COMP3236 v1: February 12, 2021

Laboratory 0: Using Vagrant, C and Linux

Week 2

Welcome to Laboratory 0, in this lab you will learn how to use basic tools that we will be using in this module. For the sake of completeness, we added how to connect to Discord, just incase you viewed this document after the session. Then we will explain how to install the experimental setup for you to do your assignments. Lastly, we set a couple of exercises to prepare you for the next lab.

Submission Instructions

There is no submission for this Lab.

Joining Discord

As we are teaching fully online this year, many cyber security modules are being run on Discord instead of Microsoft Teams, which allows demonstrators and lecturers a number of tools to engage with students through text and voice chat.

What is Discord? it's like teams except it has superior audio and video quality, also has a lot of features that make your learning experience much better. Want to know more? read this article this article.

In order to join the laboratory support discord, you will need to sign in and associate a discord account with your university account.

1. First, you will need to join the University VPN. To do this, go to the following link to download and install the client for your OS here. Make sure you are connected via VPN before you continue.
2. Make a Discord account at https://discord.com/, or make sure you are signed into your existing account either on the native app or in the web browser.
3. Login to https://discord.ecs.soton.ac.uk/ with your university login.
4. Press the lab link to join the virtual lab, and accept the discord invite.

You should now be able to see the COMP6236 Secure Software category in the sidebar. If you can't, post in #general and say you are from COMP6236 and can't see the category for it.

Experimental Setup

For the laboratory sessions in this module we will be making use of tools.

Vagrant is a tool which allows virtual machines with specific configurations to be spun up from configuration files. In order to use it, you will need to download some software. This tool will ensure that you have a clean experimental setup every time. Also, help you get up and ready for the lab in no time, all the tools and data will be inside the vagrant script. All you have to do is run it and enjoy.

VirtualBox is a tool that allows you to run virtual machines on your computer and its a pre-requisite for Vagrant.

Git is a tool that implements source control. You will be using this to download the new lab every week.

Please download and install them in this order:

1. Download Virtualbox from https://www.virtualbox.org/wiki/downloads.
2. Download Vagrant from https://www.vagrantup.com/downloads.
3. Download git (Windows) from https://git-scm.com/downloads. If you're running mac or linux, you should already have this installed.
4. When installing git, make sure you tick the option to "Add to path", which will allow you to run git from the terminal.

Once both of these packages have been installed, you should be able to open up a terminal on your computer (cmd on Windows, Terminal on Mac/Linux) and type vagrant to check it is installed and working.

Listing 1: Verify that vagrant is installed

```
#!/bin/bash
vagrant -v
```

If you get an output showing the vagrant version number, you are good to move on to the next step.

Your First Vagrant Machine

Vagrant works by reading configuration files in your current directory and using them to create VMs.

Hence, in order to spin up the VM, you may need to use the cd command in your terminal to change-directory into the correct folder. Change directory into a working folder from which you wish to complete the labs. **You will need to make it if it doesn't exist, and then cd into it for example:**

```
#!/bin/bash
cd Documents/comp6236-labs
```

Download the lab0 VM image from the UoS Git server by typing the following:

```
#!/bin/bash
git clone https://git.soton.ac.uk/comp6236/lab0
```

Change into the lab0 folder so we can use vagrant:

```
#!/bin/bash
cd lab0
```

You should now be able to run the following command to

```
#!/bin/bash
vagrant up
```

If you have the virtualbox window open, you will notice that a new VM appears, and vagrant begins to build it. It should take a couple of minutes.

When the build process is complete, you can connect to a shell on your newly created VM using the following command:

```
#!/bin/bash
vagrant ssh
```

Challanges

Before we introduce any assignment, we will present you with several challenges, that escalate in difficulty to help you off your feet, and break up what you need to learn to smaller chunks.

Challanges 1 - Compile a C Program

Inside your VM, you can use a number of commands to make, move, and modify files. If this is your first time using a command line, check out some online tutorials about how to interface with the command line, for example https://ubuntu.com/tutorials/command-line-for-beginners.

Inside lab0/task1, there is a file called **helloworld.c**.

Using online resources, find out how to compile the program using gcc and run it. What does the program output when it is run?

Challanges 2 - Extracting Strings from a Program

Inside lab0/task2 is another identical C program, but it has been compiled with a secret string which is never output. The source code is available in **task2-redacted.c**.

```
#!/bin/bash
$ ./task2 <number>
Exiting...
```

Can you find a tool available in linux for extracting strings from files?

What is the secret string that would be printed if the program was compiled to output it?

Challanges 3 - Decompiling a C Program

Inside lab0/task3, you will find a number-guessing C program. As the user you have to guess a number, and if it's correct, you get to see a secret string.

The source code is available in **task3-redacted.c**.

```
#!/bin/bash
$ ./task3
Usage: task3 <guess>

    guess: what number i'm thinking of

$ ./task3 10
That's not the number i was thinking.
```

Using online resources, explore how to use **objdump** to decompile the binary.

What number do you need to input to make the program output the secret string?

You may find this short guide useful for understanding x86 assembly pneumonics.

What is the output of the program when you get the number correct?

Golden Challange - Advance Decompiling in C Program

Inside lab0/task4 is another C program which is very similar to task 3. The source code is available in **task4-redacted.c**. The only difference is that this program doesn't use a direct comparison to check the number, and instead uses a series of mathematical functions in a certain order.

Can you reverse engineer the **check** function using **objdump** to determine what number to input for it to output the secret string?

What string is output when you get the number correct?