b01lers

Purdue University

Siddharth Muralee, Muhammad Ibrahim, Jacob White, Bo-Shiun Yen, Ashwin Nambiar, Alan Ma
Advised by: Dr. Antonio Bianchi, Dr. Aravind Machiry
April 24, 2023

Design Overview

Design Philosophy

Define a comprehensive threat model, especially for buffer overflows and side-channels

Avoid over-engineering our protocols, to reduce risk of introducing vulnerabilities

Limit the impact and scope of exploits, even if compromise does occur

Protocol Overview

Car Unlock	**Randomized** challenge-response by car to fob Symmetric key AEAD Encryption using **Ascon**
Fob Pairing	**Salted** and **Hashed** 6-digit pairing PIN **Persistent** 4 sec. timeout on each PIN attempt
Feature Package	**Unique** 32-bit feature password for each car **Salted** and **Hashed** feature stored on car

Attacker Goal / Capability	Brute forcing pairing PIN	Unauthorized car unlock	Unauthorized car features	Unauthorized fob duplication
Access to car	No PIN on car	Symmetric keys on car/fob	Unique feature passwords	No PIN on car
Temporary fob access	Delay	Unique challenge-response	Unique feature passwords	Salt-then-hash pairing PIN
Access to car with features	No PIN on car	Symmetric keys on car/fob	Unique feature passwords	No PIN on car

Attacks

Shared Secrets : Shared secrets allowed reusing fobs on other cars.

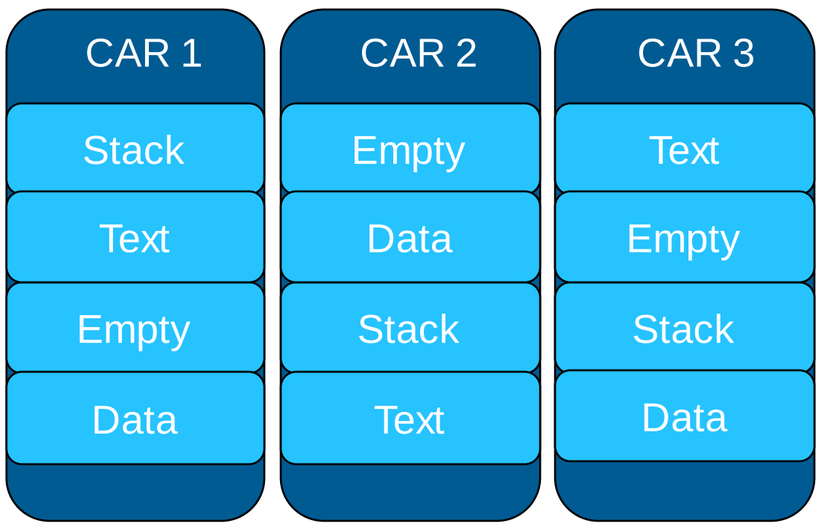
Brute Force : No limits on the number of attempts allowed to brute force the PIN on the fob.

Buffer Overflow : We wrote exploits to leak flags and pins from various teams.

Replay Attacks : Weak or predictable random number generation allowed replay attacks.

Defensive Highlight

EEPROM Layout Randomization (ELR): Our manufacturing process involves the creation of **a randomized EEPROM layout** for each car produced. This security measure ensures that any attacker who gains access to the EEPROM will be unable to discern the location and content of stored data.

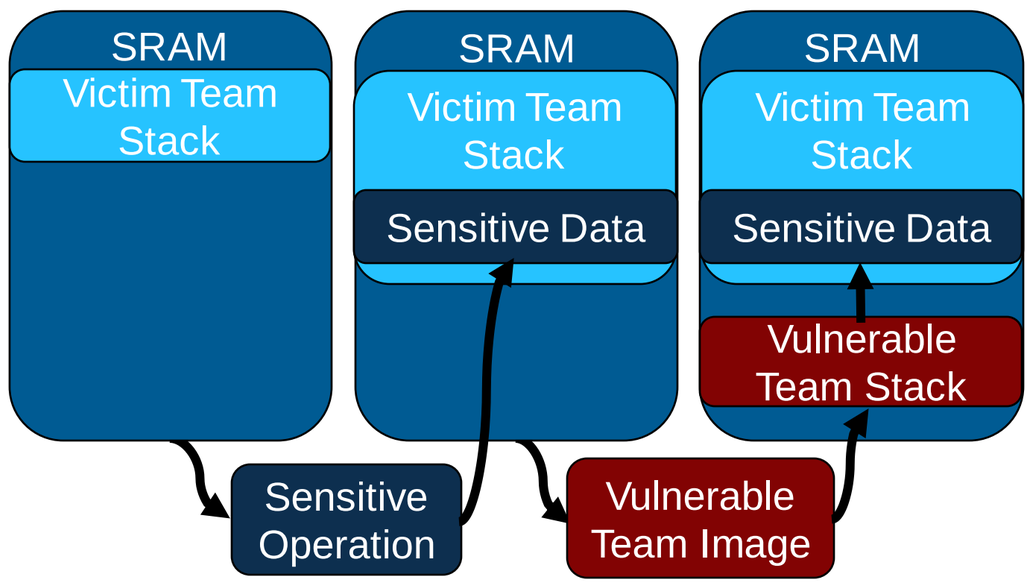


Binary Layout Randomization in car firmware

Binary Layout Randomization (Compile-Time): We believe that modifying our defense strategy to encompass **randomized layout** for other sections, such as the *.text* and *.stack*, would have further strengthened our defenses. This would have resulted in a more formidable challenge for teams seeking to mount successful attacks [3,4].

Offensive Highlight

Stack Leaks: Boards with flags can only run signed firmware images. However, the attacker can flash any correctly signed firmware **at any point** on the car/fob. By flashing a vulnerable and a victim firmware on the car/fob, we leveraged the vulnerable firmware to extract sensitive data left behind from victim firmware images. This attack is shown in the figure below:



By leveraging these leaks, we successfully extracted private keys and pairing pins on the test boards. However, this **attack did not work on keyed boards** since the bootloader clears the SRAM and removes any sensitive data left by the victim team.

References

1. https://ascon.iaik.tugraz.at/
2. NIST SP 800-90A Rev. 1
3. https://css.csail.mit.edu/6.858/2013/projects/an24021-sa23885.pdf
4. https://phrack.org/issues/49/14.html