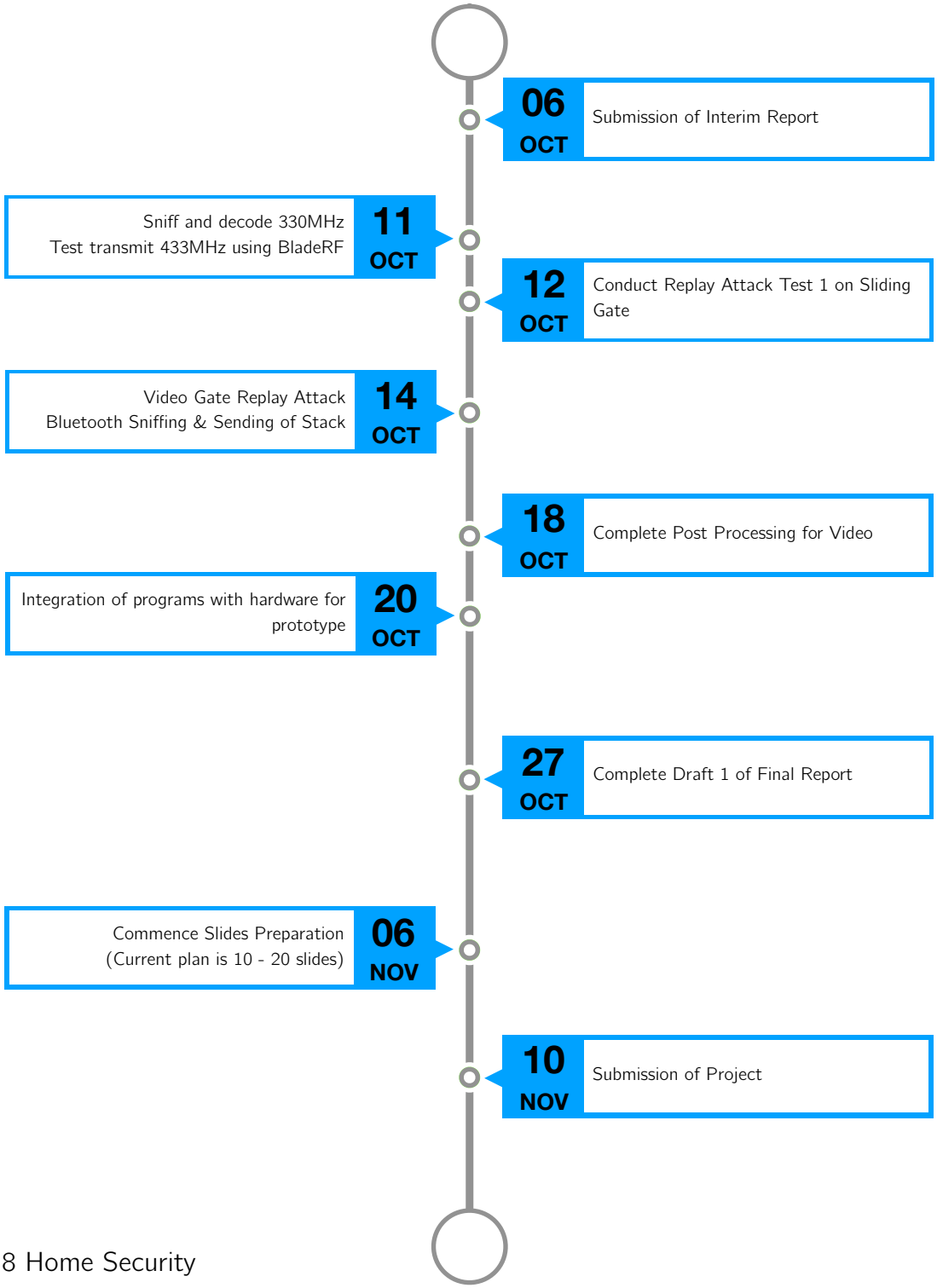
Introduction

With the boom of technological advancement in recent years, smart homes have emerged to improve the convenience factor living in the house: smart locks, smart lights... But how secure are those items? This project aims to look at how secure at some home devices by conducting and testing if a replay attack would allow the adversaries to gain access to the house. In particular, this project will look into the first point of security for homes, i.e. gates and locks.

This report aims to update you on our progress and our target for the next month approaching the submission deadline of the project. Below is our timeline plan:



Loaned Items

Description	Quantity
BladeRF	01
Software Defined Radio	01
Adafruit Bluetooth LE Sniffer	01
Raspberry Pi Model B	01

On top of the equipment that we loaned from you, we are also making use of these other equipments / hardware:

1. Arduino Uno clone
    - To communicate with the receiver and transmitter pair.
    - For final prototype to communicate with the motor.
2. 433MHz Arduino transmitter and receiver
    - For learning how the radio frequency works and testing.
3. PuckJs
    - Programmable bluetooth button authentication for prototype.

Work Done So Far

Category	Description
Setting up	• Set up and downloaded relevant dependencies and softwares for the devices / hardware to be used.

Radio Frequency Sniffing	• Learnt the operating frequency of the gate (330MHz)
	• Sniffing successfully done with BladeRF loaned using GQRX.
	  • Sent out a 433MHz signal using a Arduino Uno.
	  • Listened and exported that signal as a .wav file.
	  • Used Audacity to manually decode.
	  • Was able to get back the same string of 0s and 1s.
	• Installed GnuRadio Companion.

Bluetooth Sniffing	• Identified the Bluetooth name to listen to for house lock. • Filtered and exported sniffed data to WireShark. *Kevo is implementing another company's software (UniKey). If a replay attack is successfully, the vulnerability should apply to all locks that are using the same version of UniKey's software.*

Problems & Difficulties Encountered

Category	Description

Radio Frequency	• Transmitting a signal using hardware transmitter is easy.
	  • Done on Arduino with the 433MHz with RCSwitch library.
	  • However, it is hard to find a 330MHz transmitter / receiver.
	  • Turned to SDR for listening.
	• BladeRF was loaned to help with the transmitting of signal. Clueless about how to transmit signal using GRC, will need to look into it.

Bluetooth	• Bluetooth signal that was sniffed was in binary / hexadecimal format.

Future Work

Category	Description
Radio Frequency	• Getting familiar with GnuRadio Companion and BladeRF. • Sniff and decode the 330MHz signal from gate remote. • After decoding the signal, transmit the signal to test replay attack at 330MHz. • Test the maximum distance that can still listen to the gate signal.
Bluetooth	• Transmit Bluetooth data in binary or hexadecimal format. • Test out if the lock is susceptible to replay attack.

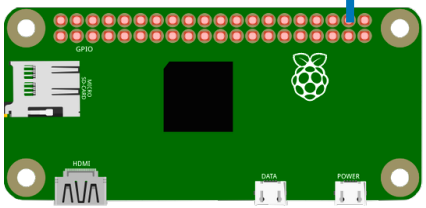
Prototyping (Appendix A)	• Using Raspberry Pi, PuckJs and Arduino
	  • Raspberry Pi and PuckJs for authentication.
	  • Arduino to power the motor.
	• Design and prototype a miniature gate mechanism for demo purposes.

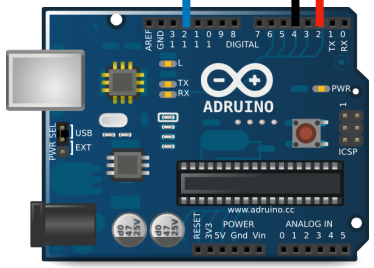
Conclusion

Originally, we thought that the gate was operating on infrared as a medium for transmitting commands to the gate motor. However, upon further research, it was using a 330MHz, so the project scope shift from replaying infrared to replaying radio frequency. This project is progressing slower than expected due to the difficulties encountered but we still believe that we can complete the project by the deadline. The pace of the project should pick up once mid-term ends. ¶

Appendix A

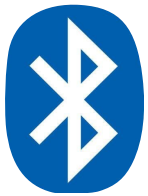
Serial connection to Arduino
To send command when user is authenticated



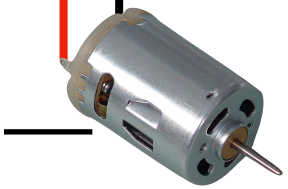


When Arduino receives the command to open / close the gate, it writes to the pin to start the motor.

- Raspberry Pi constantly listens for advertisement.
- PuckJs sends out advertisement with the current date and time as a header, encrypted.
- Raspberry Pi receives and checks with it's current time, if within the threshold, user is authenticated.



The motor then causes the gear to spin which enable the gate motion with the gear rack attached.



For the actual gate application, sensors must be added to signal to Arduino when to stop the signal else the torque of the motor might damage the gate. However, for the purpose of the demo, we plan to hardcode the amount of time that the motor spins.