BU College of Engineering

Smart Home: If you connect it, protect it!!

EC-545 - Group Project (Fall 2022)
Team: Intruders
Cyber-Physical System

Students in the team:

- Harshit Agrawal - harshit@bu.edu - U92272648
- Zhiqing Wang - joeyw@bu.edu - U28111142
- Román Vélez-Alicea - rgva@bu.edu - U83639966

Advisor: Prof. Wenchao Li

Date: Dec 16, 2022

Github Repository: https://github.com/agrawalharshit/EC545---Smart-Home-Security/

Contents:

Introduction:

According to a survey by *Statista*, the number of Internet of Things (IoT) devices worldwide is forecast to almost triple from 9.7 billion in 2020 to more than 29 billion IoT devices in 2030 [1]. In recent years, the trend toward IoT has led to increased interest in Smart Homes, [2] or homes where key items are connected to the internet. This manifests itself in different ways – voice-controlled light bulbs, live security camera feeds directly to your smartphone, or programmable sprinklers for your lawn. But while the rise in IoT devices means an increase in consumer convenience – it may also be an exploitable weakness.

Our project seeks to simulate how many of the systems in homes of this nature may be vulnerable to attack from bad actors with access to specialized equipment. To illustrate this, the team sought to develop a miniature home scale model, a platform with systems such as RFID-controlled home access, energy-saving lighting through sensor detection, Wi-Fi-controlled room lights, and IR-controlled garage access. Perhaps differently from a real Smart Home – this platform allows us to attack systems beyond just the Wi-Fi protocol.

Using various attack vectors - including a signal-spoofing multi-tool, an RFID programmer tool, and a Wi-Fi MCU – we can disrupt all of these home systems, and observe how they may react beyond the boundaries of their intended design while an intruder attempts to gain access to the house. With this project, we hope to drive home how important proper cybersecurity measures in software are, and how a system with too few fail safes can lead to catastrophic results.

[1] https://www.statista.com/topics/2637/internet-of-things/
[2] “A Guide to Smart Homes in 2021 | Learn All About Home Automation,” *Security.org*. https://www.security.org/smart-home/

Design:

The design of our system is focused on two primary elements – the physical house model, and the accompanying circuit architecture to achieve our specifications. For the physical house model, we used a 1-sq ft plywood platform of ¼-inch thickness, elevated by two 4-inch plywood slats of ½-inch thickness on opposite sides which act as both a supporting structure for the peripherals mounted on top, as well as created an enclosure space for the hardware necessary for the system to function.

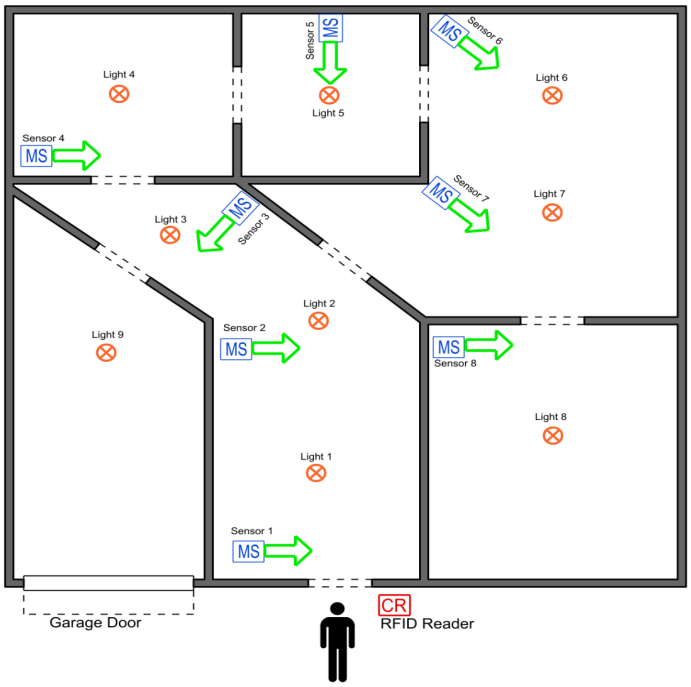


Figure 1: Floor plan design of the physical house model. Several important components are visible here, including the RFID Reader (labeled 'CR'), the Garage Door, as well as the sensors, the direction in which they are facing (green arrows), and the light each one corresponds to (Sensor A corresponds to Light A, etc.).

A power drill with a 0.2090-inch drill bit was used to create the slots through which we would pass the square white LEDs acting as room lights, as well as the slots through which we would place the sensors. This drill bit size was chosen such that these components would fit snugly through the spaces and there would be minimal movement of the parts. In addition to this, two miniature 'person' models for both the 'homeowner' and 'hacker' of approximately 1.5-inches in

height and 0.5-inches in width were cut and glued together, as well as props to give some life to the home - including a car with an approximate footprint of 1 x 3-inches, and a laptop with an approximate 0.5 x 0.25-inches footprint.

The circuitry of the smart home relies on three different Arduino platforms - one Arduino MEGA 2560, one Arduino UNO Wifi Rev.2, and one Arduino UNO. These microcontrollers interact with the system in the following ways:

- The Arduino UNO connects the RFID circuit which allows or denies access to the home. This is an entirely self-contained circuit that does not interact with the other two microcontrollers.
- The Arduino UNO Rev.2 connects to the Arduino MEGA 2560 through the Universal Asynchronous Receiver-Transmitter (UART) [3]. The Arduino UNO Rev.2 TX pin connects directly to the Arduino MEGA's RX pin, and the UNO's RX pin connects to the MEGA's TX pin. This allows for serial communication between the Arduinos, which is a requirement for our system since we are using the Wi-Fi protocol to control the lighting system in the house via the Arduino smartphone application.
- The Arduino MEGA 2560 connects to the peripherals in the smart home system - namely, the LEDs, the photoelectric proximity sensors, the IR Receiver, and the Stepper motor. This Arduino is used to manage inputs and outputs for the sensors and actuators in our system. By receiving input from the sensors, the light corresponding to it turns on, while the remaining lights turn off - this is known as occupancy-based lighting [4]. Likewise, the IR Receiver receives the signal emitted by a remote control (transmitter) which enables the servo motor simulating a garage door to open when button-1 is pressed and close when button-2 is pressed. Likewise, as explained previously, this MEGA takes inputs from the UNO WiFi Rev.2 to control the lighting in the house manually should the user decide to do so.

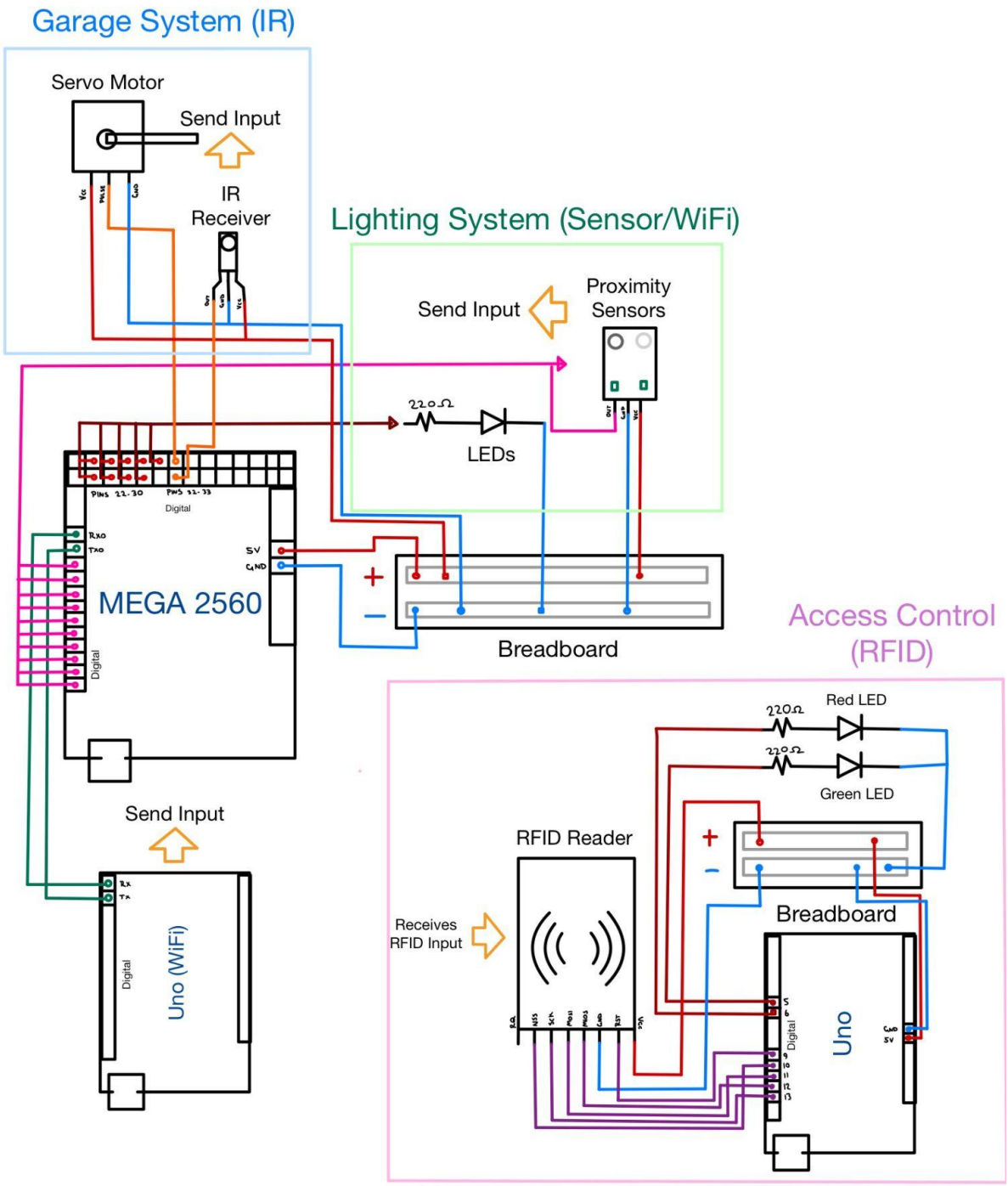


Figure 2: Circuit diagram of the Smart Home system, the outlined sections specify the intended functionality of the circuit, as well as the input methods. The yellow arrows define how inputs are received between the various microcontrollers and peripherals.

Hardware Implementation:

From the circuit diagram in figure 2, we see how all the electrical components interact cohesively to achieve the three distinct subsystems for this smart home. The specifications and electrical behaviors of the system are as follows:-

RFID Control –

The hookup scheme for this subsystem is:

- The Arduino UNO 5V and GND pins are connected to the corresponding (+) and (-) rails on the breadboard, respectively.

Peripherals and Connections			
RFID Reader		LEDs	
GPIO	Port	GPIO	LED ID
9	RST	5	Green
10	NSS	6	Red
11	MOSI		
12	MIOS		
13	SCK		
Ground	GND		

- RFID Reader:
  - The RFID reader is grounded on the breadboard ground rail.

- LEDs:
  - The LEDs are connected to the GPIOs in series with a 220 Ω resistor, by Ohm's Law, this would provide $I = 5V/220\Omega = 22mA$, which meets the forward current rating of 10-30 mA for these standard LEDs. They are both grounded on the breadboard ground rail.

The behavior for this subsystem is:

- The RFID tag programmed for access is read by the RFID reader - if the RFID tag read matches the expected input, the green LED flashes indicating the user has gained access to the home. If the RFID tag does not match the expected input, the red LED flashes indicating the user is denied access. After three consecutive denials, the system enters lockdown mode, whereby both the red and green LEDs flash in unison to indicate the lockdown period has begun. During this lockdown period, the RFID reader will take no inputs from any RFID card – for demonstration purposes, this lockdown period is set at 5 seconds.

Lighting System –

The hookup scheme for this subsystem is:

- The Arduino MEGA 5V and GND pins are connected to the corresponding (+) and (-) rails on the breadboard, respectively.

Peripherals and Connections			
LEDs		Proximity Sensors	
GPIO	LED ID	GPIO	Sensor ID
22	Light 1	2	Sensor 1
23	Light 2	3	Sensor 2
24	Light 3	4	Sensor 3
25	Light 4	5	Sensor 4
26	Light 5	6	Sensor 5
27	Light 6	7	Sensor 6
28	Light 7	8	Sensor 7
29	Light 8	9	Sensor 8
30	Light 9		

- LEDs:
    - The LEDs are connected to the GPIOs in series with a 220 Ω resistor, by Ohm's Law, this would provide $I = 5V/220\Omega = 22mA$, which meets the forward current rating of 10-30 mA for these standard LEDs. All LEDs are grounded on the breadboard ground rail.

- Proximity Sensors:
    - The rotary potentiometer on the sensor dictates the range in which it can detect - adjusting the potentiometer limits the output of the light-emitting element, which corresponds to a shorter range of detection for the light-receiving element in the sensor within a 35-degree detection angle [5]. Through experimentation, the range of approximately 3.8cm perpendicular to the sensors was found to be appropriate for our application, corresponding to a 30-degree turn in the potentiometer. All sensors are grounded on the breadboard ground rail.

The behavior for this subsystem is:

- Each sensor 1-8 corresponds to a light 1-8, where Sensor A sends input to Light A, and so forth. Once a target is detected within the aforementioned range of the sensor, the Arduino MEGA sends an input to the corresponding light, turning it on, and ensuring all other lights are off. This ensures that no matter the path taken around the house, the light previous to the one currently activated by the individual will always be off, achieving occupancy-based lighting.

Garage System –

The hookup scheme for this subsystem is:

- The Arduino MEGA 5V and GND pins are connected to the corresponding (+) and (-) rails on the breadboard, respectively.

Peripherals and Connections			
Micro Servo Motor		IR Receiver	
GPIO	Port	GPIO	Port
32	Pulse	33	OUT

- Both elements' VCC and GND pins are connected to the breadboard positive and ground rails, respectively.

The behavior for this subsystem is:

- A 22-key IR controller is used to send an input to the IR Receiver. This input could be the '1' key (to open the garage door), or the '2' key (to close the garage door). If '1' is pressed, the Arduino decodes the signal and moves the servo 90 degrees from its initial position. It also turns on the light in the garage - the only light that does not take a proximity sensor input. If '2' is pressed, the Arduino likewise decodes the signal, and moves the servo back to its initial position - this occurs at the highest possible rate of 120 RPM.

Materials List –

Smart Home Materials List	
Component	**Quantity**
Jumper Wires	~ 50, varying lengths
220-Ohm Resistors	11
Square White LEDs	9
Red LED	1
Green LED	1
IR Reflective Sensor Module (Songhe)	8
MG90s Micro Servo Motor	1
HXI838 NEC IR Remote Control (KOOBOOK)	1
TSOP38238 IR Receiver	1
RC522 RFID Module	1
S50 Mifare Card	1
13.56MHz RFID Keyfob Token	3
830-Point Breadboard	2
Arduino MEGA 2560	1
Arduino UNO Wi-Fi Rev.2	1
Arduino UNO	1

[3] F. Troya, “Communicating with Four Boards Through UART with Nano Every | Arduino Documentation,” *docs.arduino.cc*, Dec. 15, 2022. https://docs.arduino.cc/tutorials/nano-every/run-4-uart (accessed Dec. 15, 2022).
[4] C. Dilouie, “All About Occupancy and Vacancy Sensors,” *Lighting Controls Association*, Aug. 21, 2017. https://lightingcontrolsassociation.org/2017/08/21/all-about-occupancy-and-vacancy-sensors/ (accessed Dec. 15, 2022).
[5] OMRON, “Overview of Photoelectric Sensors | OMRON Industrial Automation,” *www.ia.omron.com*. https://www.ia.omron.com/support/guide/43/introduction.html (accessed Dec. 15, 2022).

Software Implementation:

Coding for Wifi controlled lighting — *wifi_control.ino* (This code references the tutorial[6])

This implementation of the wifi control property allows two different methods to send a control signal to the lighting system. It could be either using the web page generated by the code, which is running on the server hosted by Arduino UNO wifi rev2, or sending a control signal directly to the server using the Android phone application Arduino Automation[7].

- WiFiNINA library for Arduino was used for connecting the Arduino UNO wifi rev2 to the network with provided SSID and password.
- SoftwareSerial library for Arduino was used to create a software serial connection port to allow communication between Arduino UNO wifi rev2 and Arduino MEGA 2560. This is also the key channel for lighting control through wifi to interact with the local lighting system that has been implemented on Arduino MEGA.

In the setup section of the code, we begin the Serial connection, both hardware and software serial. Next, we try to enable the wifi module on Arduino UNO wifi rev2, if not successful, the script will print “Communication with WiFi module failed!” and use a while(true) loop to stop the code from running and further lines. If it has successfully enabled the wifi module, it will attempt to connect to the network with the given SSID and password(if applicable).

In each attempt for connection, we wait for 10 seconds and start another attempt if the connection was not successful.

After the network connection is successful, the code will start up a server on the Arduino, and print out the IP address to the Serial monitor. This IP address will be used in the phone application Arduino Automation(see Figure 1: Arduino Automation) for the house owner to send the control signals to the server hosted on Arduino UNO wifi rev2.

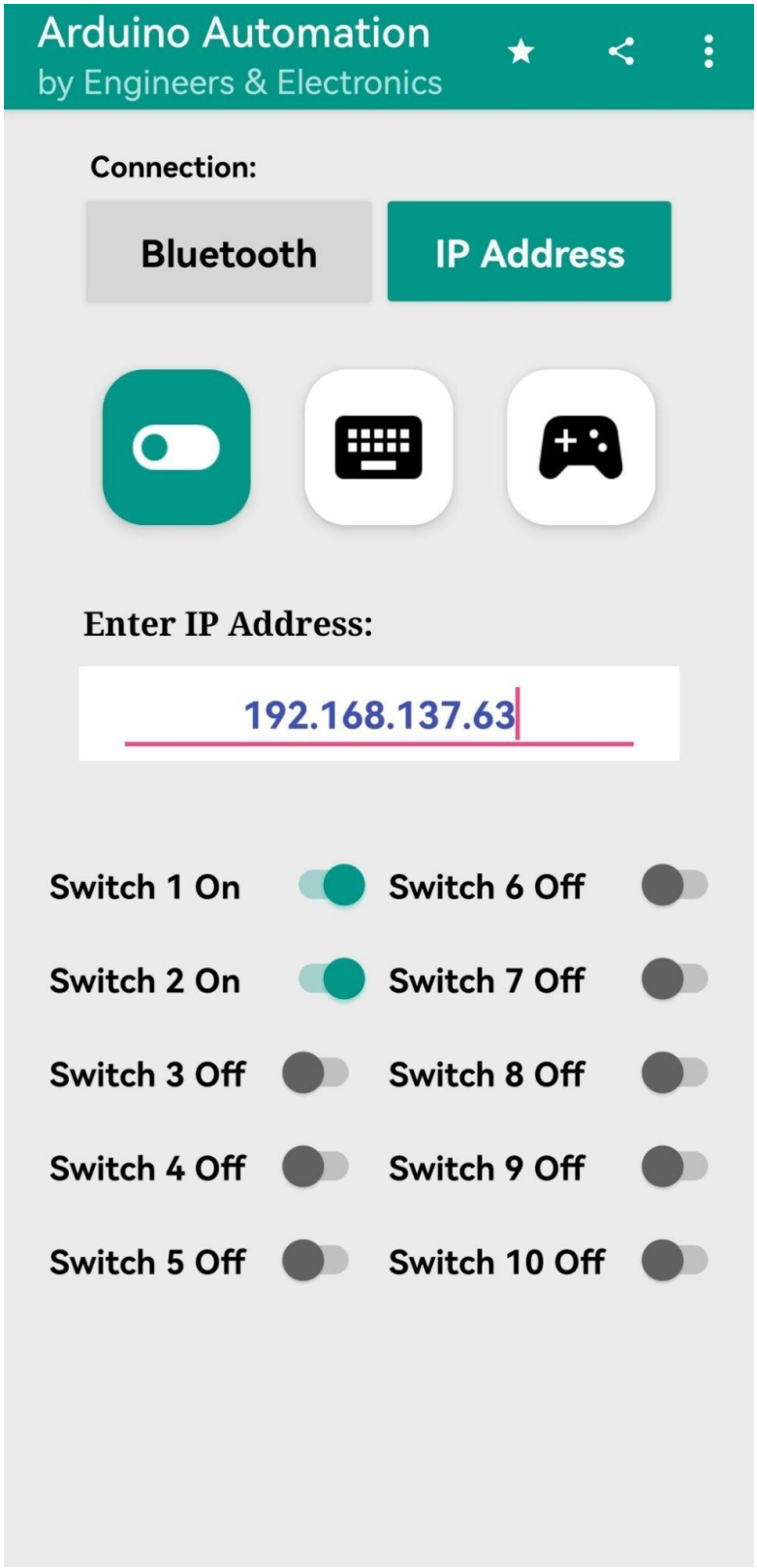
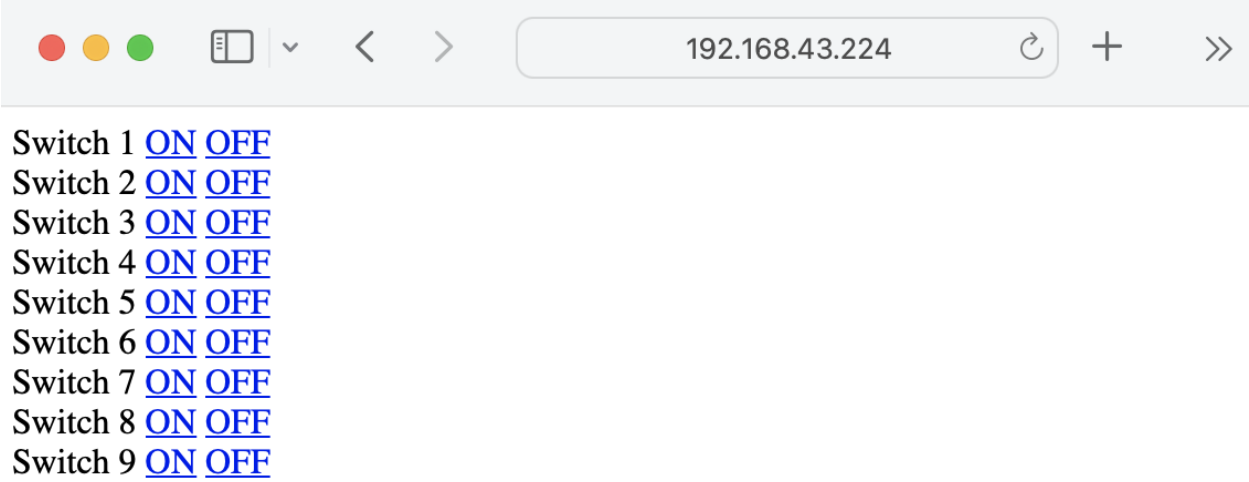


Figure X: Arduino Automation

In the loop section of the code, we will set up a webpage that can also be used to control the lighting system remotely.



The server will keep receiving control signals from either webpage or a phone application. On the Arduino Automation, turning switch 1 on will send an uppercase alphabet “A” to the server, and if switch 1 is turned off, it will send a lowercase “a” to the server. For switch 2 we have “B” and “b” corresponding to the on/off signal. Similar strategies were used in other switches. The signal received by the server will hence be forwarded to the Arduino MEGA 2560 through the software serial port, and perform the actual lighting control. For the webpage control, clicking on the ON links will send uppercase letters corresponding to the switch index, and the OFF links will send lowercase letters.

Coding for the RFID authentication system — authentication_RFID.ino (referencing to examples provided by Arduino library MFRC522[8])

- SPI library: The RFID reader, MFRC522, required the Serial Peripheral Interface to send data to Arduino UNO.
- MFRC522 library: Provide all necessary functions to operate the RFID reader.

In the setup section, we define two pins into output mode for the red and green instruction LED; Initialize the SPI bus for data transition. In the loop section, the code is written to constantly look for a new card; if no card is present to the RFID reader, the code returns to the beginning of the loop section. And next the code checks if the card presented is readable, if not it will also return. Otherwise, the code will print the card UID read by the RFID reader to the serial monitor. Hence compare the UID to the authorized UID, if the same, grant access and turn on the green instruction light for 2000ms and reset the failed attempt count to zero. If UID is not the same, it will turn on the red instruction light for 2000ms and increase the failed attempt count by one.

If the failed attempts have increased to more than 3, it will trigger a lockdown mode simulated by a while loop that will block the RFID reader from reading any new card for some time. Meanwhile, both the green light and the red light will be flashing until the lockdown mode ends.

Coding for Occupancy-based lighting and Garage systems – (also receive control through wifi) — ECE545_SmartHomeMEGA.ino

- IRremote: This library decodes the signals from the IR transmitter in the remote.
- Servo: This library is used to program the micro servo motor used for the garage door system.

Lines 6-7: Define garage door keys used for opening and closing - the ‘key codes’ for buttons 1 and 2 are 12 and 24, respectively. The IR Receive Pin is defined as GPIO 33.
Line 11: Initializes the serial communication of characters incoming from the Arduino UNO Wi-Fi Rev.2.
Lines 14-31: Defines all 8 sensors (GPIO 2-9) and 9 room lights (GPIO 22-30) used within the home.

Void setup: Initialize reception of the IR signals through the ‘IR Receive Pin’. This portion also initiates serial communication with the Arduino UNO Wi-Fi Rev.2 at the default 9600 baud rate. In lines 41-48, the sensors are defined as inputs, and in lines 50-58, the room lights are defined as outputs. In lines 60-61, the servo motor pulse pin is attached to GPIO 32, and the initial position of the servo is set to 0 degrees.

Void loop: Lines 66-73 initialize the reading of the IR Sensor pins for any deviation from a value of ‘1’, which represents ‘no object detection’. In lines 76-164, the algorithm for occupancy-based lighting is defined. If the Sensor Status drops below ‘1’, the light of the same number as the sensor lights up, and all others remain off. In lines 167-216, the Wi-Fi control scheme is defined, comparing the Serial1 readings received from the Arduino UNO Wi-Fi Rev.2 to the letters assigned to each light switch. Case statements were used for code clarity – this control scheme is of lower priority than the physical sensor readings. In lines, 219-237, the garage door control scheme is defined. If an IR signal is present and decoded, it is assigned to the ‘command’ switch statement. In the case that the decoded signal is ‘12’, the switch moves to the Garage_Open portion, moving the servo counterclockwise 90 degrees, and turning on the garage room light. In the case that the decoded signal is ‘24’, the switch moves to the Garage_Close portion, moving the servo counterclockwise 90 degrees, and turning the garage room light off.

---

[6] T. A. Team, “Host a web server on the Arduino Uno WIFI REV2: Arduino documentation,” Arduino Documentation | Arduino Documentation. [Online]. Available: https://docs.arduino.cc/tutorials/uno-wifi-rev2/uno-wifi-r2-hosting-a-webserver. [Accessed: 15-Dec-2022].

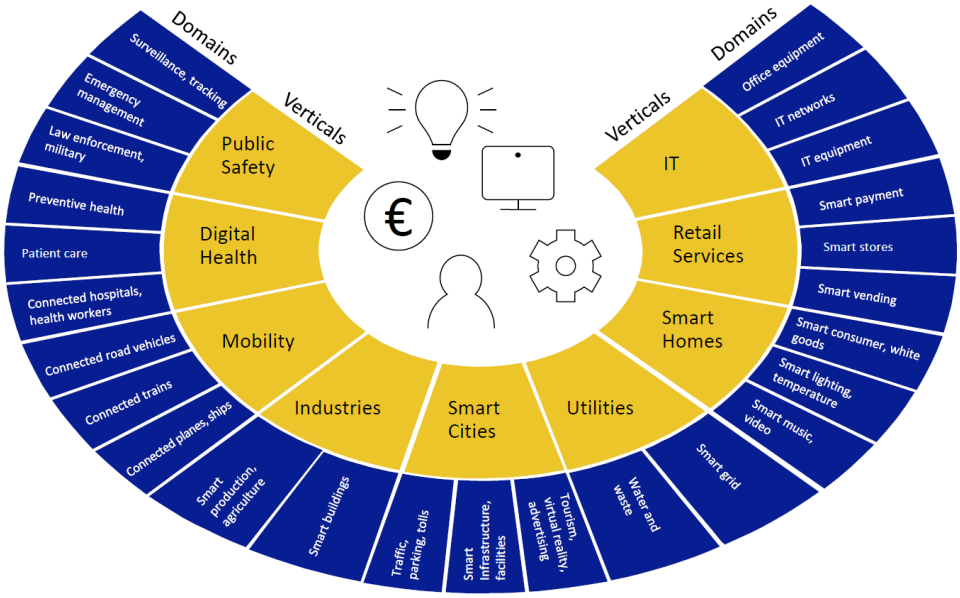
[7] E&E: Arduino Automation - Apps on Google Playhttps://play.google.com/store/apps/details?id=com.himanshu.ArduinoAutomation&hl=en&gl=US&pli=1.[Accessed: 15-Dec-2022]

[8] Miguelbalboa, “Miguelbalboa/RFID: Arduino Rfid Library for MFRC522,” *GitHub*. [Online]. Available: https://github.com/miguelbalboa/rfid. [Accessed: 15-Dec-2022].

Attack Vector Implementation:

The Internet of Things is a bridge from the physical world to the cyber world or in other words the broad network of devices or sensors that exchange data without human interference and communicate with each other to bring much of the physical world from commonplace devices to industrial assets to people into a connected ecosystem, resulting in enhanced experience and better outcomes.

With increased digitization and connectivity of industry, it has produced exponential growth in attack surface, for example, Airport security can be improved in multiple aspects using IoT, such as using NFC tags, geolocation, and WiFi beacons can regulate overall traffic flow easily, and facial recognition or biometrics can streamline security checkpoints, whereas RFID tags and sensors help with reducing the baggage loss. Given the potential and influence that IoT has on the world, it is worth looking at its transformational impact across various sectors.



(Image Source: Srinivas Bhattiprolu, Nokia Software Trends and Best Practices in IoT Security)

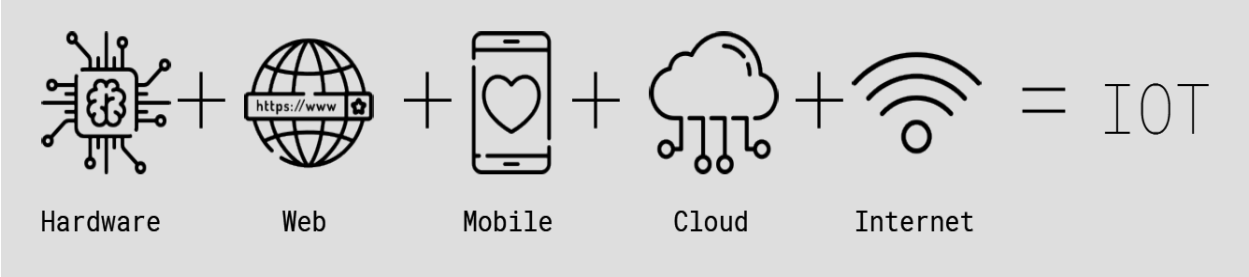
Generally, all IoT devices are capable of:

- Collecting data
- Processing information
- Transferring data

The architecture of Smart Home Security can be broadly divided into three categories:

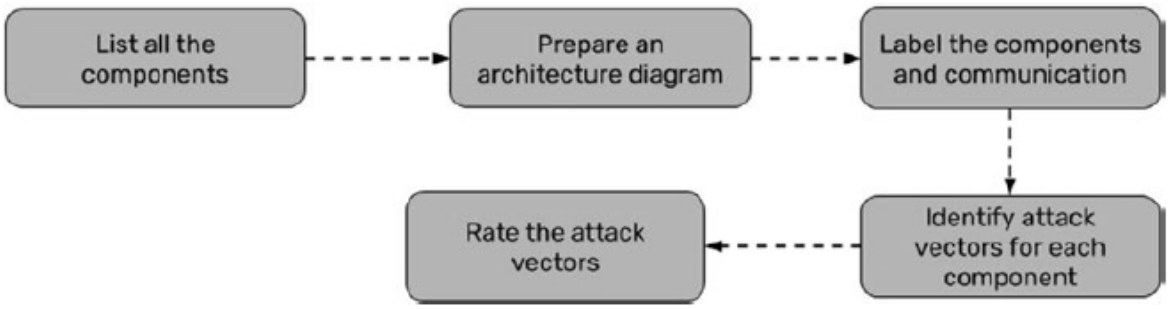
- Embedded device
- Firmware, software, and applications
- Radio communications

When we talk about secure architecture, our thought process should be to categorize the potential vulnerabilities according to its functionality and threats to each category.



IoT Attack Surface Mapping –

The first step and one of the most important steps in overall IoT Threat Modeling methodology is to understand all the components in the product, its architecture diagram from a pentester's perspective, and map out all the different entry points that attacker can exploit in our device ecosystem. Attack surface mapping helps in understanding the overall architecture of the device, and also helps to establish a priority order for various tests we want to run against the product.



Additionally, when adversaries find a new target with high-severity vulnerability, the security team needs to immediately inform the security architect, and vendor with its detailed summary, and this is handled the same day instead of waiting for the engagement to be completed but before that team needs to familiarize themselves with the device. Starting an assessment with missing information is one of the biggest mistakes pentesters can make. This can be assessed by gathering information from Device documentation, and manuals and going through all possible channels.

Attack priority = exploitability x impact

- High priority, high severity vulnerability:- If the exploit is very simple and successfully compromises and retrieves sensitive data from the device
- Low-severity, low-priority vulnerability:- If the exploit is difficult to execute and output obtained during testing is not very useful

---

Threat Modeling of Smart Home –

To get a better understanding of the different phases of threat modeling, we did the security modeling of our smart home.

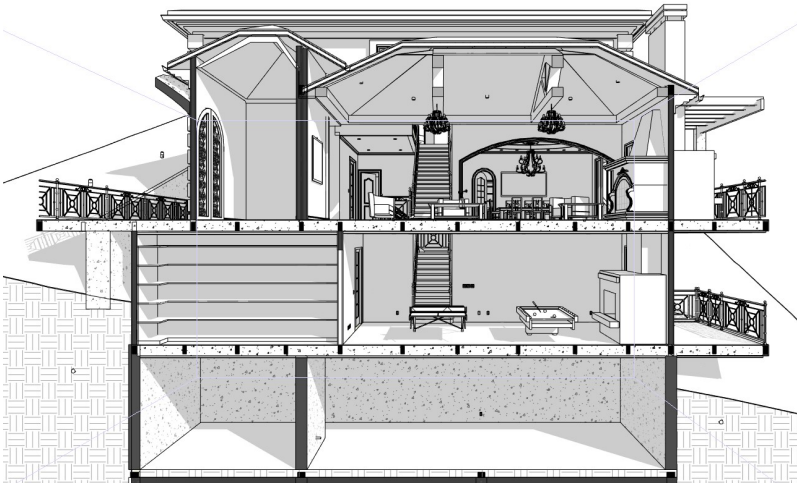
Before we start we need to understand the basic assets a home could have, and what we are trying to protect.

- People in the home
- Items within the home
- The home itself

And what concerns do we have?

- Accessible windows on the ground floor
- Garage access
- Woods in the back, an adversary might come from there
- Woods in the back, one might fall on the home
- Woods in the back, forest fires
- Lack of lights and fencing around the home
- Possible for natural disaster area (tornado, earthquake, floods, etc)

When threat modeling, it is important to understand what we are trying to protect in our system and the first step towards that is to break down the features to understand what was built, and how can someone enter or exit the building. To make it more clear we can look at the top-level or rear view of the house, similar to these blueprints, whenever we develop software, we must create a technical design for the requirements as it helps programmers fully understand what are the requirements, libraries, components, database, and constraints. We need this information to understand threats and risks to the design.



How can someone enter or exit this home is similar to data coming in and out of the system

- Windows, the doors leading outside

Where can this diagram be more clear?

- Where do those stairs lead to?
- Is there an easy way for someone to gain access at a different house level?
- There looks like a door leading out of the family room, etc
- What protections are there at the windows, doors, etc?

If we think about the different types of attacks that can harm this home we can list them as raccoons, bears, rats, or even humans. Concerning humans, there are various types of human attackers, kidnappers, burglars, or thieves which can be understood as different attack methods by the same threat actor. It's important to define the scope of risks in the real world, for this example, we can stick with thieves. Understanding the attacker can help determine what can go wrong, and help prioritize the risk (high, medium, or low).

If we are worried about a burglar maybe they will be brazen enough to steal things while we are in the home and we will need public-centric protections. If we are worried about a thief, perhaps we need to worry about our things rather than the people in the home. This is similar to Threat Modeling in our application. If we are concerned with script kiddies, we will consider certain security defenses versus if we are concerned with various nation-states attacking us.

We can consider these seven threats for the group:

- Breaking in through unlocked windows
- Insider threat: Security guard might be bribed to let someone in
- We have a cheap lock for our doors
- We have a simple garage door opener
- Network is not secure
- Wi-Fi is not having adequate security protocols
- RF Shielding is not done to protect systems

When we prioritize these seven threats it's important to evaluate which one should be a highest and lowest priority by looking at the weakness of the system as that helps mitigate the threats efficiently.

Although there are many methods used to accomplish this, not all are comprehensive, some methods are public-centric which requires different teams to work together for the assessment, some methods are specific for risk or privacy concerns while others encourage granularity and focus on abstraction. Security architects suggest a well-rounded view of the system and for creating a more robust view different threat modeling methods can be combined.

RFID Attack –

In our system we used RFID cards as the access management system to protect the main doors of the home, which is operating on 13.56MHz Frequency based on Mifare Classic card, to break into the system, we used FlipperZero and Proxmark RDV3 chipset to learn about the tag type, and break the encryption keys.

On scanning with the "hf search" command, we learned the tag is of ISO14443A type. We performed three different types of attack on it:

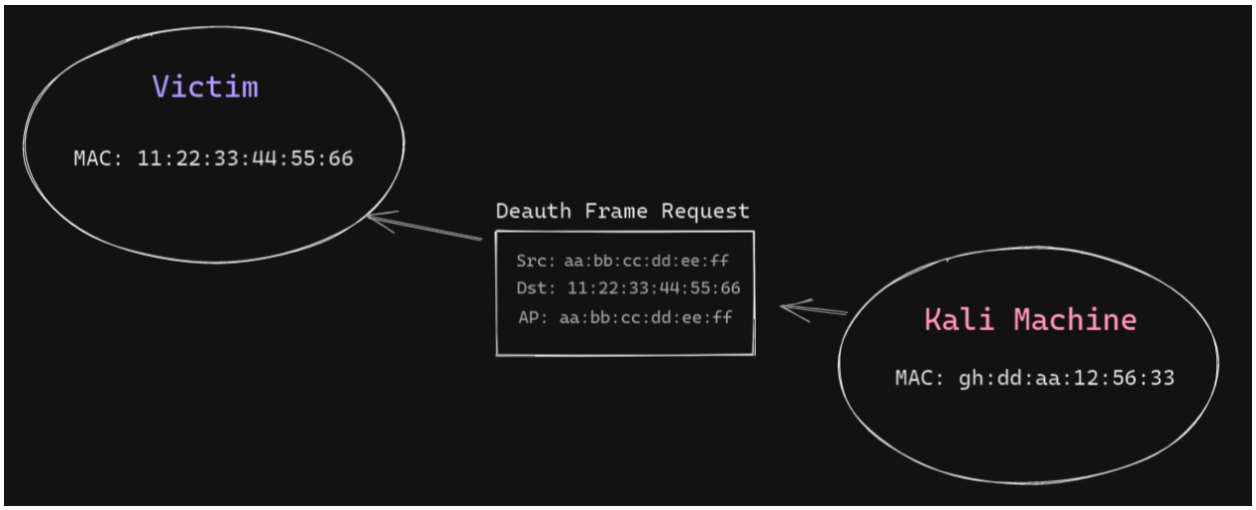
1. Spoofing and Cloning
2. Eavesdropping and Replay
3. Man-in-the-Middle & Sniffing

Once learned about the encryption key and card type, we can clone the UID of the owner card into the magic card which supports the Block0 Byte-4 UID changeable feature. Once the UID is cloned, we can either emulate it from Proxmark or the card, meanwhile, Flipperzeo can be used for performing MITM, but it doesn't allow copying the metadata of the tag to other cards, we can

only emulate the signal from it to the receiver. To perform MITM with the Proxmark, we used hf standalone firmware by user crazybyte on Github, allowing us to emulate scanned 14a UID and store it in device memory which we later reused for analysis.

WiFi Attack:-

WiFi was used to control lights over smartphones. Here we performed two different types of attacks using ESP8266 Chipset, and Alfa Atheros cheapest in monitor mode. The first attack was to deauthenticate the owner's mobile from Access Point, by sending a Deauth Frame Request using AlFa, meanwhile, for the second attack we used ESP8266 to create a fake SSID with the same AP name as of victim to let the owner connect to attacker network for intrusion based attack:



airodump-ng mon0 -0 30 -c <channel> -a <mac address of AP> [9]

- -0 sends the deauth packet
- 30 refers to the number of packets
- -c is the MAC address of the client to be deauthenticated.
- -a is the MAC address of the AP

This attack is performed at the Datalink layer, as associated with MAC Address

---

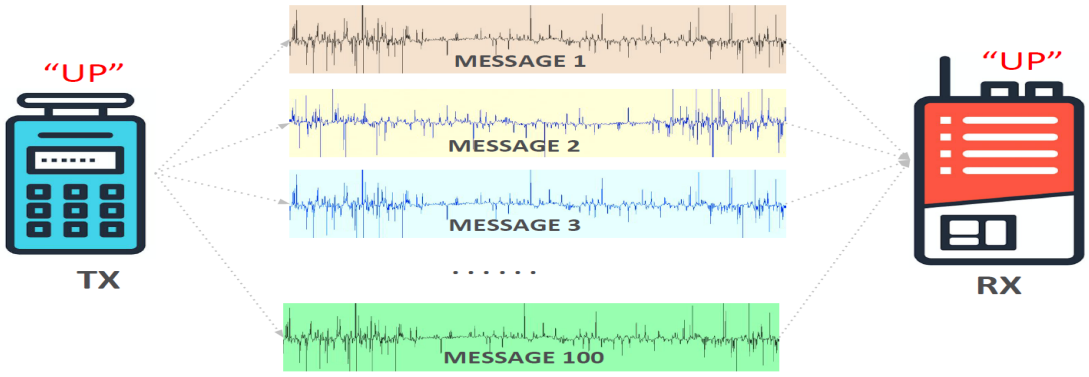
[9] http://www.aircrack-ng.org/doku.php?id=aireplay-ng

Network Intrusion Attack –

To perform this attack, we need the victim to either connect with our network, or else the attacker needs to proxy chain the victim to perform MITM. In this case, the smart home application to control the light was using the HTTPS protocol which is an encrypted data exchange, and the attacker can't read any information other than metadata in Wireshark. To read the data packets and create custom commands to control the light it was necessary to understand the underlying communication for which we used the BURP Suite Professional tool to intercept the packets and by phishing the victim to install fake CA certificates to decrypt the HTTPS. Once done successfully, we were able to see all HTTP get & post requests. Here we used the Repeater to learn about server response when the lights are turned on. Attackers can send custom HTTP requests to the server without the need for the victim to control lights once the packet format is learned and modified.

Car Key Replay Attack –

To exploit this vulnerability we used HackRF One to record the RAW signal at frequency 315.600MHz, which we identified in two ways, first by pressing the button and looking at the spectrum to locate spike (i.e SDRsharp, GQRX), while the other way is to search for FCC ID on car key and searching in datasheets to see RF transceiver information. The second method is only effective when we can access the victim's car key for inspection, while for the first method attackers need to be nearby when the victim is unlocking his car. We used the SDRsharp as a spectrum analyzer to record the RAW signal in WAV format, and Universal Radio Hacker to perform analysis on it. It was easy to identify the signal type as ASK format based on a waveform, else sigwiki dataset could be used here to look for similar signals in that frequency to learn more about it. Once we know the modulation, the digital signal could be broken into chunks to extract the preamble, data, and sync word for further analysis, creating a new signal for transmission.

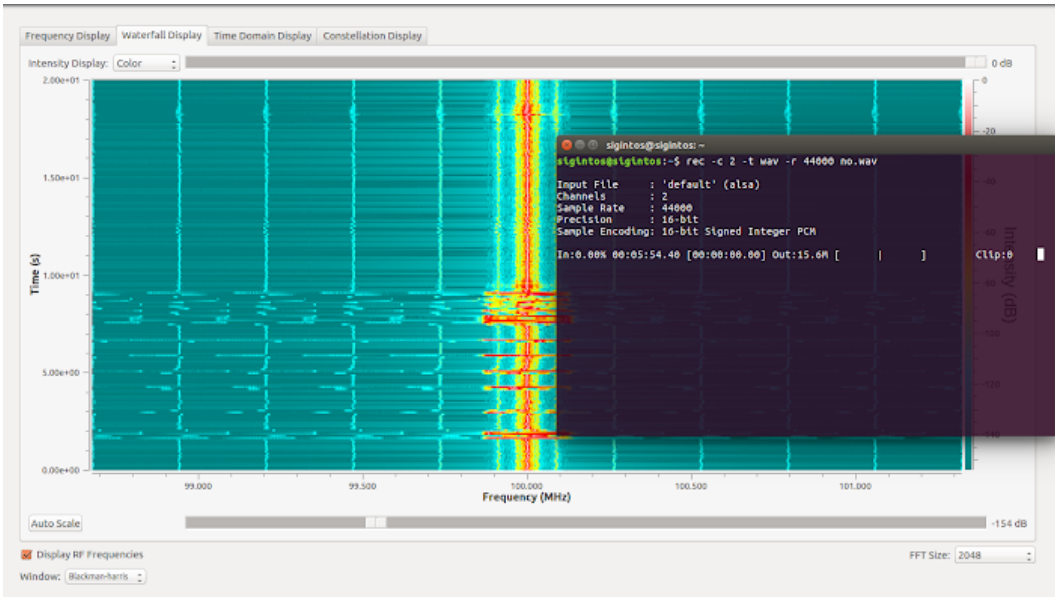


BadUSB Attack –

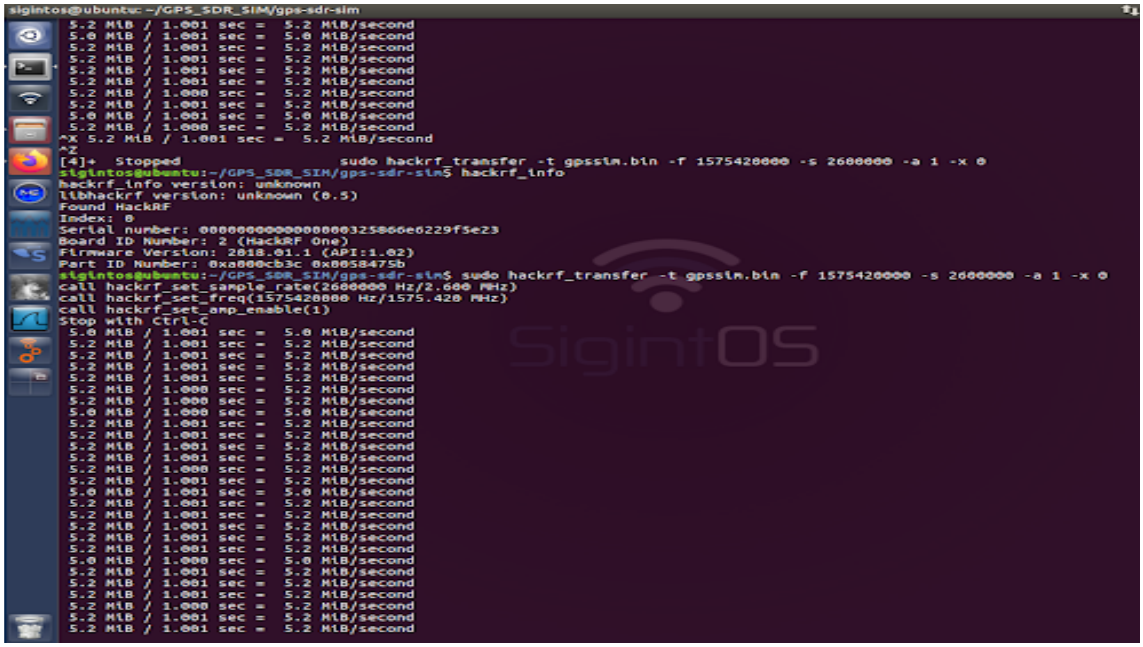
We have written the PowerShell script for FlipperZero, which when connected to a Windows PC acts as a standard HID device instead of storage media, and executes the script at super fast speed. The attack vector was to demonstrate, if an intruder can get access to the victim's computer even for a few seconds, they can create a backdoor. When plugged in the script creates a note as follows:

Other Attack Vectors we tried were Live Fraudulent FM broadcast by Jamming, GPS Signal Hijacking to spoof location, and TEMPEST Attack to render the victim's TV screen from another room based on EMF signals.

FM Broadcast:-



GPS Spoofing:-



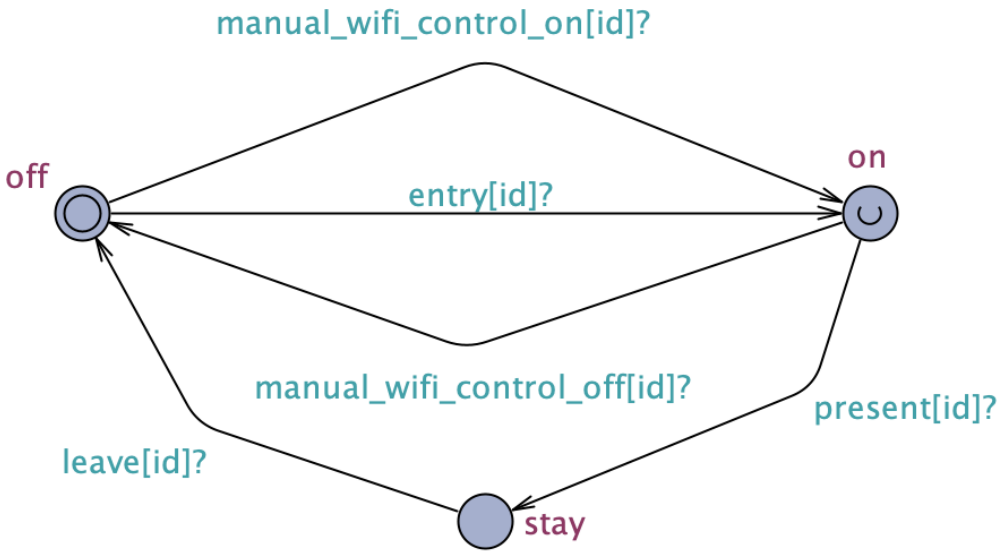
Specification/Modeling:

Smart Home specifications:

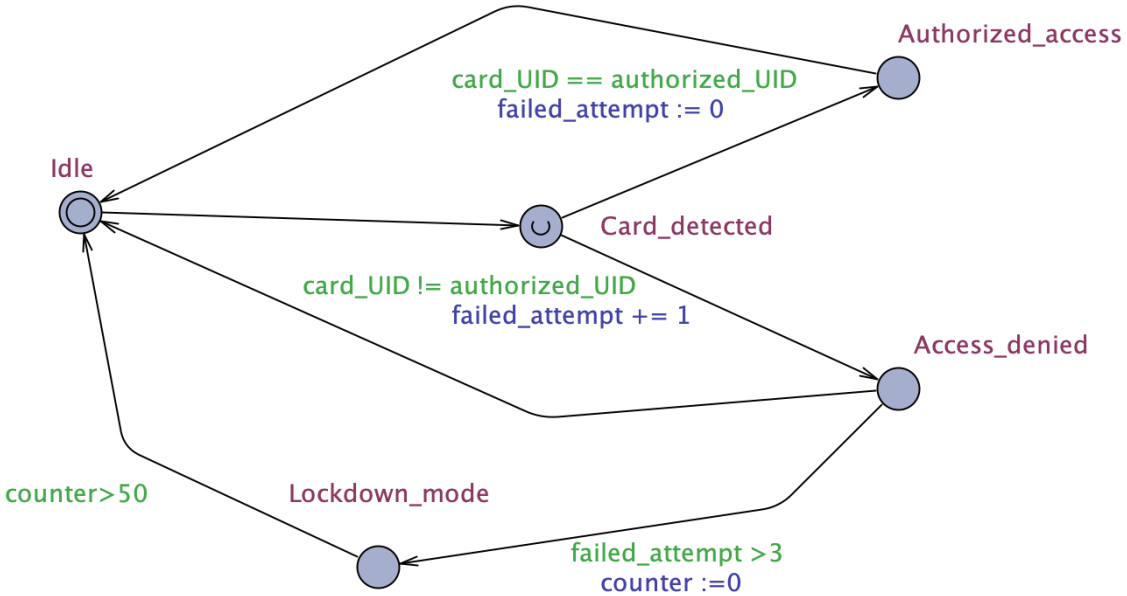
1. Occupancy-based lighting system needs to achieve: When a person presents a room, the light is automatically turned on. And it will not turn off unless the person leaves or there is a manual on/off action via WiFi control.
2. Photoelectric Proximity Sensors for detecting the presence of a person in the room.
3. RFID authentication system on the main entrance of the house. A lockdown mode should be triggered after more than 3 failed attempts.
4. Using Arduino Uno Wifi rev2 to allow lighting control over an Android phone application.
5. Arduino MEGA 2560 is responsible for controlling the lighting system based on the sensor's feedback and Serial communication received from Arduino Uno Wifi rev2.
6. Arduino Uno is controlling the RFID authentication system corresponding to the main door.

UPPAAL model

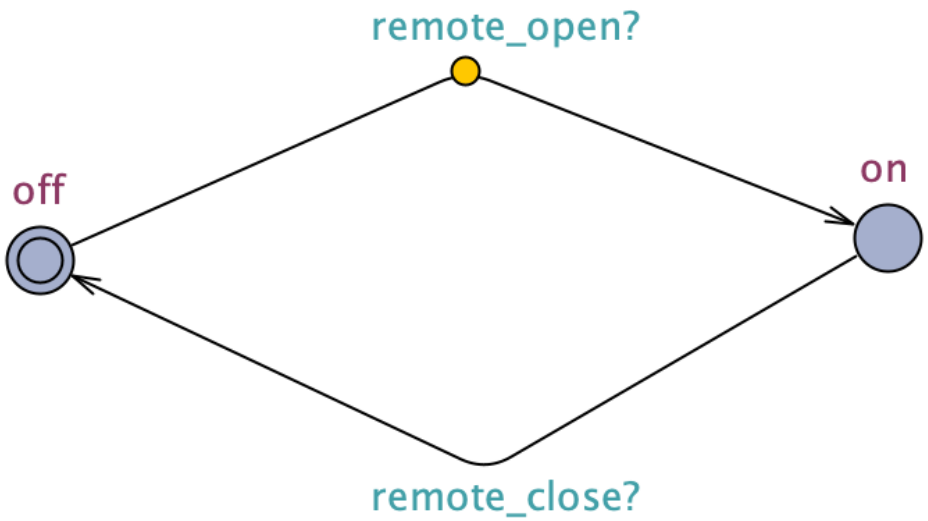
Occupancy-based & wifi control lighting system:



RFID access authentication system:



IR-controlled garage door



- The remote control opening the garage door will trigger the light in the garage to turn on, and the light will turn off when the garage door is closed.

Analysis:

Smart Home Functionality Analysis Overview –

1. The occupancy-based lighting system is indeed achieved by the physical house model, see the demonstration video section “Occupancy-based lighting”
2. Wifi-control of lighting is also been proven to work as specified, see demonstration video section “WIFI control lighting”
3. RFID demonstration included both successful access and access denial scenarios. For the performance of lockdown mode, see the demo video “Lockdown_mode.mov”

Proximity Sensor Analysis –

$$f\big(s(d)\big) = \begin{cases} 1 - as(d) & if\ s(d) \le R \\ 1 & if\ s(d) > R \end{cases}$$

Eq. 1 – Affine model of the Proximity Sensor, where s(d) stands for sensing as a function of distance, and R stands for the range of 3.8cm [10].

This function has two cases, if the sensing distance is greater than 3.8cm, the value of the sensor is set to 1. As soon as the sensing distance is less than or equal to the maximum range, the value of 1 drops proportionately to the sensing distance. The code only distinguishes a *change* in the value of 1, thus the expected behavior is always achieved in the sensor barring some technical malfunction.

Lighting System Scheduling Analysis –

The lighting scheme in this home follows a *preemptive priority-based scheduler* model. This scheduler supports the arrival of tasks at any time during runtime and continually executes the

task with the highest priority [11]. This scheme works for our system, since all tasks are aperiodic, and the sensor-mediated lighting takes priority over the manual WiFi control.

The following trace tested on the physical design illustrates this:

- Activate sensor 1 → Light 1 Turns on, all others off
- No sensor input
- Activate lights 3, 4, and 5 via Wi-Fi control → Lights 3, 4, and 5 turns on, light 1 remains on, all others off.
- Activate sensor 6 → Light 6 Turns on, and all other lights turn off.

Thus, more generally:

- $Sensor\ Input\ =\ P1$
- $Manual\ Input\ =\ P2$

  If:

  - $P(Current\ Input)\ >\ P(Last\ Input)$: Execute current process
  - $P(Current\ Input)\ <\ P(Last\ Input)$: Resume executing last process
  - $P(Current\ Input)\ =\ P(Last\ Input)$: Execute whichever light input last arrived

---

[10] E. Ashford and S. A. Seshia, *Introduction to embedded systems: a cyber-physical systems approach*. Cambridge, Ma: MIT Press, 2017, p. 182.

[11] E. Ashford and S. A. Seshia, *Introduction to embedded systems: a cyber-physical systems approach*. Cambridge, Ma: MIT Press, 2017, p. 327.

Technical Challenges:

Throughout the project, several obstacles were identified and mitigated to the best of our ability. The root cause of these challenges varied – from time frame issues to parts not arriving, to the system simply breaking down at the last minute. One such challenge was the intended use of an IR controller to control the garage door opening and closing – before integration into the system, this system was working perfectly well. However, upon integration, we suspect the controller's IR transmitter failed. With no replacement available at the time and the deadline approaching, we opted to continue the demonstration without it. This was significant, as it cut the attacks intended for demonstration by the 13th. Additionally, we had intended to implement an RF component to demonstrate usage of the HackRF – in the end, this idea was rejected in favor of parts we already had access to, namely the IR control scheme, as we were already concerned with the growing complexity of the system. Finally, at the beginning of this stage we suggested that our project should provide tangible mitigation strategies to an attack on the home – however, no defense strategies (other than the lockout mechanism for RFID), and no distinction between homeowner and intruder could be made, as it was thought that it would detract from the purpose of the system. We did learn much in the process, especially patience, and as the project concluded we are happy with the results despite not being able to implement all elements as planned.

The following tables summarize some of the changes made to the scope of our project during the iterative design process, for both the Smart Home and Attack Vectors:

Smart Home Specifications			
Previous Specification	**Current Specification**	**Reason for Change (if any)**	**Current Specification Achieved?**
When a person occupies a room, the light will not turn off unless via WiFi control.	No change.	(N/A)	Yes
Can distinguish between homeowner and intruder, and trigger a defense mode.	**RFID door access locks out after three failed attempts**, triggering a defense mode.	The scope narrowed down.	Yes
Two Arduinos - Automated Systems Control (MEGA), WiFi Control (WiFi Uno) which can override commands from the MEGA.	**Three Arduinos - Automated Systems Control (MEGA), WiFi Control (WiFi Uno), and Door Access (Uno).**	Easier implementation, MEGA board GPIOs oversaturated.	Yes
Defense strategy against an intruder in different cases: owner absent/present.	Not implemented.	Revised scope, simplification.	N/A
When no one is in the house, all lights turn off after 5 minutes of no input.	Not implemented.	Revised scope, simplification.	N/A

Attack Vector Specifications			
Previous Specification	**Current Specification**	**Reason for Change (if any)**	**Current Specification Achieved?**
Smart lighting **controlled via RF**, attack focused on jamming signal	Smart lighting **controlled via Wifi**, attack focused on deauth and intrusion	RF relay never arrived.	Yes
RFID Door Access, an attack focused on using Proxmark3 to clone cards.	No change.	(N/A)	Yes
IR controlled lights, attack focused on replicating signals via FlipperZero	**Proximity sensor/WiFi controlled lights**, attack focused on control over WiFi host	Revised scope, practicality concerns.	Yes
Arduino MEGA takes input from Wifi Arduino, deauthentication through ALFA WiFi chip / ESP8266 in promiscuous mode.	No change.	(N/A)	Yes
Attack strategy against a victim in case of owner absent/owner present	Not implemented.	Revised scope, simplification.	N/A

Division of Labor:

- **Harshit Agrawal:** Attack Vector Implementation, Threat Modeling, Analysis
- **Román G. Vélez-Alicea:** Design, Hardware/Software Integration, Analysis
- **Zhiqing Wang:** Modeling, Hardware/Software Integration, Analysis

---

Conclusion:

The more devices we connect with our smart home, the much more likely it is to get hacked. Attackers can program these devices to attack others, vendors can collect metadata of user activities. Manufacturers design these devices for low prices without considering adequate security protocols, which raises many concerns. Through this project, we demonstrated how different IC chips, communication protocols, sensor integration, and relay capabilities help with daily tasks.

In contrast, through threat modeling, we tried to look into security considerations from the design phase. Cyber-Physical Systems helped us understand the concept of designing, Modeling, and Simulating the functionality of all the systems we integrated into the Smart Home security project.

When we talk about attack vectors, it's important to learn about mitigations to secure them. During our analysis phase, we found several best practices that, if followed, can ensure some layer of security and safety to leverage the benefits of connected devices.

Key steps to mitigate smart home:-

- Keep all software, firmware and libraries updated
- Know what IT assets you own
- Connect only what you need, and use what protocol they communicate with
- Secure Passwords
- Authentic Vendor who does a safety check, and provide privacy options
- Secure and separate network connection for personal usage and guest users

References:

1. Mary Asante, Carsten Maple, Gregory Epiphaniou, "Prevention of IoT-Enabled Crime Using Home Routers (PITCHR)", Science and Technologies for Smart Cities, vol.442, pp.470, 2022.
2. Manju Lata, Vikas Kumar, "IoT Network Security in Smart Homes", Cybersecurity in Smart Homes, pp.155, 2022.
3. Noureddine Amraoui, Belhassen Zouari, "Securing the operation of Smart Home Systems: a literature review", Journal of Reliable Intelligent Environments, 2021.
4. N. A. Chattha, ‘Nfc — vulnerabilities and defense,’ in 2014 Conference on Information Assurance and Cyber Security (CIACS), 2014, pp. 35–38. doi: 10.1109/CIACS.2014.6861328.
5. N. K. Singh, ‘Near-field communication (nfc),’ Information technology and libraries, vol. 39, no. 2, 2020.
6. S. A. Weis, ‘Rfid (radio frequency identification): Principles and applications,’ System, vol. 2, no. 3, pp. 1–23, 2007.
7. W. Huang, Y. Zhang and Y. Feng, ‘Acd: An adaptable approach for rfid cloning attack detection,’ Sensors, vol. 20, no. 8, p. 2378, 2020.
8. OWASP Foundation. OWASP IoT Top 10- 2018. Tech. rep. OWASP Foundation, 2018. url: https : //owasp.org/www- pdfarchive/OWASP-IoT-Top-10-2018-final.pdf
9. CYBER; Cyber Security for Consumer Internet of Things: Baseline Requirements. Standard European Telecommunications Standards Institute (ETSI), 2020. url: https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.01_60/en_303645v020101p.pdf
10. Geir M Koien and Thomas Haslestad. “Security aspects of 3G-WLAN interworking”. In: IEEECommunicationsMagazine41.11(2003), doi: 10.1109/MCOM.2003.1244927.