User Controlled Hardware Security Anchors: Evaluation and Designs

Dr David Oswald, Prof Mark Ryan, Prof Flavio Garcia

The University of Birmingham

Industry partners: HP Labs, Yubico

UNIVERSITY OF BIRMINGHAM

RESEARCH INSTITUTE FOR SECURE HARDWARE & EMBEDDED SYSTEMS

User Controlled Hardware Security Anchors: Evaluation and Designs

- **WP1:** Evaluate the security of available security anchors and <u>T</u>rusted <u>E</u>xecution <u>E</u>nvironments (more later)
- **WP2:** Establishing secure channels between TEE and the user through …
  - Auxiliary devices
  - Platform features for secure I/O

User Controlled Hardware Security Anchors: Evaluation and Designs

- **WP3:** Enhancing user authentication
  - Basis: FIDO(2) and U2F
  - Addressing enrollment and revocation
  - Authentication policies (e.g. location, ...)
  - Formal modelling and verification
- **WP4:** Demonstrators
  - TEE implementation
  - Smartphone app
  - Authentication token

WP1: Evaluating the state of TEE security

An overview

Hardware Security Anchors in a nutshell

- Main technologies at present:
  - Trusted Platform Module (separate chip or firmware)
  - Intel Software Guard eXtensions (microcode w/ HW)
  - AMD Platform Security Processor (separate core)
  - ARM TrustZone (software w/ HW support)
  - Apple Secure Enclave Processor (separate core, same die)
  - Google Titan (very recent, separate chip)
- All provide some form of running code or crypto operations in isolation
- Most require cooperation with the silicon/device manufacturer (to different extent)

Intel SGX

- Highest flexibility for the user, can run arbitrary code in "enclaves" – interesting for SW TPM
- Currently "dead" from a security perspective
  - Cache-timing side channels (https://arxiv.org/pdf/1703.06986.pdf, https://arxiv.org/abs/1702.07521, https://arxiv.org/pdf/1702.08719.pdf)
  - MemJam (https://arxiv.org/abs/1711.08002)
  - Spectre and Meltdown variants (Foreshadow, see previous talk ☺)
  - *Software-driven faults (akin to RowHammer)?*

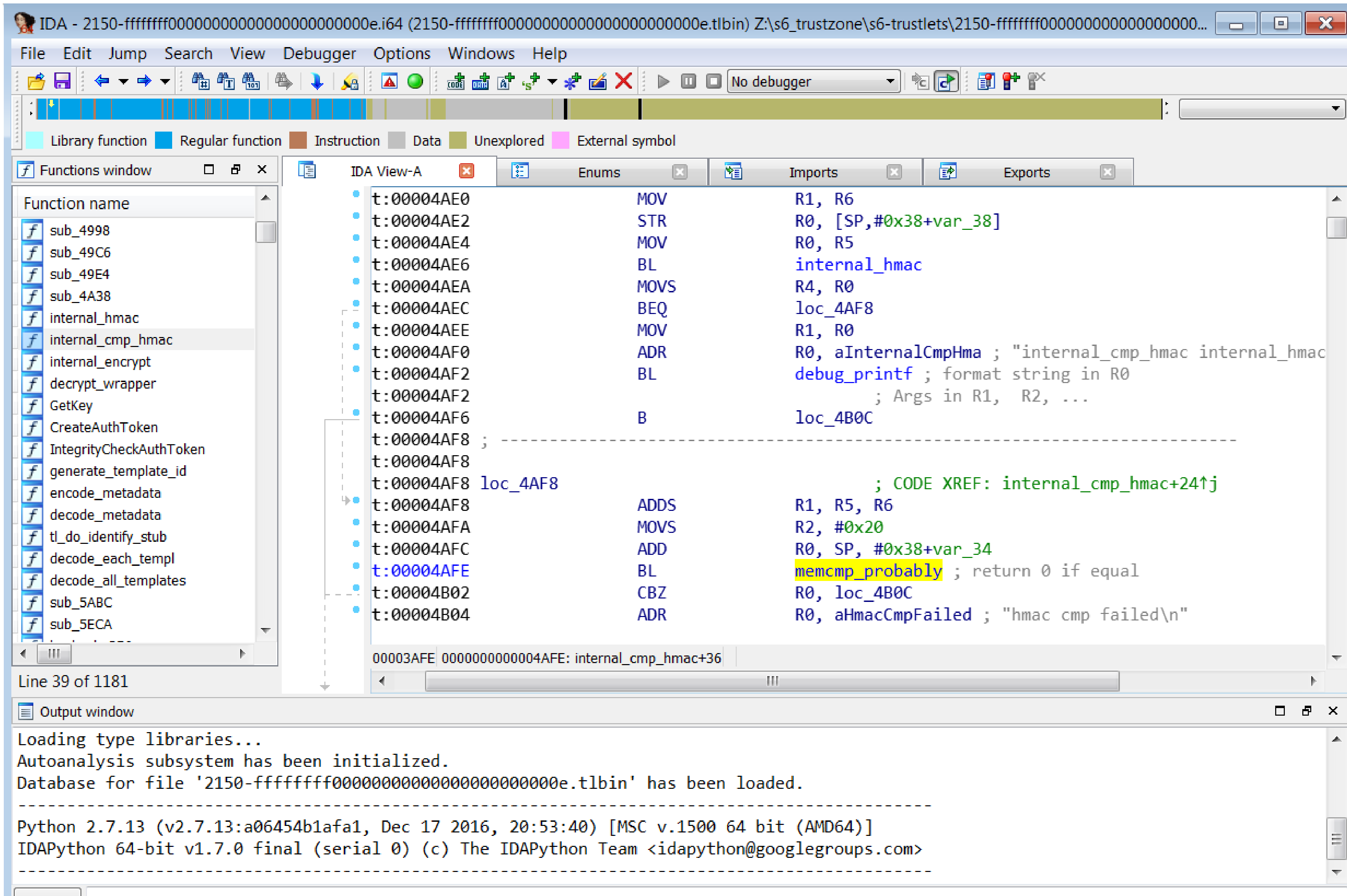
ARM TrustZone

```
                 DCB 0, 0, 0
aStartOfRawMeta DCB "Start of Raw Metallica OTP Collected Data",0xA,0
                                         ; DATA XREF: sub_30CE6+C↑o
                DCB 0
aBoot0Data      DCB "Boot 0 Data",0      ; DATA XREF: sub_30CE6+20↑o
aBoot1Data      DCB "Boot 1 Data",0      ; DATA XREF: sub_30CE6+32↑o
aSectorData     DCB "Sector Data",0      ; DATA XREF: sub_30CE6+48↑o
aEndOfRawMetall DCB "End of Raw Metallica OTP Collected Data",0xA,0
                                         ; DATA XREF: sub_30CE6+58↑o
```

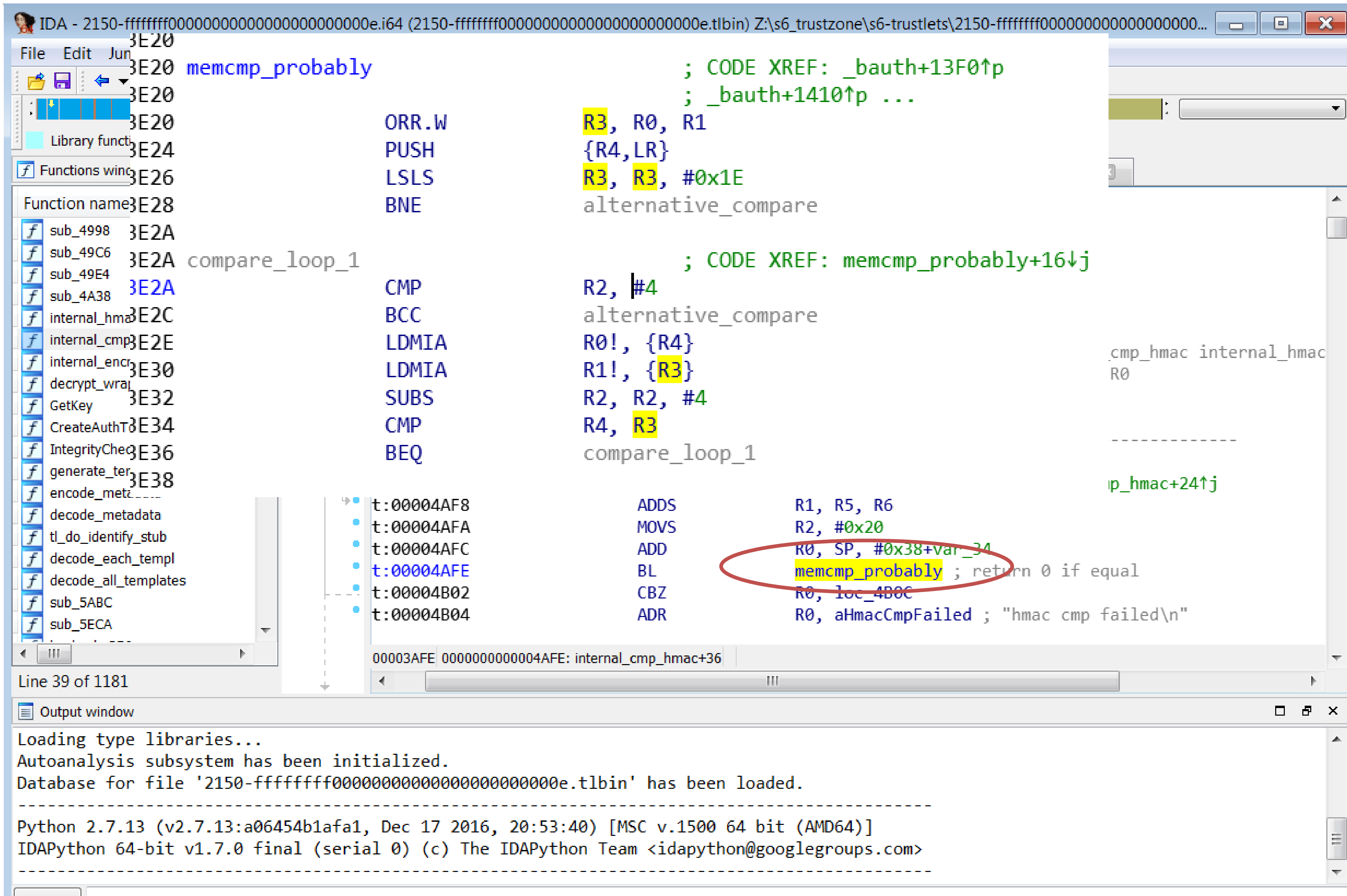
Previous attacks on Samsung TZ

- Long history of SW attacks on TZ,
  https://googleprojectzero.blogspot.co.uk/2017/07/trust-issues-exploiting-trustzone-tees.html
- Up to Galaxy S7, attacker can roll back to old (vulnerable) versions of trustlets
- Beniamini discovered buffer overflow in OTP trustlet, allowing code execution in the context of this trustlet
- Lapid & Wool showed that KeyMaster Key Encryption Key can be extracted via OTP vuln or cache-timing side channel

Example: fingerprint matching trustlet



Example: fingerprint matching trustlet

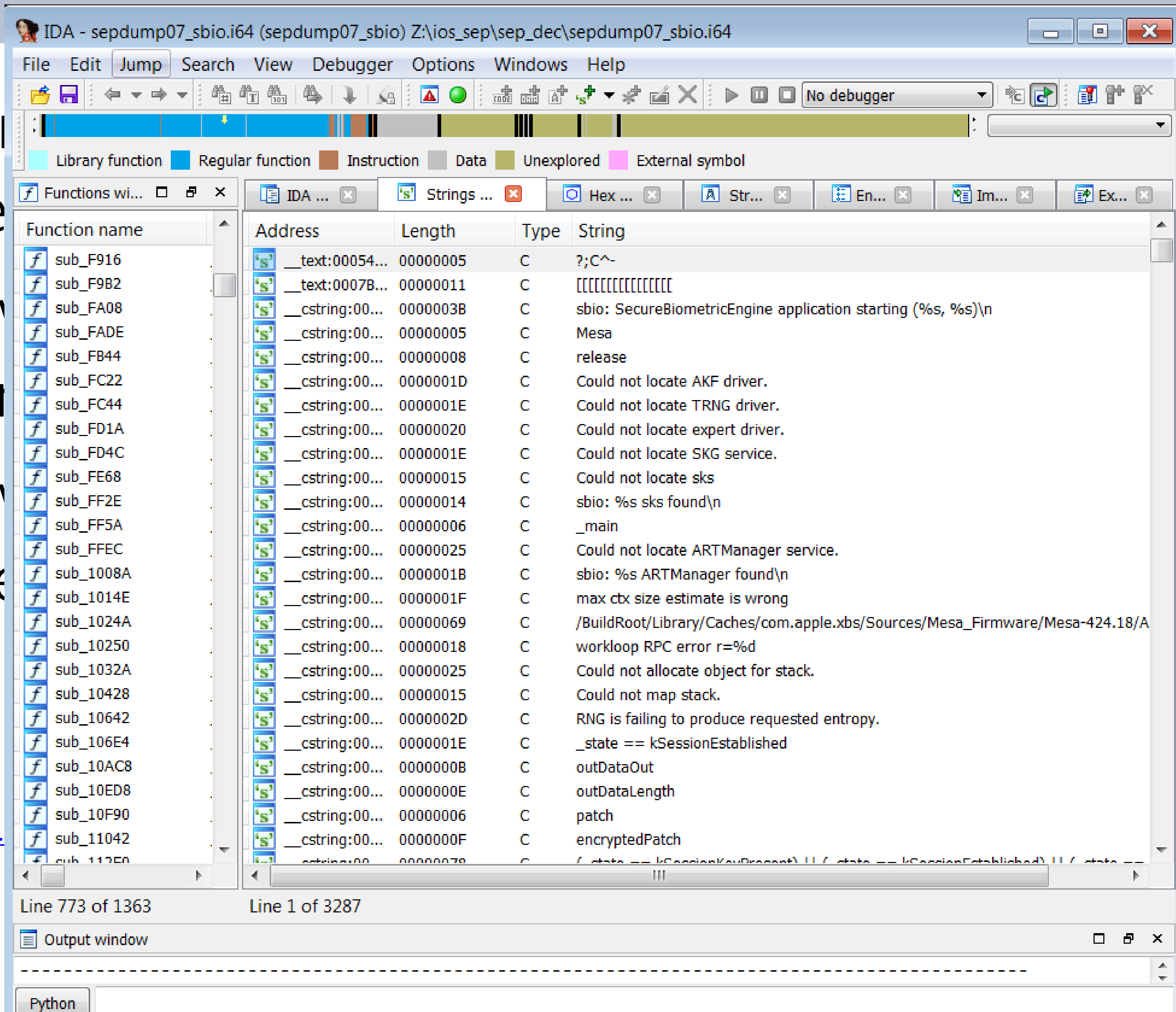


Apple SEP

```
__const:0004BB60 ; Segment type: Pure data
__const:0004BB60                 AREA __const, DATA, ALIGN=4
__const:0004BB60                 ; ORG 0x4BB60
__const:0004BB60 aDerivedKey     DCB "derived_key",0     ; DATA XREF: sub_
__const:0004BB6C aSepDerivedKey  DCB "SEP derived_key",0
__const:0004BB7C aSeWhat         DCB "SE what?",0        ; DATA XREF: sub_
__const:0004BB85                 ALIGN 4
__const:0004BB88                 DCD sub_14716+1
```

Understanding Apple SEP

- OS a
  in de
- Firm
  iPho
- Firm
  loade



1 https://www. ... pdf

```
- open file "sepdump07_sbio"

  offset   num  description [bits.endian.size]
  --------------------------------------------
  0007b1f0 874  SHA256 Hash constant words K (0x428a2f98) [32.le.256]
  000bc5cc 536  CRC-16-IBM maxim/usb [crc16.0xa001 lenorev 1.512]
  000bc5cc 529  CRC-16-IBM maxim/usb [crc16.0x8005 le rev int_min.512]
  000bc7cc 648  CRC-32-IEEE 802.3 [crc32.0xedb88320 lenorev 1.1024]
  000bc7cc 641  CRC-32-IEEE 802.3 [crc32.0x04c11db7 le rev int_min.1024]
  000bd20c 897  Rijndael Te0 (0xc66363a5U) [32.be.1024]
  000bd60c 906  Rijndael Td0 (0x51f4a750U) [32.be.1024]
  000bda0c 894  AES Rijndael S / ARIA S1 [..256]
  000bdb0c 895  AES Rijndael Si / ARIA X1 [..256]
  000bdc30 878  Hash constant words K for SHA-384 and SHA-512 [64.le.640]
  000bdeb0 1036 SHA1 / SHA0 / RIPEMD-160 initialization [32.le.20&]
  000bdeb0 2402 Lucifer (outerbridge) DFLTKY [..16]
  000bdebc 2053 RIPEMD-128 InitState [32.le.16&]
  000bdee4 1030 SHA256 [32.le.288&]
  000bdee4 876  SHA256 Initial hash value H (0x6a09e667UL) [32.le.32&]
  000bdee8 2364 Crypton kp [32.le.16]
```

UNIVERSITY OF BIRMINGHAM

Thanks for your attention!

Questions?

d.f.oswald@bham.ac.uk