Liam McNamara
July 28, 2020
CSE 4471 Information Security
Project Report

Physical security is one of the most fundamental security threats facing information systems, property, and people. The best way to mitigate this threat is by monitoring the surroundings of the entity you wish to keep secure. Though us humans can provide a good means of keeping an eye on things, there are many times when we are unable to monitor at a level that can provide us with the reassurance we desire. The security camera provides a solution to when human observation falls short and we wish to preserve some level of physical security. In my project, I have created a security camera using a raspberry pi along with some additional hardware.

A raspberry pi is a single board computer that provides most of the same components as a desktop personal computer, in a much smaller form. This was my first time using a raspberry pi, so I dealt with a bit of a learning curve. By the end of the project I felt extremely capable and confident with my knowledge of the pi as well as its operating system and capabilities. During the implementation of the security camera, I utilized the pi's WiFi capabilities, operating system, remote access, as well as connectors for a camera module, USB, and HDMI display. Once the initial set up was complete, the only requirements for keeping the raspberry pi operational were the camera module, a stable internet connection, and a power supply.

I have detailed the process of my implementation of the camera in a step-by-step instruction guide observable in a GitHub repository, which I have linked at the bottom of the report. This repository also contains hardware requirements, links to my sources, and a video demo of the camera in action.

Without getting into too much detail I will summarize how the security camera was made operational. I first installed NOOBS onto a micro SD card, which I placed in the raspberry pi. I connected a keyboard, mouse, and display to the raspberry pi and used these tools to install Raspbian; the most common operating system used on pi's. With Raspbian installed; along with the keyboard, mouse, and display connected, the pi functioned much like a regular personal computer. I then attached the camera module and positioned it at an appropriate filming angle [Figures 1 and 2].

I found one of Raspbian's most useful features to be the "Terminal" application. Terminal on a raspberry pi is a Linux command prompt very similar to that of Mac OS, or "Console" as I believe it is called on windows machines. I used terminal to install "motion" which is a software that allowed me to create the camera server. I set the camera driver to start up automatically when the pi turns on, and I made many adjustments to the motion configuration file in order to set up a server. Using the IP address of the pi on my local network, I was able to access the stream on any device also connected to the WiFi network [Figure 3].

For my own personal use of the security camera, I chose to position it facing my home driveway, from inside of my kitchen. [Figures 4 and 5] From this angle I was able to observe any

cars or people coming and going from the driveway, thus mitigating the physical security threat facing my place of residence, and everyone inside.

Though this brief summary of the camera implementation may seem straight forward, it did not come without its challenges. Throughout the process of setting up the camera I ran into many problems that required more attention and problem solving to overcome. The following includes some of the more significant challenges I faced.

Determining the hardware required for the project proved to be an ongoing process. As previously mentioned, this was my first time working with a raspberry pi, and so I was forced to consult many resources to determine what I actually needed to purchase for the build. The raspberry pi, camera module, and micro SD card were intuitive hardware requirements I purchased initially. It wasn't until I attempted to begin working that I realized the pi did not come with a power supply. This was not something I expected to have to purchase in addition to the device itself, because the vast majority of electronics are sold with the power cord included. In addition to this, I quickly realized I needed hardware to set up the raspberry pi and use the terminal application on its operating system. This required a purchase of a USB keyboard and mouse, along with an HDMI-to-micro HDMI cable. My inattention to the port specifications on the pi resulted in my purchase of an incompatible HDMI-to-mini HDMI cable. This of course, did not fit the display ports on the pi which are made for micro HDMI only, and forced me to exchange this cable for a compatible one.

The WiFi connectivity range proved to be challenging. My desk that acts as my personal workstation is on the edge of the connectivity range that my internet router provides. Though this does not have an effect on my laptop, it proved to be an issue when attempting to connect the pi to the internet which is a critical step. To resolve this issue, I moved my internet router 10 feet closer to my workstation. This increased the strength of the Wi-Fi signal enough to enable the pi's ability to connect.

Inconsistencies in the pi's IP address also posed a challenge. I was only able to record the IPv6 translation of the raspberry pi's IP address using a traditional web browser look up. However, I quickly realized that in order to access the stream online I needed the IPv4 version of the address. To resolve this, I went into my router settings for a listing of all the devices connected and found the IPv4 of the Pi.

During the design stages of my project, I thought a useful feature would be the ability to access the security camera stream from any device outside of the local network. After the pi camera was set up to function locally on my network, I began the steps to enable access from devices outside the network. This process involved creating a dynamic DNS and configuring my home router to deal with incoming requests from outside untrusted networks. I set up a dynamic DNS using a free "no-ip" service, and then began to configure the router. To handle outside requests, I had to enable port forwarding to direct requests from outside networks to the pi. This required administrative log-in to my internet provider account and following their steps to enable port forwarding. I quickly realized an issue specific to my router and internet provider, which is Xfinity Comcast. This provider enforces strict security policies that are able to detect and block suspicious sites and requests. In fact, I was surprised to see that my provider reported the blocking of 12 malicious requests of the form of spyware and malware in the past

3 months on my personal devices alone! Xfinity Comcast conveniently blocks these requests, so the malicious request reports were just something for me to be aware of, and not something that posed a continuous threat. Enabling port forwarding on the Xfinity router shuts off these advanced security features that were a proven means of security to the device on my network. I thought it would be in my best interest to keep the features in place and restrict the stream access to devices that were on the same network.

The final product of my project is, again, a raspberry pi configured to function as security camera with a live feed going to a web address. This stream is accessible from any device connected to the network. For the past few days I have kept the camera in continuous operation and found it useful for viewing the comings and goings of people through my driveway and front door, to which the camera is positioned to point at. I distributed the web address to my family members living with me and they too have been making use of the stream. This project introduced me to raspberry pi software and hardware, as well as web streaming. I was able to apply my knowledge of IP addresses, terminal commands, configuration files, and general networking to bring my initial project design plans to fruition.

GitHub: https://github.com/LiamAMcNamara/RasPi_Security

References:

Michael Parreno. “Spy Your Pet with a Raspberry Pi Camera Server.” *Hacker Noon*, 14 July 2017, hackernoon.com/spy-your-pet-with-a-raspberry-pi-camera-server-e71bb74f79ea.

Engineers, Being, and Instructables. “How to Make Raspberry Pi Webcam Server and Stream Live Video || Motion + Webcam + Raspberry Pi.” *Instructables*, Autodesk, Inc, 21 Sept. 2017, instructables.com/id/How-to-Make-Raspberry-Pi-Webcam-Server-and-Stream-/.

Parreno, Michel. “How to Access Your Raspberry Pi Camera from Anywhere.” *Medium*, HackerNoon.com, 28 Dec. 2018, medium.com/hackernoon/how-to-access-your-raspberry-pi-camera-from-anywhere-544ab9e5bacc.

Sherwood, Chris. [Crosstalk Solutions] (Dec. 30, 2019) *Raspberry Pi 4 Getting Started* [Video] Youtube. youtube.com/watch?v=BpJCAafw2qE

Figures:

Figure 1

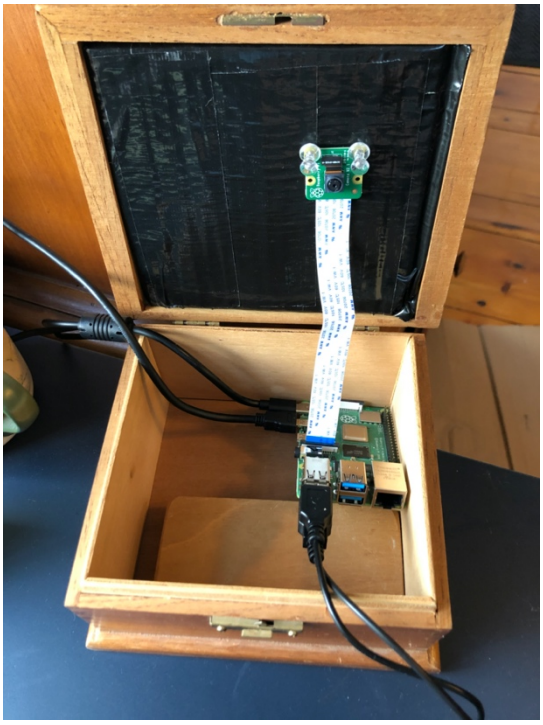

Figure 2

Figure 3

Figure 4

Figure 5