ZOO_MES

University of Massachusetts Amherst

Advisor
Professor Wayne Burleson

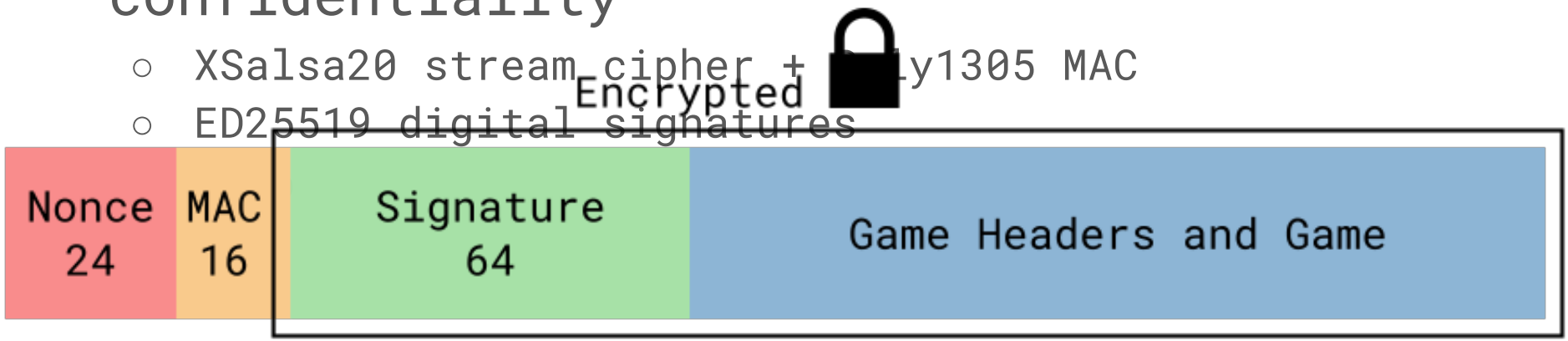
Team
Christopher Barbeau, MS ECE ‘19
Samuel Harris, BS CS ‘20
Ryan Lagasse, BS CSE ‘19
Ryan Lee , BS CS ‘21
Zachary Little, BS CSE ‘19

Outline

- Our Secure Design
- Attack Phase
  - Accomplishments
  - Techniques
- Reflections

Secure Design: Overview (I)

- Library used: Libsodium (+ PyNaCl for Python bindings)
- Sign-Encrypt-MAC for game integrity, confidentiality
  - XSalsa20 stream cipher + Poly1305 MAC
  - ED25519 digital signatures



Secure Design: Overview (II)

Game Table Integrity

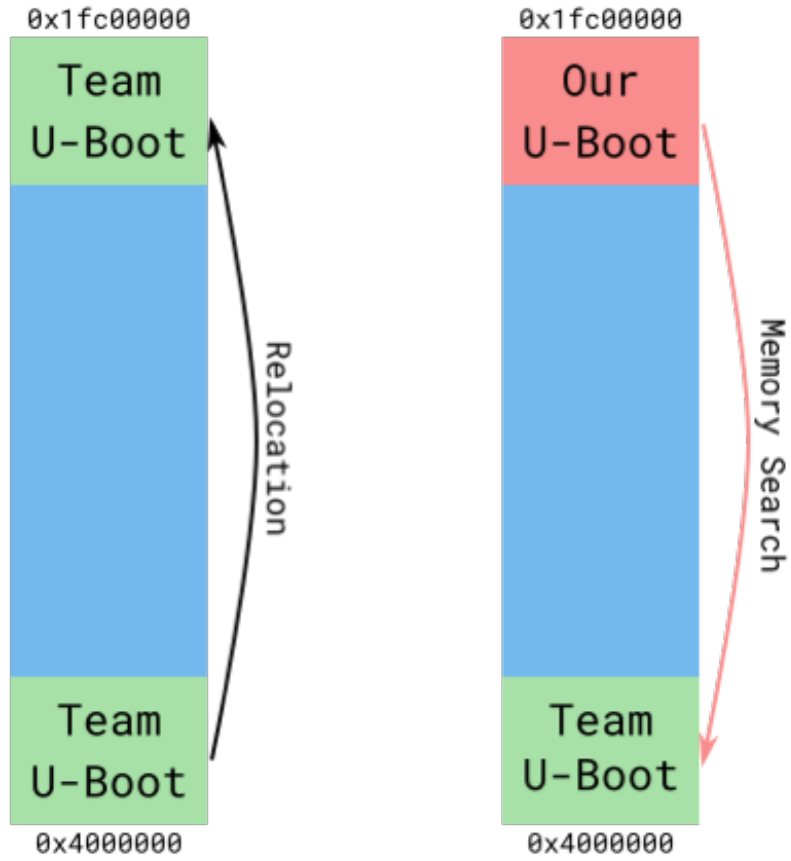
- Checksum each flash record
- Blake2b with secret key

Rollback Flag Corruption Prevention

- If mismatch is detected but game exists, bump version to highest available, then mark uninstalled

Attack #1: U-Boot Memory Reading (II)

- U-Boot initially loads into a specified address (CONFIG_SYS_TEXT_BASE)
- While booting, relocates to the end of memory



Attack #1: U-Boot Memory Reading (II)

- Wrote U-Boot tool to search and dump memory
- Used to capture IP flags
- Built systems ourselves to get addresses of headered keys, extracted keys from locations to get other flags

Remedy

Find alternative for U-Boot for MESH, or

Attack #2: PIN Recovery via User Key

- bcrypt a strong password hasher, properly configured
- <team> directly XOR-ed PIN with part of system key for credentialed access to game keys
- Brute-forcing the modified key was much faster

Remedy

Attack #3: MACs without Signatures

- AES-GCM provides symmetric-key MAC
- Security depends on the key remaining secret
- If discovered, AES-GCM provides guarantees of integrity

Remedy

Use asymmetric public-key signatures as well, instead of relying only on MACs

Reflections: Design Improvements

- Encrypt memory, wipe sensitive info as needed
- Implement MESH in Linux to take advantage of /dev/urandom, ALSR and MMU, stack protectors, etc.
- Generate game keys with user generated info (like PINs) without hard-coding cryptographic info
- Utilize FPGA to make key extraction harder

Reflections: Attack Phase Takeaways

- Dump FPGA contents to extract memory encryption key
- Extract PINs early for more brute-force cracking time
- Identify need for tools, develop them early
  - Memory and flash analysis ✓
  - Side channel and fault injection attacks ✗
- Saw a canary that was optimized-out by

ZOO_MES

University of Massachusetts Amherst

Advisor
Professor Wayne Burleson

Team
Christopher Barbeau, MS ECE ‘19
Samuel Harris, BS CS ‘20
Ryan Lagasse, BS CSE ‘19
Ryan Lee , BS CS ‘21
Zachary Little, BS CSE ‘19