're thinking, Professor, if Ghidra is open source, can't I just download it and run it on my laptop?

Uh-huh, that's what we're all thinking. You can of course download Ghidra and run it on your own personal computer. You don't have to use the AWS servers provided with this class. Although not strictly necessary, you'll have to install your own Java runtime environments and developer's kit. And there's going to be no installer, you download a package that you then extract to your machine and run.

More information on how to install Ghidra can be found on Ghidra's website, ghidra-sre.org. However, the bigger problem here is that this is a malware analysis class. If you want to run Ghidra on your local machine, you will also be handling malware samples on your local machine, rather than on GT's safe AWS locked down servers.

This is not recommended. Please stay on the Georgia Tech AWS servers, and keep your computer safe. We have carefully vetted the AWS setup to ensure that there is no way for the malware that is part of this class to escape the AWS setup, and get back to anyone's personal computers.

Of that, we are sure. However, what you choose to do with your own computer is up to you. So please stay safe, stay on the GT AWS servers. Once you're connected to the Ghidra servers, you're technically in the land controlled by the digital learning at Georgia Tech. Georgia Tech's learning tools and platforms are supported by the Digital Learning Team.

Which is a component of the Academic Research and Technology or ART Directorate in the Office of Information Technology here at Georgia Tech. They are happy to provide tech support for any problems you're encountering with the AWS setup, or any of the programs installed within there. There's an online help request form that you can find on the student resources page for your PE program.

For example, cybersecurity students can go to pe.gatech.edu/degrees/cybersecurity/student-resources. All of the different PE programs have a similar student resources page that will take you to the technical support help request form. Any technical support you need, please feel free to reach out to them, they will be more than happy to help.

If you have questions specific to the labs themselves and how to perform the reverse engineering, that's a much better question for your TAs. Feel free to reach out to them as well.

## L2 GHIDRA Tips and Tricks

>> Hello, everyone and welcome back to advanced topics in malware analysis. In this lesson, I'm gonna give you some tips and tricks for getting started with your lab in Ghidra. When you start up Ghidra in the AWS environment, you will be prompted to select your analysis target. Ghidra starts on what's called the project window.

Here you can see previously opened projects or create a new one. All work in Ghidra is done within these so called projects. To start a new project, go to File > New Project. Ghidra has the option of creating a shared or individual project, but for this class, you'll be doing individual projects.

Once your project is created, you can go to File > Import File to bring the binary your reverse engineering into Ghidra for analysis. Ghidra will then ask you for loading instructions to help get that binary loaded correctly. Most of these defaults are gonna be correct, because Ghidra has very powerful analysis to try to understand exact formats of binary files when you load them.

Unless you're dealing with very nasty malware that will trip up these analyses and make it harder for Ghidra to understand the binary file. Don't worry, the malware that I've given you for this class will not trip up Ghidra. So you can leave these defaults and click OK for the labs in this class.

You'll then be able to see your binary disassembled, and you can click further through the different sections of the code. After importing your binary, double-click on the binary name to open it in the CodeBrowser window. Ghidra will open in a disassembly view and offer to analyze the binary for you.

Again, the defaults here will be fine, so go ahead and allow Ghidra to do the analysis that it wants. It's only trying to help you. You may want a different point of view than just staring at the straight disassembly. This is where it can be helpful to switch to a function graph view.

This will give you an idea of the control flow through the program. Click on the graph symbol on the toolbar to open the control flow graph window. You're going to want to put comments into the disassembly to keep track of what's going on while you're reverse engineering. To add comments, right-click on a line of the disassembly, go to Comments > Set eol Comment or end of line comment.

Or press the semicolon key when highlighting a line in the disassembly. This will allow you to add some comments to keep track of what's going on during your reverse engineering. Here's a pro tip, rename labels as you go to help you keep track of where they're pointing to.

To do this, click on an element to rename it. Press the L key, and a pop-up box will ask you to enter a new name. You can enter the name you want, and that name will be propagated automatically by Ghidra throughout the entire disassembly. This can make keeping track of labels much easier and gives you a much easier to read disassembly.

Another pro tip, hit the space bar at any time to switch between different views. This can be very helpful when you're reverse engineering some straight line code and want to see the control flow around that code. Here's another pro tip, see a label but don't know where it goes?

Double-click on the label and Ghidra will automatically jump you to the place in the code where that label is defined. Now you want to go back? Don't worry, Ghidra remembers. Click on the back and forward buttons in Ghidra just like a browser to navigate through where you've clicked.

Renaming your symbolic constants will also make your disassembly easier to read. Ghidra knows the names for many constants that are defined in header files. To replace a number or value with its name from a header file, right-click on that value, go to the Set Equate option, and then look through the listing for the corresponding names that you would like to replace it with.

Double-click on that and Ghidra will propagate this knowledge through the binary, so that anytime that you would have seen that nasty constant, you now see the human readable string representing what that constant's name is. If you can't find the symbol in the list, you can always add it to Ghidra.

First, look up the symbol's definition in a header file. Then add new enums, structures, or constants in the data type window. Right-click on your project, New > Data Type. More information can be found at this very helpful YouTube video. Another pro tip, test out Ghidra's included analyses. Ghidra is designed to help you out as much as it can.

From the menu bar, go to Analysis > One Shot. For example, the Propagate External Parameters analysis pass will try to auto-comment the arguments to different library functions for you. You can even script your own analysis passes to add to this analysis drop-down. Don't worry, we'll see a lot more of this in the labs to come.

One warning, please save your work often. Malware may kill computers, but losing all of your reverse engineering will kill you. Please hit that Save button. Allow Ghidra to save all of your progress to a static file periodically before something crashes by accident and you lose all of your data.

Finally, you may want to export an assembly listing. Ghidra offer several exporting options so that you can look at your project outside of Ghidra. You'll often turn in exported data for the reverse engineering assignments in this class. Simply go to File > Export Program and choose ASCII. And that's the end of this lesson.

We looked at how to connect to the Ghidra servers, and I offered some pro tips that will help you while working on your assignments.