EECS 388: Embedded System

Assignment: Understanding Timing SCA Total Points: 2

Background:

In this assignment, you will examine and improve a C program designed for the Intel 8051 architecture, specifically tailored for the AT89C51 microcontroller from Microchip. The program is intended to grant privileged access based on a user-supplied password. However, the existing implementation has overlooked potential side-channel vulnerabilities, particularly related to the timing of password verification. This oversight could potentially allow an attacker to infer the correct password based on the time it takes for the system to respond.

Part 1: Timing Different Password Attempts (Possible points: 1)

Use the provided code to setup UART communication on a 8051 microcontroller development board. Ensure that the UART is properly configured for sending and receiving characters by trying different passwords in UART terminal.

Experiment Procedure:

Try entering five different passwords, varying in length and content as given in the table below. Record the time/cycles it takes from the completion of the password entry to the system's response for each password. Please note that use "." as a null termination character. Please follow the lecture slides and videos to calculate the cycles from logic analyzer.

No	Password Attempt	Rising edge of TR0 TR0 (0->1)	Falling edge of TR0 TR0 (1->0)	Cycle Difference	comments
1.	verysecure				
2.	aaaaaaa				
3.	kaaaaaa				
4.	kuaaaaa				
5.	kuastha				

Report:

Document each password attempt, the exact timing observed, and any other relevant observations in the comment section of above table. Add your comments with the table on analyzing the data to determine if there's a correlation between password characteristics and verification time.

Part 2: Implementing Time-Constant Password Verification (Possible points: 1)

In part 2 we will try to enhance the security of the password verification system by making the processing time for password checks constant, thereby mitigating timing attacks. Investigate methods to implement time-constant comparisons for strings. Understand the concept of side-channel attacks and why they are a risk.

Code Modification:

Modify the provided code to ensure that the password comparison takes the same amount of time regardless of how much of the input matches the actual password. Implement a method that

compares all characters of the input string with the password without short-circuiting upon the first mismatch.

Report:

Test the modified code with the same five passwords from Part 1. Measure and record the time it takes for the system to respond for each password attempt and fill the table again for part 2. Also add detailed explanation of the modifications made to the original code. Finally, compare the timing results from Part 1 and Part 2 and discuss the effectiveness of the modifications.

Deliverables: (Do not attach any screenshot please)

Students will submit a report detailing their findings from both parts of the assignment, including code snippets, timing data, and a discussion of the security implications and improvements.