2023-UCCS1

University of Colorado Colorado Springs

Minhajul Alam Rahat, Vijay Bannerjee, Smita Khapre, Pi Chandramouli, Zainab Olalekan, Nick Waggoner, Sydney Petrehn, Omolade Ikumapayi, Noah Rodgers, Mark Stidd, Jordon Scott, Uchenna Ezeobi, Connor Gurule, Katrina Rosemond

Advised by: Professor Gedare Bloom, PhD

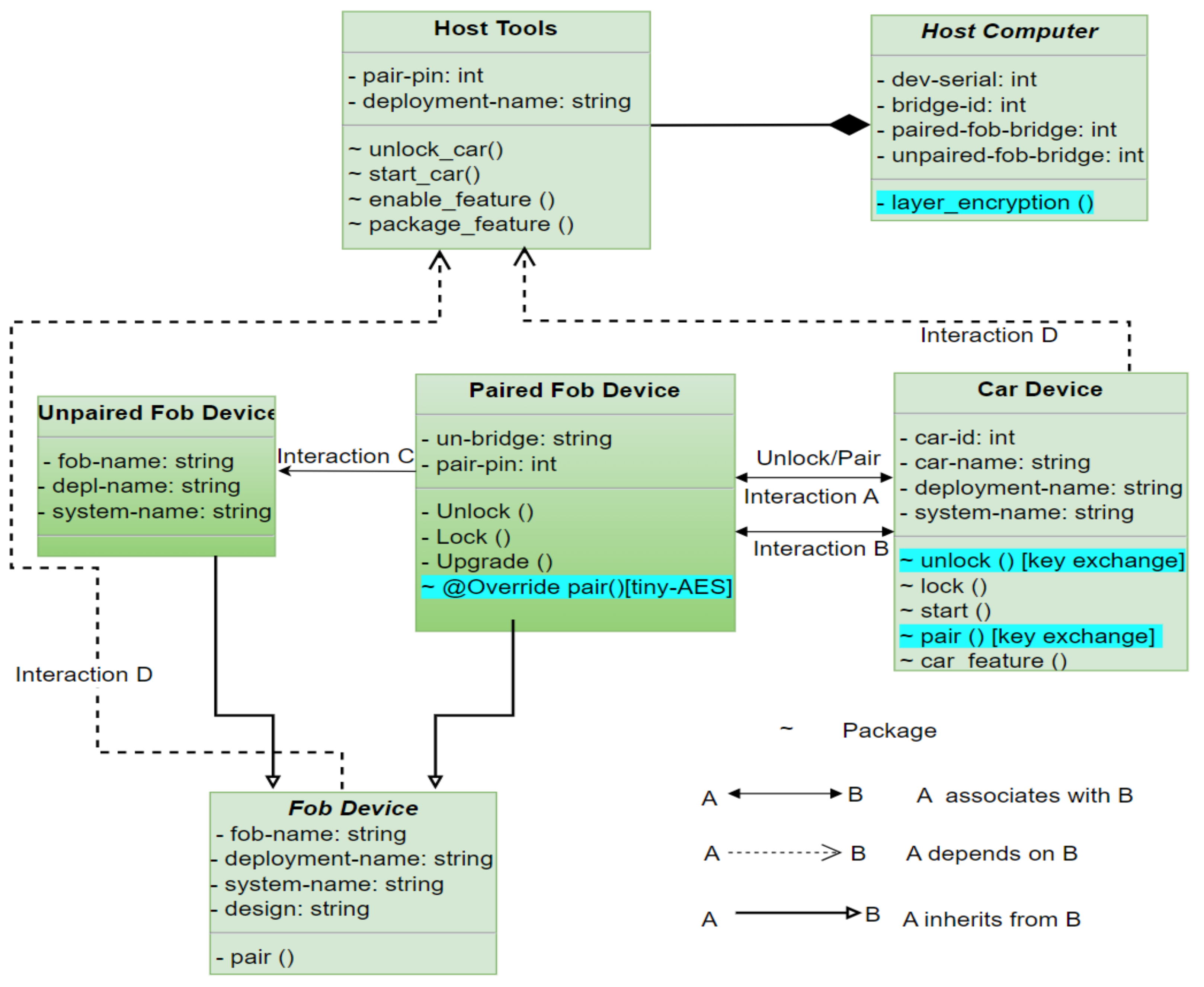
April 20, 2023

Design Overview

A high-level overview of our design is shown in Figure 1, highlighting the security mechanisms used in each step. The security goals are implemented using these security mechanisms:

- HMAC Framework SHA-1 for authentication of paired fob and pairing pin
- Car verifies the message integrity
- Encryption of all secrets stored in the ROM
- Both the car and fob use a symmetric encryption key
    - Pre-shared key for encrypting each unlock message using tiny-AES [1].
    - The car decrypts a fob's message using tiny-AES
    - Verification the manufactured packages

Figure 1: Design Overview



Defensive Highlight

We chose to include a three-way handshake to increase the security of the message and overall system. We chose a symmetric encryption scheme to balance computation tradeoff and security posture. Our security design included several design principles such as: Principle of Economy, Principle of Least Privilege, Principle of Open Design, Principle of Complete Mediation [2].
Our design used HMAC Framework SHA-1 to authenticate message integrity and generate a message digest to include to messages between the Host and Fob. During the pairing operation, the paired fob stored the car ID and feature number. The paired fob shared keys to decrypt the XOR encrypted packaged feature by the Host Tools.

Authentication

Our design was based on Symmetric encryption with the shared key pre-shared during the build operation. This key was used for encrypting and decrypting each message using tiny-AES.
This mechanism could be made better by adding HMAC into the enable feature. Our authentication methods can be made better by adding asymmetric encryption. The random number we generated was pseudorandom.

Figure 2: P-U Message Transfer Protocol

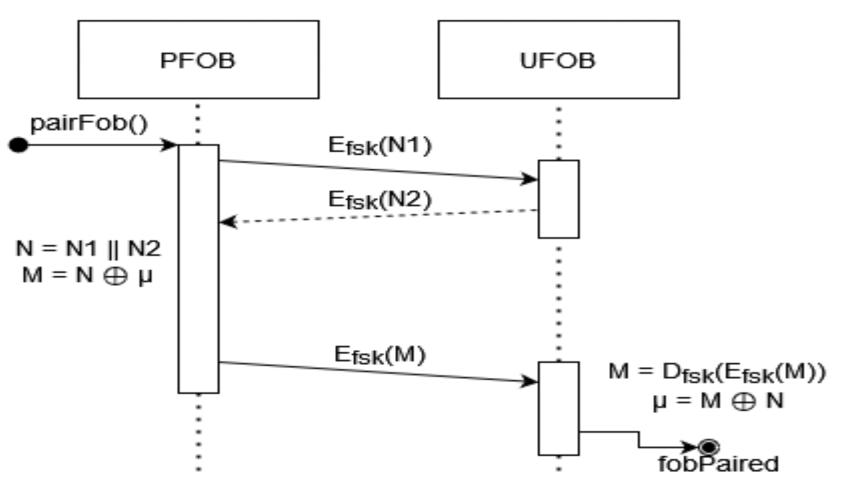


Figure 2 shows the message transfer protocol between the paired fob and unpaired fob. The full description can be found in our design document.

Offensive Highlight

Given additional time, we would execute an attack model consisting of the following steps:

- We will dump the firmware of the design on the board by using tools like binwalk, debugger, etc.
- With access to the binaries, we can extract pins used to unlock the car and other secrets. We will be able to carry out a spoofing attack to mimic a paired fob's functionality. If a spoofed/altered signal match that of a paired fob, the car will unlock.

As an example, one of the design's car firmware compares a pre-calculated sha256 hash with the received fob message buffer, which contains a sha256 hash of the password and car id. The unlock function does not depend on any encrypted message. We can read the calculated hash value using a debugger to extract this hash value from the sha256_test() function in car firmware. Then using host tools, we can send the hash value to the car and unlock the car.

Preventing an attacker from reading a paired fob's binaries and secrets prevents them from mimicking a paired fob's functionality. This can be done by encrypting the code and implementing read-write protection . If the unlock function is encrypted such that the attacker does not have access to the hash of the password, this attack can be avoided.

References

1. Kokke. (2017). tiny-AES-c. GitHub. Retrieved January 20, 2023, from https://github.com/kokke/tiny-AES-c
2. Bishop, M. (2018). Computer Security: Art and Science, 2nd edition. Addison-Wesley Professional PTG. Chapter 14, p. 457.