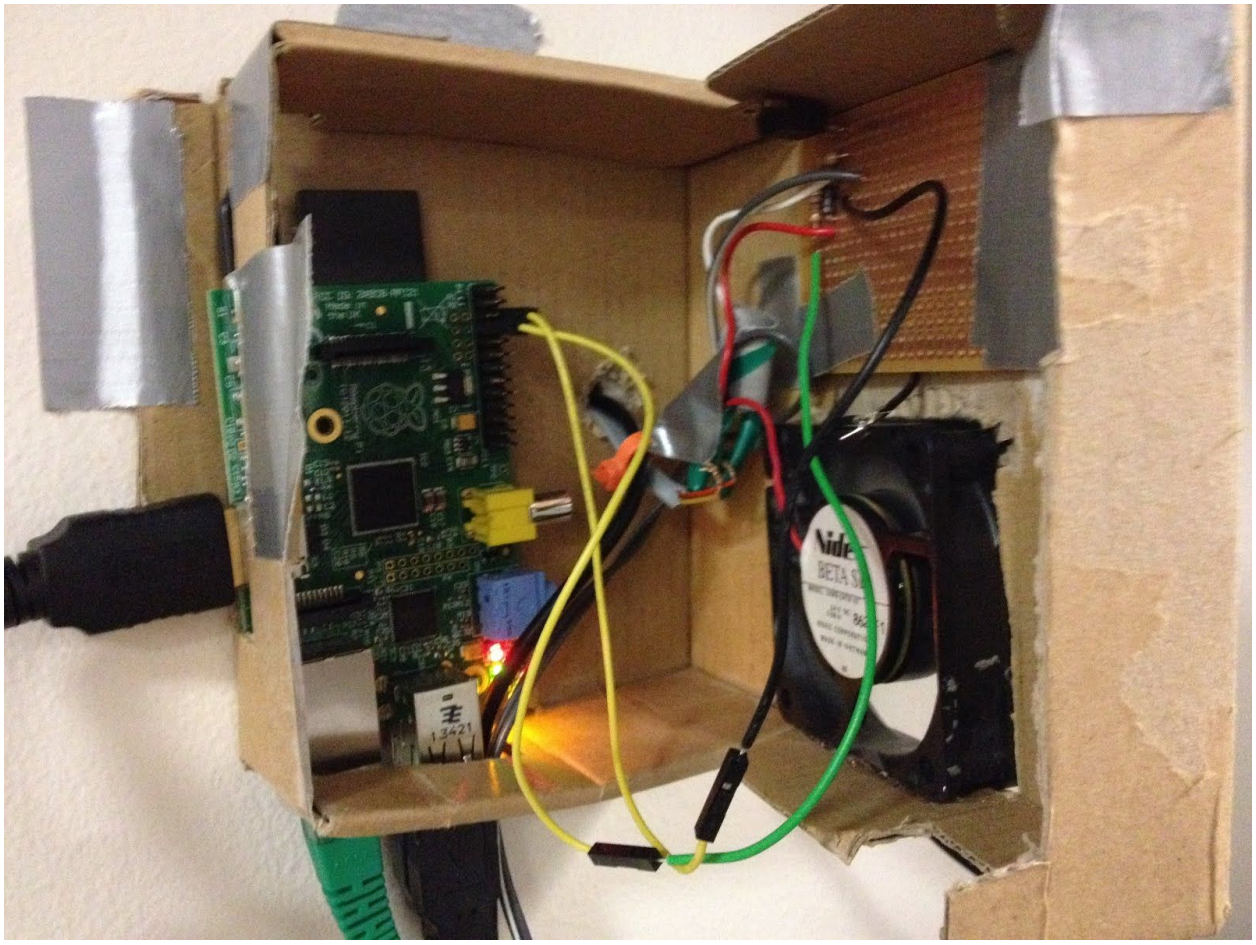
Hardware System:
Key Elements

- Raspberry Pi running Raspbian on 8 GB SD Card
- Transistor board
- Card Swipe model ZCS100 from Zhenzhen ZCS Technology Co.
- Door strike of unidentified brand



Card Swipe:

- Card swipe uses a 3 track MagStripe to read data from swiped cards
- The card reader is currently set up to read 21 characters of any swiped card. Testing has found no card shorter than this length (21 characters is the length of the UMD ID card data)
- The Card swipe has 2 lights, both should be on in general, green light blinks during data transmission. Transmission is followed by a beep from the card swipe

Door Strike:

- The door strike uses a solenoid to pull back a pin and allow free access to door
- When the solenoid is successfully triggered by an access permitted card there is an audible click as the pin is moved
- The door and door strike are somewhat at odds, leading to issues with opening the door on successful reads. This is due to the installation of the door strike and the quality of the door rather than any problems with the operation of the door strike system

Raspberry Pi

- The RPi accepts input from the card swipe via USB, treating it as a USB keyboard.
- The RPi uses the GPIO ports to control the door strike itself, controlling the GPIO ports via the Python RPi.GPIO library
- The RPi casing includes an always on fan for cooling purposes, and the casing includes holes for 2 USB, HDMI, Ethernet, and micro usb power inputs

Software

The RPi is currently set up to run entirely remotely. The watch dog program doorWatchDog.py is in the RPi's boot sequence. The watch dog runs the main door control software in a try except sequence. If the door controller (doorIB.py) crashes, the watch dog automatically sends an email to administrators and reboots the pi. If the program terminates without error (which should really never happen) the boot sequence automatically reboots the pi, resetting the door control program. The door control program switches the terminal to character input mode, and takes characters in the input stream sys.stdin using the python termios and sys libraries. The program reads the first 21 characters from the standard in buffer and then throws out the rest (through unknown means). This means that once registered, any card may be used for door access. Access control is based on a MySQL database doorMaster hosted locally on the pi. This is

accessed using the python library MySQLdb. On a good swipe (card id is in the accepted cards database) GPIO port 7 is turned on, triggering the solenoid for 10 seconds.

The MySQL database is called doorMaster, and is accessed using user account master, with password “password”. (The root account for MySQL installation on the RPi is “root”, “password”) This database contains two tables, acceptedCards and log. acceotedCards has 2 columns, card (which contains the id info for the card, and name, which is the users name. The log table has three columns, accessRequest (the id for the card attempting access), action (either Invalid ID - Door Not Opened, or Door opened for [user name]), and timestamp which is just a string logging the time and date of attempted access. Early data entries in the log are incomplete due to the initial partial implementation.

To add a new card to the accepted cards table, one must swipe said card entering it into the log as an invalid id. Then the program addLastCard.py is run through a terminal window or through the python shell. This program will request a name to add to the database, and will then add the last ID in the log to the database under the given name. This means that someone in the room must be available to add new IDs to the database on the RPi.

Methods of access to the RPi:

Currently the RPi automatically starts all necessary operations when it boots up. This may be accomplished by simply plugging in the RPi. Once booted, the RPi itself is locked into door control mode, and cannot be used for anything else locally. Anyone wishing to modify the RPi door code may do so by logging on to the RPi using ssh, (the RPi may be accesssed on the hackerspace network at 198.162.1.111) To access the RPi locally, (please don’t do this unless it is absolutely vital for some reason.) one must log in via ssh, and then modify /etc/rc.local, removing the line sudo python doorWatchDog.py and sudo reboot at the bottom of the file. This will restore the RPi’s boot sequence to standard and allow local use of the RPi. If this is done for some reason please be sure to re add the lines and reboot the RPi to test functionality when local use is complete.

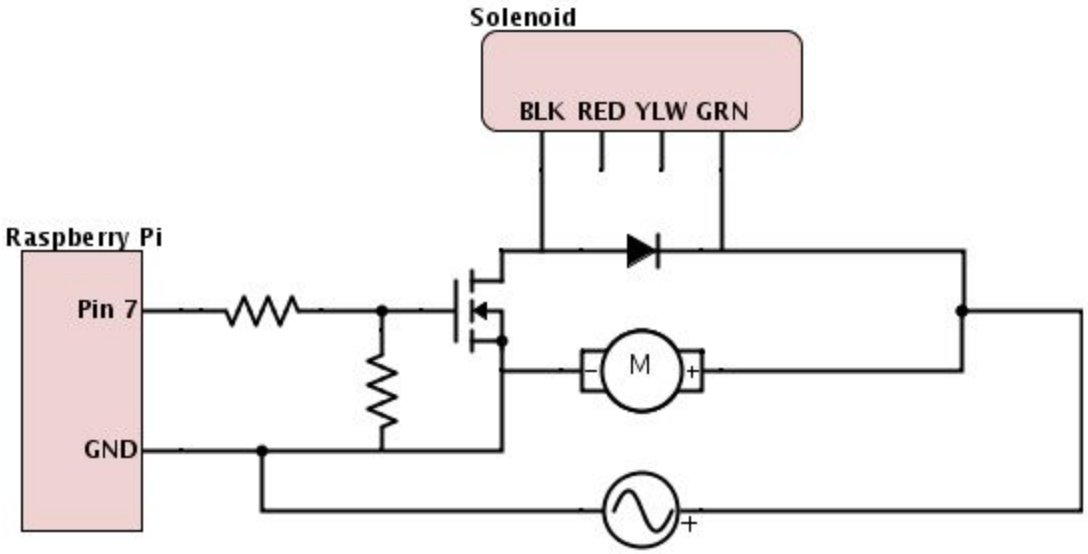
User Interface

The external always on user interface for the door strike is provided by a Nexus Tablet running our mobile website in display mode. This website will provide the user visual queus as to the status of their swipe. If the swipe was accepted the screen flashes green, if not the screen flashes red. On a bad swipe the user is given a chance to request registration. In this case they are then directed to enter first and last name and email address. This will then add an entry to the access requests table.

Admin interface

The Admin interface contains a secure login page, which directs to a home page. This page contains some basic information about the system, and will eventually contain links to all necessary documentation and files needed to remake the system in the event of catastrophic failure. There is another page which displays all log entries, a page which contains accepted cards table, and another for access requests table. Both the accepted cards table and access request tables will include check boxes along with all entries. A button near the top of the table (only for these 2 tables, not log) will allow the selected entries to be removed in the case of the accepted cards table, and added to the accepted cards table in the case of the access requests table. This gives administrators an easy method of adding new cards to the system and removing outdated cards, as well as examining door access.

Hardware



Solenoid:

- Four pins (use only 2 pins)
    - Black (BLK) - Ground
    - Green (GRN) - Power
    - Red (RED) - unknown
    - Yellow (YLW) - unkown

Raspberry PI:

- Two pins
    - Pin 7 - Pulse Width Modulation of the power
    - GRN - Common ground

Fan:

- Nidec 24V .09A

Power Supply:

- 12 V common power supply

Circuit:

- 2 Resistors
    - 1k Ohm
    - 10k Ohm
- 1 Diode
- 1 Mosfet IRLB8721 http://www.adafruit.com/datasheets/irlb8721pbf.pdf

Set Up

Introduction:

To control access to our hackerspace at the University of Maryland Human Computer Interaction Laboratory and log movement in and out of the room we built a magswipe card reader verified magnetic door strike system with an android tablet interface. The system uses a raspberry pi as both the host of the web interface allowing easy modification of access lists as well as for reading and comparing card information. This instructable will include our method for implementing the full system including both an admin page for changing the whitelist of user cards and viewing the log of entries, as well as a slideshow interface to be displayed on a tablet.

Step 1: Assemble Parts (Physical Parts)

- Circuit
  - 1 Common 12 V Power Supply
  - 1 Solenoid
  - 1 Nidec 24V .09A Fan
  - 1 Raspberry PI
  - 2 Resistors
    - 1k Ohm
    - 10k Ohm
  - 1 Diode
  - 1 Mosfet IRLB8721
  - 1 Card Reader ZCS100 from Zhenzhen ZCS Technology Co.
  - Wires/Solder/Perf board/Cardboard/Nails/Glue
- Front End
  - 1 Android Nexus 7 1st Gen tablet
  - 1 Android Power cable (Tablet)
- Back End
  - 1 Monitor
  - 1 Keyboard
  - 1 Mouse
  - 1 HDMI Cable
  - 1 Ethernet Cable
  - 1 Power Cable for PI
- Tools:
  - Wire Strippers
  - Soldering Gun
  - Drill/Drill bits

Step 2: Assembling the Circuit

https://docs.google.com/document/d/1bfwpQHzeOXjBhBCZIY-skSZky7vCUFwD-MJsi70nD6g/edit

Step 3 Wiring

- Put door strike into door frame, route wire through wall and out on inside
- Drill hole in wall for card swipe cable. Run the cable through the wall to the inside
- Drill hole for tablet charger to inside
- Ideally all of these cables would come out of the wall together on the inside

Step 4 Setting up the RPi

- Download the most recent raspbian image here http://www.raspbian.org/
- Write a new SD card (at least 8GB). Details for all OS can be found here http://www.raspberrypi.org/documentation/installation/installing-images/
- Install Mysql, PHP, and Apache, a good instruction guide is located here: http://elinux.org/RPi_A_Simple_Wheezy_LAMP_install. This will allow you to run a small web server to control access to your door, as well as to store information needed by the door access control system
- We will start by setting up Mysql. You should have created a root account and password when you installed mysql. Log into mysql now using mysql -u root -p, followed by your password.
- We will now set up a Mysql Database for our door to use. I called mine doorMaster. Here is an example of how to get started with a Mysql database: http://www.wikihow.com/Create-a-Database-in-MySQL
- Our Database uses 5 tables, acceptedCards, accessRequests, log, adminLogin, and status. To implement our code exactly it is important that all of these tables are identical to ours.
- A mysql dump is available in the repo for convenience
- The following images show the structure of each table in our doorMaster database, the name of the database is at the top of the picture.
-

```
Tables_in_doorMaster
------------------------
acceptedCards
accessRequests
adminLogin
log
logTesting
status
```

```
mysql> describe acceptedCards;
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| card  | text       | NO   |     | NULL    |       |
| name  | text       | NO   |     | NULL    |       |
| email | text       | NO   |     | NULL    |       |
| admin | tinyint(1) | NO   |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```

-
```
mysql> describe accessRequests
    -> ;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| card  | text | NO   |     | NULL    |       |
| name  | text | NO   |     | NULL    |       |
| email | text | NO   |     | NULL    |       |
+-------+------+------+-----+---------+-------+
3 rows in set (0.02 sec)
```
-
```
mysql> describe log
    -> ;
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| accessRequest | text | YES  |     | NULL    |       |
| action        | text | YES  |     | NULL    |       |
| timestamp     | text | YES  |     | NULL    |       |
+---------------+------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```
-
```
ysql> describe adminLogin
   -> ;
+----------+------+------+-----+---------+-------+
| Field    | Type | Null | Key | Default | Extra |
+----------+------+------+-----+---------+-------+
| Username | text | NO   |     | NULL    |       |
| Password | text | NO   |     | NULL    |       |
| Salt     | text | NO   |     | NULL    |       |
+----------+------+------+-----+---------+-------+
 rows in set (0.01 sec)
```
-
```
mysql> describe status
    -> ;
+-----------+------+------+-----+---------+-------+
| Field     | Type | Null | Key | Default | Extra |
+-----------+------+------+-----+---------+-------+
| status    | text | NO   |     | NULL    |       |
| timestamp | text | NO   |     | NULL    |       |
+-----------+------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```
- You should now have an RPi Lamp server with a complete doorMaster Mysql database. You are ready to begin writing your control code, or implementing ours.

Step 5 Backend Coding

- Our code is available here: https://github.com/jonfroehlich/HCILHackerspaceDoorStrike
- The www directory should replace the empty directory /var/www/
- The doorControl directory should be placed in the /home/pi/doorControl
- To have the door swipe watchdog and control program start when the pi boots simply add them to the boot file. A sample boot file is available in the repo. The goal of this file is to start the watchdog on boot, and then reboot whenever the process crashes. This is a messy solution but it fits our needs.
- You now need to update all the passwords. The following files require access to the mysql database, most of these require the password to be added at the top of the files. You may also take this chance to replace the root user login if you created a limited access mysql user
    - doorIB.py - requires mysql username and password

    - doorWatchDog.py
    - all php files in /var/www/ require access to the mysql database (unfortunately in our current implementation we neglected to use includes file for database access so each file must be updated individually)
    - You may use the code as is without updating any files simply by setting your mysql root user password to password, however this is not recommended long term for security reasons.
    - You must add at least one username/password login combination to the admin login table to be able to use the admin page.
- Reboot your pi. Once it's booted again everything should be up and running. Just make sure that your card swipe is plugged into one of the pi's USB slots, the ethernet cable is in, and your door strike is properly plugged into your pi.
- If you instead want to implement your own system, we changed the terminal to character read mode and once data is found in the input buffer we read until no more data can be found (the code is commented for more details)

Step 6: The admin page

- If you correctly set up your RPi's web server, you should be able to access the admin page simply by entering your RPi's ip address into your browser's url bar. Log in using the name and password you added to admin login earlier and you will now be able to view the log of door use, accept or deny user requests for access, and of course remove users from the whitelist. The home page also includes a link to our documentation if further information is needed

Step 7: The User Display

- Our user display consists of a wall mounted Nexus 7. Any internet connected device with a touch screen (or a mouse and keyboard I guess) will do. This display should be pointed towards ipaddress/Testing2/. This URL will display the current status of the door, and is controlled by the doorIB program by changing the index (We recognize this is poor style but it works for our purposes). This front end allows the users to request access
- It's worth noting that this page was specifically designed to be displayed using a Nexus 7 and a kiosk app and will likely not display correctly on other devices.
- This page should be hooked up to a google photos feed to display background images.

Troubleshooting

This is a short guide to troubleshooting the doorstrike. It is not comprehensive but it will try to cover frequently seen issues.

Case: The tablet and Pi are on but the door won't unlock

Step 1: Check that the admin page is working, (PI's IP address, in our case 192.168.1.111 locally).
Step 2: If it is, make sure that cards are being logged correctly and that users are recognized

Result: This is probably a wiring issue. Compare the wiring to the diagrams provided here. In the past the connection between the board and the solenoid has become unsoldered. If the fan is off you may have a problem with your power supply.

Case: The Pi does't turn on.

Result: This is most likely a problem with the SD card. SD card's can become corrupted after many read/writes. Remove the card from the Pi and look at it with another machine. If it looks corrupted simply make a new SD card. (It may be a good idea to create an image of your card to replace later)

Case: No emails are being sent/Google reports unauthorized login attempts.

Result: Google attempts to stop less secure clients from connecting to gmail. You can stop this security measure under account settings, uncheck block low security apps. This is only recommended if you are using a gmail account for this purpose and no other.