Ghidra Introduction

1 Overview

This lab introduces the Ghidra software reverse engineering suite `ghidra-sre.org`. You will use Ghidra to analyze a binary executable to determine some of its properties.

1.1 Background

The student is expect to have some background in low level programming and basic networking.

2 Lab Environment

This lab runs in the Labtainer framework, available at http://my.nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer ghidra
```

A link to this lab manual will be displayed.

3 Network Configuration

The lab includes two computers, one of which is a network server that is only visible via the network, i.e., it has no terminal. The visible computer, name `ghidra` contains the Ghidra tool suite and a copy of the service that is running on the server. You have two terminal windows to this computer. Use the `moreterm.py` command to get more.

The server IP address is 172.25.0.2. Use ping to confirm a connection:

```
ping 172.25.0.2
```

4 Lab Tasks

4.1 Context

A copy of the software service is on the `ghidra` computer in your home directory in a file named `cadet01`[1]. That program is running on the server. Your goals are to communicate with the service; cause it to display an "easter egg"; and then crash the program. You will use Ghidra to analyze the `cadet01` program to achieve these goals.

4.2 Start Ghidra

Ghidra is pre-installed on your computer. Start it by running `./ghidra` from your home directory. After accepting the license terms, you will see two windows. One of those is online help. Use that to familiarize yourself with the tool.

[1] This program originated in DARPA Cyber Grand Challenge as the simple example of a vulnerable network service.

4.2.1 Create a project and import cadet01

Use the Ghidra main window `File / New project` menu item to create a new project. Then use `File / Import file` to import the `cadet01` program. After the `cadet01` program appears int the `Active Project` window, double click on it. When prompted to analyze it, select `Yes`, and accept the default Analyzers.

You should then see a new window titled `CodeBrowser`... On the left edge middle window pane, titled `SymbolTree`, expand the item named `Functions`. View that list, it is the set of functions that Ghidra has identified within the executable. Find the `main` function and select it. Note that the large middle pane now contains the disassembled listing of the main function, and the right-most pane contains a decompiled listing of pseudo-code resulting from the Ghidra analysis.

Explore the program by looking at what functions are called by main, and what functions are called by those functions.

4.3 Find the service's network port

Look through the `cadet01` functions to find the port number that is used when binding to the network socket. Once you have found the port number, use it to communicate with the service. For example, use the `netcat` program:

```
echo "Hi" | nc 172.25.0.2 <port number>
```

where `port number` is what you found. If you are successful you should see a reply from the service.

4.4 Find the Easter egg

A particular input to the service will cause it to display an Easter egg. Use Ghidra to identify the required input. Use that knowldege to send the service a string that will cause the Easter egg to display.

4.5 Crash the service

Review how the cadet01 service handles input read from the network. Find the buffer variable into which the service receives network data and rename it to "buffer" – do this in functions that reference the variable. Make note of the buffer size. Also identify the variable that constrains the quantity of bytes that will be read into the buffer. Give it label of `byte_count`.

Finally, based on what you found above, find an input that will cause the service to crash. You will know it has crashed if `netcat` displays a "Connection reset by peer" message.

5 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab
```

When you stop the lab, the system will display a path to the zipped lab results on your Linux system. Provide that file to your instructor, e.g., via the Sakai site.