Name : Mohit P Sathyaseelan

HW 1: UART Security Evaluation

Terminology (10 Pts):

1. What is Risk?

a) Risk is vulnerability into threat.
   i) Threat is the danger posed to the system operation.
   ii) Vulnerability is how likely the attack is gonna succeed

2. Give an example of Confidentiality in security.

a) Disclosing inclusive company information. no information leakage should be the goal.

3. Give an example of Integrity in security.

a) Using botnets to attack the system, to modify its performance(output)

4. Give an example of Accessibility in security.

a) A brick. It can be secure but there is a question of accessibility there.

Security Methods (70 Pts):

1. Is a default (no implemented defenses) UART console low or high risk? Why?

a) Default UART console is a high risk, because it requires physical access to the device(GND,TX,RX). Since it has a baud rate, and parity bit for communication it is easily attackable making it vulnerable. So since it has a high threat and high vulnerability, it makes it high risk.

2. Would this be true for other methods of serial communication such as I2C or a CAN Bus? Why?

a) In case of I2C :
   ref: https://payatu.com/blog/asmita-jha/hardware-attack-surface-i2c
   With the help of FCCID number of the device we can identify the i2c chip in the hardware. through this we can identify the i2c bus(finding the i2c pins). using logic analyzer we can find the sda and scl lines in the i2c. we can connect the pins of the chip to a suitable protocol adapter. finally we can install the framework supported by our adapter.

   ref:
   https://canvas.uw.edu/files/47669787/download?download_frd=1#:~:text=The%20CAN%20bus%20architecture%20is,security%20upfront%20in%20the%20design.
   Points mentioned as it is in the research paper.
   - Multicast Messaging: When a message is sent to the CAN bus, it has no specific destination. Every access point or controller on the bus has access to all messages. Passive attackers could listen in on the communication with ease. Listening could be achieved by tapping into the diagnostics port of the vehicle or inserting a malicious node onto the CAN bus.
   - Lack of Authentication: The typical CAN bus has no authentication process. Nodes do not have a process in place to ensure the message they receive is from a valid source. This

makes it very easy for a masquerade attack, a type of attack where someone pretends to be someone they are not.

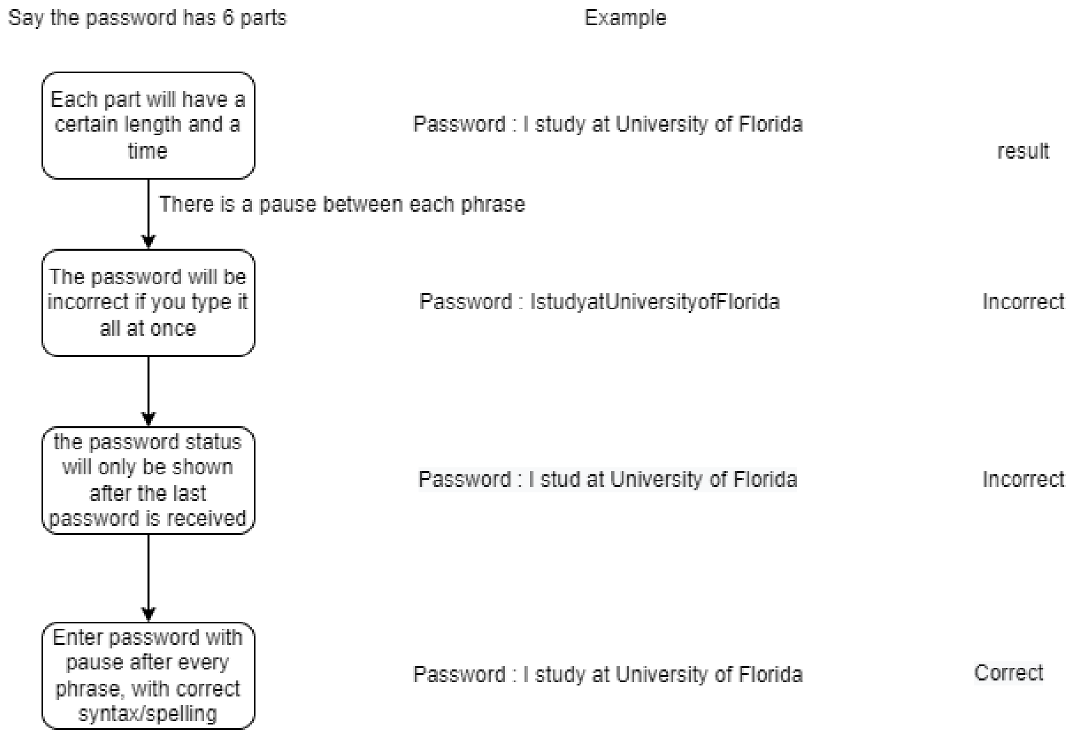
- Lack of Addressing: Nodes typically have no identification address, which allows all (real or malicious) nodes to send or receive information without any verification that the source of information is valid.
- Common Point of Entry: Once an attacker has access to the CAN bus there is no limit to what parameters the attacker can obtain. Usually, one common gateway is used to connect all vehicle CAN bus systems. The diagnostics port in a vehicle for example allows access to all of systems. OBD-II Port.

3. How long would it take to brute force an AES 128 bit password (assuming 1 guess per second)? How many guesses per second would be required to brute force it in 30 minutes?

a) 128 bit, so we know a bit can have 0 or 1. Hence we have 2 power 128 possible keys. Assuming we are brute forcing to find the key, we are making 1 guess per second. That sums upto 60 keys in 1 minute. 3600 in an hour. 86,400 in a day. 31,104,000keys in a year.
Hence it would take 10^37 years to brute force or
2^128/1 sec = 3.4028 10^38 seconds.

b) 30 mins is 1800 seconds. So it would take (2^128)/1800 guesses (i.e) 1.89 e+35.

4. Please create a flowchart that describes the ‘Timed Knock’ process.

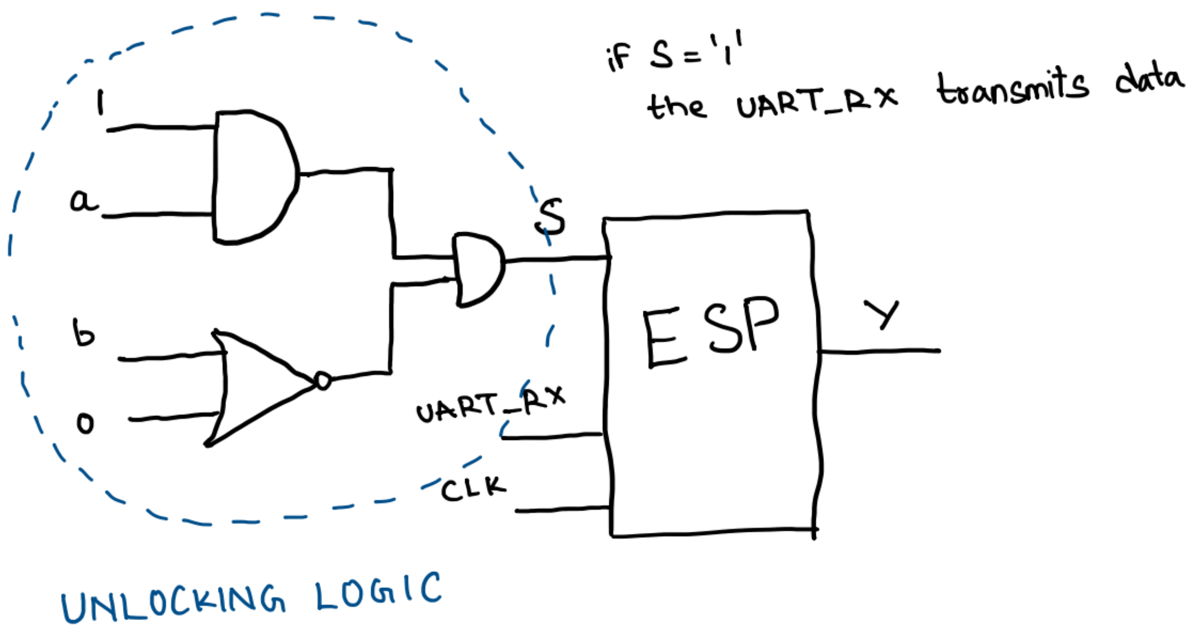

Say the password has 6 parts
Example
Each part will have a certain length and a time
Password : I study at University of Florida
result
There is a pause between each phrase
The password will be incorrect if you type it all at once
Password : IstudyatUniversityofFlorida
Incorrect
the password status will only be shown after the last password is received
Password : I stud at University of Florida
Incorrect
Enter password with pause after every phrase, with correct syntax/spelling
Password : I study at University of Florida
Correct

5. Create a table (or diagram) recording the state of four inputs (A, B, UART_RX, and CLK) that describes the unlocking of a logic locking implementation. This implementation should require A and B switch logic state each clock cycle (With A high at the start and B low) and have a 2 byte password ‘F01D.’ The UART connection uses a single Stop bit with no Parity.

a)

CLK	A	B	UART_RX
0	0	0	
1	1	0	0**11110000**1(F0)
0	0	1	0
1	1	0	0**00011101**1(1D)
0	0	1	0

Unlocking logic of logic locking



Hence the ESP should be programmed in a way that, when S = 1; the UART is enabled to enter the password. Since UART can only enter 8 bits at a time, initially only the first half of the password (i.e) F0 is passed to the esp. Although the esp waits for the other 8 bits. This will happen in the upcoming clock process.

Note: if either half of the password is wrong, the ESP will result in a wrong password. Not to mention the logic locking should be enabled to enter the password. if the logic is wrong, S will have a 0 output which will result in taking null data(w.r.t your design).

Also made mistake in the diagram, UART_RX should be UART_TX and Y should be UART_RX

Closing Thoughts (20 Pts):

1. What is the best (most cost effective) way to secure a UART Console on an IoT Device.

a) Well the most cost effective method to secure the UART console, would be to remove the physical UART connection with the IoT device. That being said, you would lose the accessibility constraint with your IoT device, but definitely secure the device.

Another solution to attain accessibility still, would be to have cooldown time enabled in the IoT device. This will save the follow-up maintenance cost, if we need to connect with the IoT device again.

2. If you were given a task to secure a low cost IoT temperature, humidity, and weather sensor that would be placed on rooftops across a city, how would you go about securing them? Note that there are no ‘right’ answers here, but please describe the reasons behind each choice.

a) I would definitely secure the IoT device with 256 bit AES encryption(128 works too well…). Another possible way to secure the device is by removing the UART connection between the sensors. Or just encapsulate the whole device in a really secure Cage.

Don’t forget to fill out the survey as well! There will be a bit of extra credit if we have 100% participation. It will be fully anonymous and confidential!