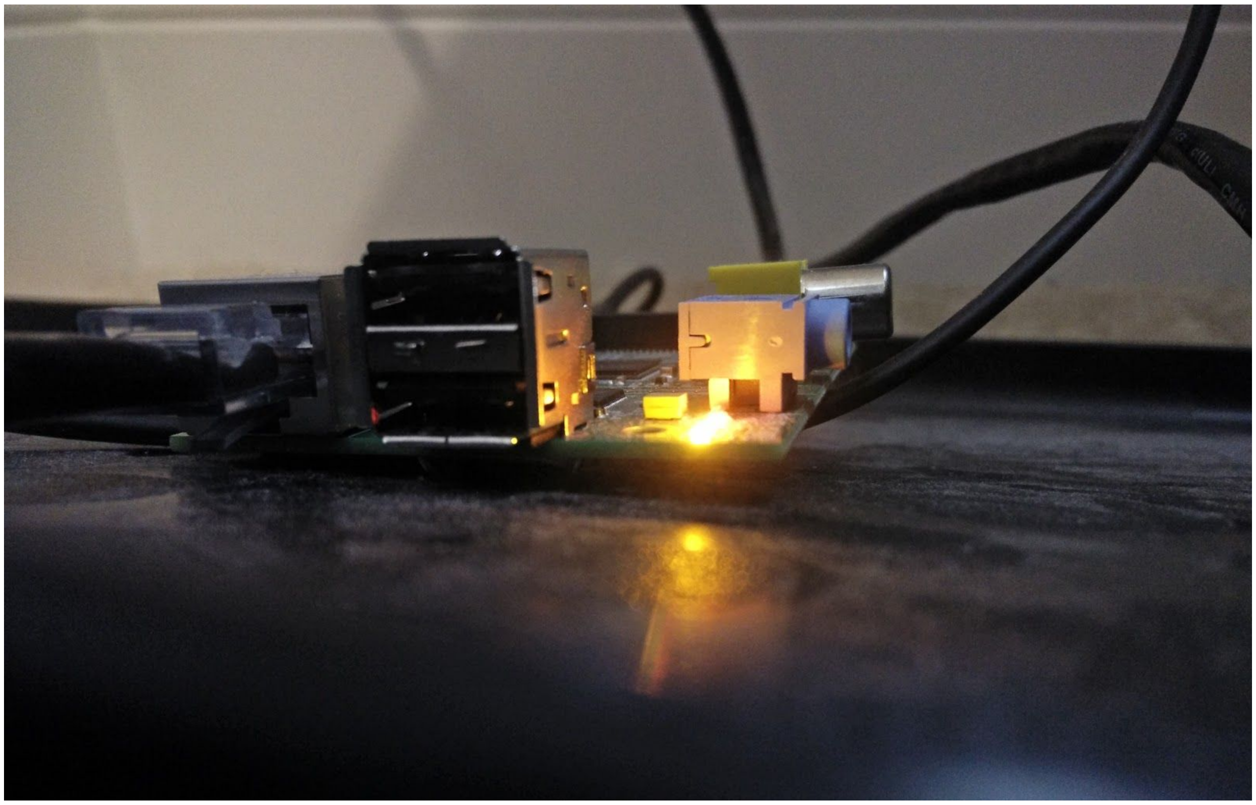
Garage Door Opener

a garage door that you can text to open



The concept

One day, I was walking home from the bus stop one day in the frigid Michigan air, I realized that I had forgotten to bring a garage door opener with me. I was stuck outside my home in the cold for several hours as I waited for my parents to arrive from work. In order to prevent this incident from occurring again, I decided to use my interest in technology to my benefit by trying to create a system that opened my garage with my phone.

Problem Statement

In order for me to open my garage with my phone, many categories must be satisfied. First and foremost, I needed to somehow make my garage door internet-facing, so that I could access it from anywhere. Secondly, I wanted it to be low-cost. I didn't want to spend monthly fees of $20 to services like Azure or Amazon EC2 just to run a web server to direct calls to my garage. With these constraints in hand, I began to brainstorm ideas.

Opening Mechanism

The entire basis of this project starts with the garage door opener. In order to open the garage, I needed to find a way to open the garage with a microcontroller like an arduino. After some quick searching online, I found that many people who made similar projects connected their microcontrollers to the switch on the side of their garage. I didn't want to do this for many reasons:

1. It's messy: If I mess around with the switch on the side of the garage and attempt to connect a microcontroller to it, there would be wires going all over the place at the front of the house, and it would look ugly to anybody walking through.
2. There's no connection: I wanted to make my garage door opener internet facing, and there was no ethernet ports nearby to connect anything to the internet. I could have used WiFi modules, but they are expensive, so that was out of the question.

And so, I decided to do something that I didn't see on the internet. I took apart a garage door opener remote, and connected a microcontroller to it. Instead of messing around with the switch, by working with a garage door remote, I was able to put the microcontroller anywhere within my house since the garage door remote had enough range to reach the garage.

The Server

The server is a node.js instance running on a raspberry pi, with port forwarding enabled. The microcontroller then constantly polls the server on a specific port, detecting for a change in state. If a change is detected, it opens the garage. Essentially, port forwarding allows me to run a server on my home network that can get broadcasted on my WiFi router’s ip address, allowing me to communicate with the rest of the internet. I chose to run the server this way because a raspberry pi is a single, fixed cost and after buying it the only thing I need to pay for monthly is the cost of electricity to run the pi as opposed to running the same node.js instance on an Amazon EC2 instance and paying $20 a month. At this point, I was able to send an HTTP request through the web browser of my phone and open the garage.

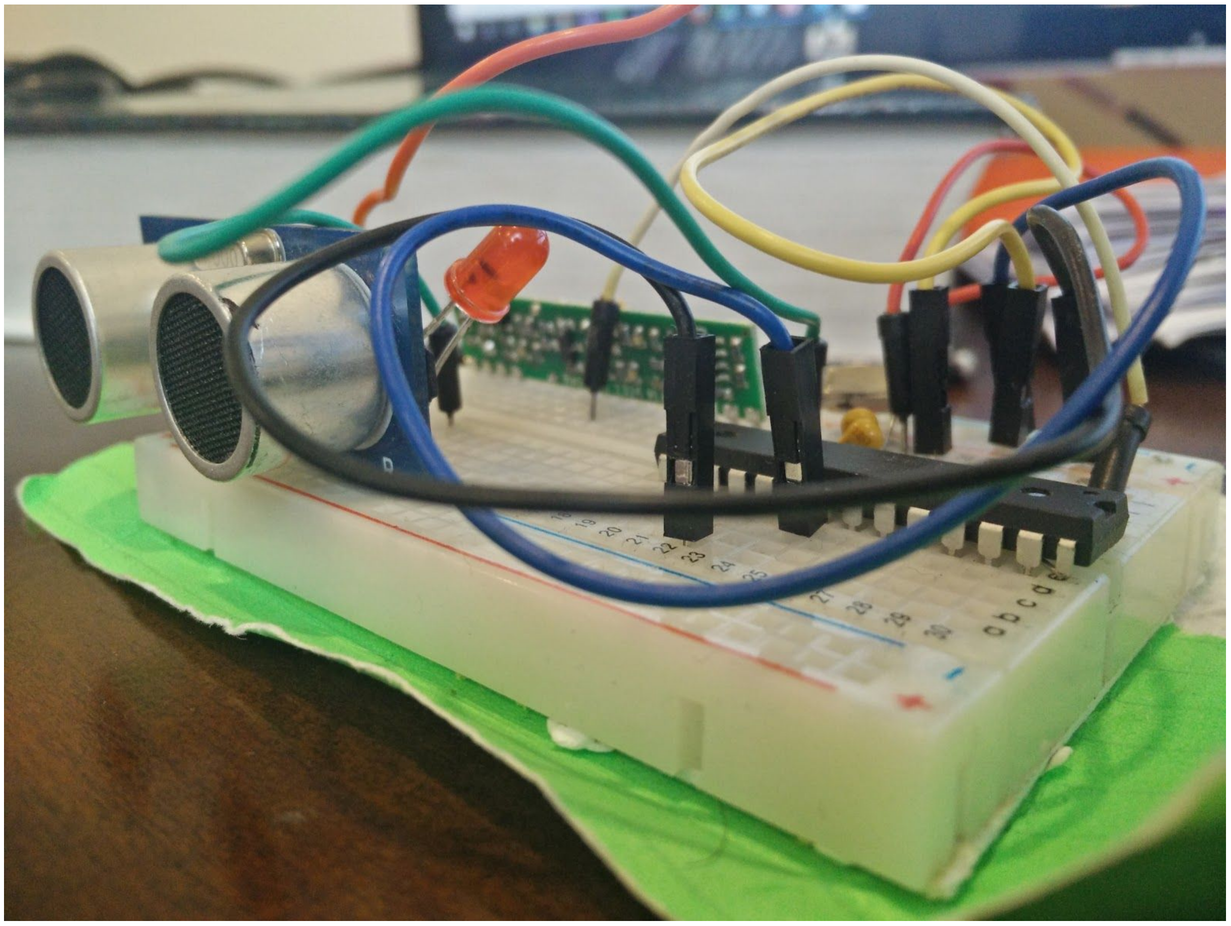
Running a home server comes with its risks, however. Since instructions to open the garage are accessible on the server, hackers have the ability to get access into my server and break into the house. Because of this, I decided that I needed to add another layer of protection.

Twilio

I added Twilio, a text-messaging API, in order to keep my garage door secure. By adding Twilio, I was able to remove access to the garage via a web browser, and only create a route on the server that communicated with Twilio. By adding Twilio, I added a pretty cool functionality to the garage: It was textable! I could send a text message to a number, and it would change the state on the server. When the server changed state, the microcontroller detected this change and opened the garage.

Status Detection

One of the big issues with the garage door opener at this point was the fact that it could not detect the status of the garage. Although I could open and close the garage remotely, there was no way for me to know if the garage was open or closed. To fix this, I built a portable device using an atmega328p microcontroller and an ultrasonic sensor. I put the sensor on the front of my garage. This sensor uses Radio Frequency Links to send data to the main arduino, and then the main arduino posts that data onto the raspberry pi server. Here is a picture of the status-detection model after I took it off of the garage:



Mobile Application

I then made a mobile application that made all of this easier for the user to use. All the mobile app does is 2 buttons. If the top button is pressed, the phone sends a “1” to the garage door opener and if the bottom button is pressed, the phone sends a “2” to the garage door opener, signifying what door to open. The app also does an HTTP GET request to the raspberry pi server to determine the status of the garage doors, and then visually shows if a garage door is open or not through the app, which you can see in the video.