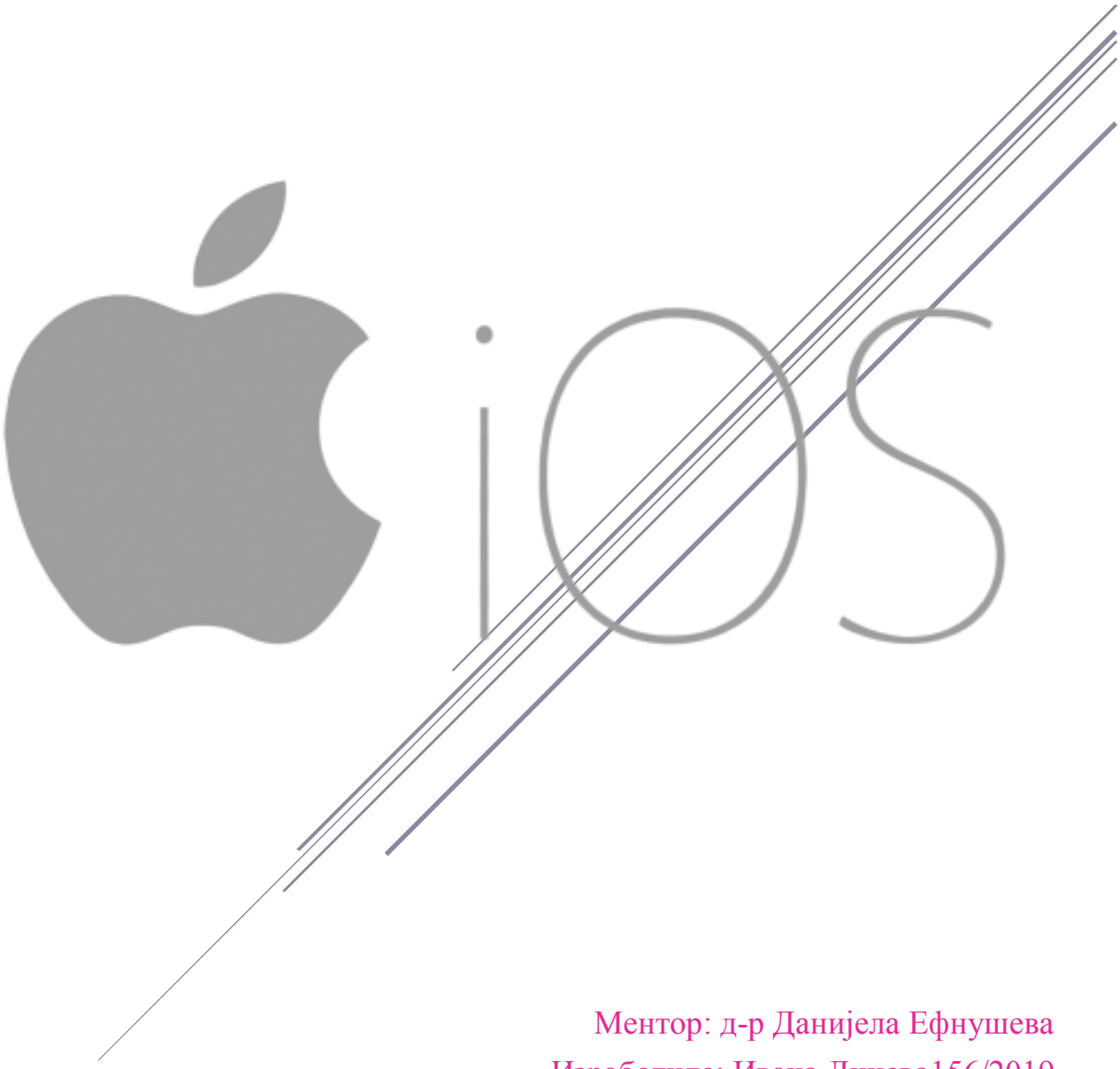
БЕЗБЕДНОСТ ВО IOS

Семинарска работа по предметот
Безбедност и заштита на компјутерско комуникациски системи и мрежи

ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И ИНФОРМАЦИСКИ ТЕХНОЛОГИИ



Ментор: д-р Данијела Ефнушева
Изработиле: Ивана Динева156/2019
Стефан Стојковиќ 9/2019

СОДРЖИНА

1.Вовед во безбедноста на платформата на Apple 3
2. Hardware security overview 3
2.1. Преглед на безбедноста на хардверот 3
3. Secure Enclave 4
3.1. Преглед 4
3.2. Процесор за Secure Enclave 5
3.3. Engine за заштита на меморијата 6
3.4. Безбедносен ROM за вклучување на Secure enclave 7
3.5. Генератор на вистински случаен број 8
3.6. Коренски криптографски клуч 8
3.7. AES engine во Secure Enclave 9
3.8. AES engine 10
3.9. Забрзувач на јавен клуч 11
3.10. Безбедна неиспарлива меморија 11
4. Процес на стартување на iOS уреди 12
5. Безбедни ажурирања на софтверот 14
5.1. Преглед 14
5.2. Персонализиран процес на ажурирање 15
6. Преглед на шифрирање и заштита на податоци 16
7. Безбедност на апликацијата 18
8. Безбедност на сервиси 19
9. Keychain services 20
9.1. Keychains 21
9.2. Keychain items 21
10. Спречување на небезбедни мрежни врски 22
11. iOS Jailbreak 23
11.1. Што е Jailbreak? 23
11.2. Видови џеилбрејкови 23
11.3. Како функционира џеилбрејкот? 24
11.4. Џејлбрејкот може да отвори и многу безбедносни дупки 25

12. Некои попознати експлоатации на ios 25
12.1. Pwnage/pwnage2 25
12.2. 24kpwn 27
13. Заклучок 27
Користена литература 28

1. ВОВЕД ВО БЕЗБЕДНОСТА НА ПЛАТФОРМАТА НА APPLE

Секој уред на Apple комбинира хардвер, софтвер и услуги дизајнирани да работат заедно за максимална безбедност и транспарентно корисничко искуство каде крајната цел е заштита на приватноста на корисниците. На пример, силиконот дизајниран од Apple и безбедносниот хардвер ги напојуваат критичните безбедносни способности. Додека пак, софтверската заштита работи за да ги заштити оперативниот систем и third-party апликации. Конечно, услугите обезбедуваат механизам за сигурно и навремено ажурирање на софтверот, создаваат екосистем на заштитени апликации и олеснуваат безбедни комуникации и плаќања. Како резултат на тоа, уредите на Apple го штитат не само уредот и неговите податоци, туку и целиот екосистем, вклучувајќи сè што корисниците прават локално, на мрежи и со клучните интернет услуги.

Клучните безбедносни способности, како што е шифрирањето на уредот базирано на хардвер, не може да се оневозможат по грешка. Другите функции, како што се Touch ID и Face ID, го подобруваат корисничкото искуство така што го прават поедноставно и поинтуитивно обезбедувањето на уредот. И бидејќи многу од овие функции се овозможени по default, корисниците не треба да вршат обемни конфигурации.

Во наредните страници ќе бидат приложени детали за тоа како безбедносната технологија и функции се имплементирани во iOS.

2. HARDWARE SECURITY OVERVIEW

2.1. ПРЕГЛЕД НА БЕЗБЕДНОСТА НА ХАРДВЕРОТ

За софтверот да биде безбеден, тој мора да се потпира на хардвер кој има вградена безбедност. Затоа уредите на Apple со iOS имаат безбедносни способности дизајнирани во силиконот. Овие способности вклучуваат процесор кој ги напојува безбедносните одлики на системот, како и дополнителен силикон кој е посветен на безбедносните функции.

Хардверот фокусиран на безбедноста го следи принципот на поддршка на ограничени и дискретно дефинирани функции со цел да се минимизира површината на нападот. Таквите компоненти вклучуваат boot ROM, кој формира хардверски корен на доверба за безбедно вклучување на уредот, AES (Advanced Encryption Standard) engines за ефикасно и безбедно шифрирање и дешифрирање и Secure Enclave. Secure Enclave е систем на чип (SoC) кој е вклучен на сите неодамнешни iPhone уреди. Самиот Secure Enclave го следи истиот принцип на дизајн како и SoC, содржи сопствен boot ROM и AES engines. Secure Enclave, исто така, обезбедува основа за безбедно генерирање и складирање на клучевите неопходни за шифрирање на зачуваните податоци, и ги штити и проценува биометриските податоци за Touch ID и Face ID.

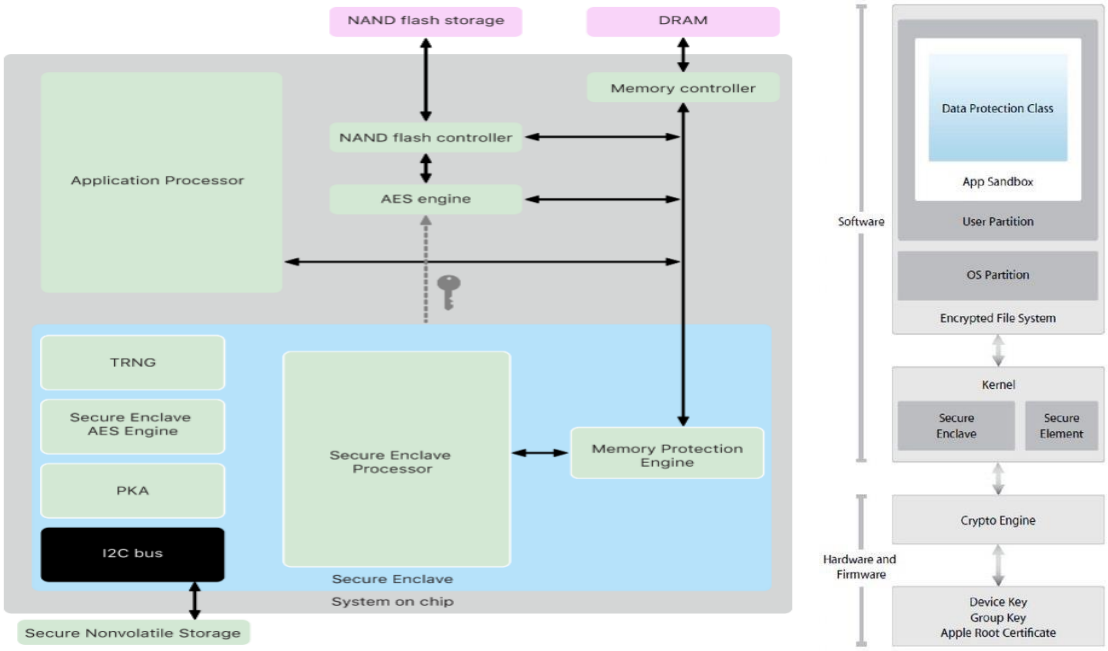
Шифрирањето на зачуваните податоци мора да биде брзо и ефикасно. Шифрирањето не смее да ги открие податоците или клучевите што ги користи. AES хардверскиот engine го решава овој проблем со извршување на брзо шифрирање и дешифрирање во линија (in-line) додека датотеките се пишуваат или читаат. Посебен канал од Secure Enclave го обезбедува потребниот материјал за приклучување на AES engine-от без да ги изложува овие информации на Апликацискиот процесор или целокупниот оперативен систем.

3. SECURE ENCLAVE

3.1. ПРЕГЛЕД

Secure Enclave е безбедносен потсистем интегриран во SoC. Secure Enclave е изолиран од главниот процесор за да обезбеди дополнителен слој на безбедност и е дизајниран да ги одржува безбедни чувствителните податоци на корисникот дури и кога јадрото на Апликацискиот процесор е компромитиран. Се состои од boot ROM за вклучување за да воспостави хардверски корен на доверба, AES engine за ефикасни и безбедни криптографски операции, и заштитена меморија. Иако Secure Enclave не вклучува складирање, има механизам за безбедно складирање на информациите во меморија одвоена

од NAND флеш-меморијата што ја користат Апликацискиот процесор и оперативниот систем.



Слика 1: Secure Enclave

3.2. ПРОЦЕСОР ЗА SECURE ENCLAVE

Процесорот во Secure Enclave ја обезбедува главната компјутерска моќ на Secure Enclave. За да се обезбеди најсилна изолација, процесорот е наменет исклучиво за употреба во Secure Enclave. Ова помага да се спречат напади на страничните канали кои зависат од злонамерен софтвер што го споделува истото јадро за извршување како и целниот софтвер што е нападнат.

Secure Enclave процесорот е дизајниран да работи ефикасно со помала брзина на clock сигналот што помага да се заштити од напади со clock и напојување. Поновите верзии од овој процесор, вклучуваат engine за заштита на меморијата и шифрирана меморија која не може да се репродуцира, генератор на случајни броеви и сопствен AES engine.

3.3. ENGINE ЗА ЗАШТИТА НА МЕМОРИЈАТА

Secure Enclave работи во специјален регион на DRAM меморијата на уредот. Повеќе слоеви на заштита ја изолираат заштитената меморија од Апликацискиот процесор.

Кога уредот ќе се вклучи, boot ROM-от за вклучување на Secure Enclave генерира случаен ефемерен клуч кој ќе биде искористен од engine-от за заштита на меморијата (Memory Protection Engine). Секогаш кога Secure Enclave пишува во својот мемориски регион, Memory Protection Engine го шифрира блокот на меморија користејќи AES и пресметува ознака за автентикација на пораки базирана на шифри (Cipher-based Memory Authentication Code) за меморијата. Memory Protection Engine ја складира ознаката за автентикација заедно со шифрираната меморија. Кога Secure Enclave ја чита меморијата, Memory Protection Engine ја потврдува ознаката за автентикација. Ако ознаката за автентикација се совпаѓа, Memory Protection Engine го дешифрира блокот на меморијата. Ако ознаката не се совпаѓа, се испраќа сигнал за грешка до Secure Enclave. По грешка при автентикација на меморијата, Secure Enclave престанува да прифаќа барања додека системот не се рестартира.

Почнувајќи со Apple A11 и S4 SoCs, Memory Protection Engine додава заштита од повторување на меморијата со која располага Secure Enclave. За да помогне да се спречи повторување на безбедносни критични податоци, Memory Protection Engine складира “nonce” за блокот на меморијата заедно со ознаката за автентикација. Nonce се користи како дополнителна заштита на ознаката за автентикација. Нонсите за сите мемориски блокови се заштитени со користење на дрво за интегритет вкоренето во специјален SRAM во рамките на Secure Enclave. За пишување, Memory Protection Engine ги ажурира нонсите и секое ниво на дрвото за интегритет. За читање, моторот за заштита на меморијата ја проверува нонсата и секое ниво на дрвото за интегритет. При неусогласеност на нонсите се постапува слично како и несовпаѓањата на ознаките за автентикација.

Во поновите верзии, Memory Protection Engine поддржува два ефемерни клучеви за заштита на меморијата. Првиот се користи за приватни податоци во Secure Enclave, а вториот се користи за податоци споделени со Secure Neural Engine.

Memory Protection Engine работи in-line и транспарентно во однос на Secure Enclave. Secure Enclave чита и запишува меморија како да е обична нешифрирана DRAM, додека набљудувачот надвор од Secure Enclave ја гледа само шифрираната и автентицирана верзија на меморијата.

3.4. БЕЗБЕДНОСЕН ROM ЗА ВКЛУЧУВАЊЕ НА SECURE ENCLAVE

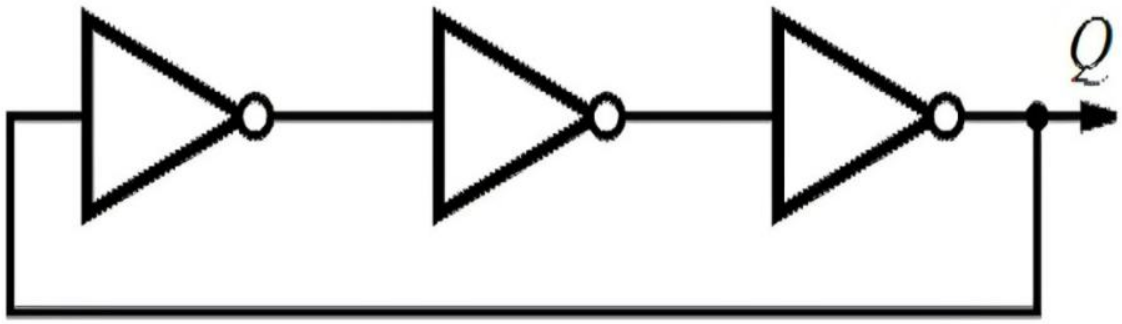
Secure Enclave вклучува ROM за стартување на овој чип. Како и ROM-от за стартување на Апликацискиот Процесор, ROM-от за Secure Enclave е непроменлив код што го воспоставува хардверскиот корен на доверба.

При стартување на системот, iBoot доделува регион од меморијата на Secure Enclave. Пред употреба на меморијата, Secure Enclave Boot ROM-от го иницијализира Memory Protection Engine за да обезбеди криптографска заштита на заштитената меморија од Secure Enclave.

Апликацискиот Процесор потоа ја испраќа сликата на sepOS (Secure Enclave Processor Operating System) до ROM-от за стартување на Secure Enclave. По копирањето на сликата на sepOS во заштитената меморија резервирана за Secure Enclave, boot ROM-от го проверува криптографскиот хеш и потписот на сликата за да потврди дека sepOS е овластен да работи на уредот. Ако сликата на sepOS е правилно потпишана да работи на уредот, ROM-от за стартување на Secure Enclave му дозволува на sepOS да превземе контрола. Доколку потписот не е валиден, ROM-от е дизајниран да спречи понатамошна употреба на Secure Enclave до следното ресетирање на чипот.

3.5. ГЕНЕРАТОР НА ВИСТИНСКИ СЛУЧАЕН БРОЈ

Генераторот на вистински случаен број (True Random Number Generator) се користи за генерирање безбедносни случајни податоци. Secure Enclave го користи TRNG секогаш кога генерира случаен криптографски клуч, семе од случаен клуч или друга ентропија. TRNG се заснова на повеќе прстенести осцилатори кои потоа се обработуваат со алгоритмот CTR_DRBG.



Слика 2: Генератор на вистински случаен број

3.6. КОРЕНСКИ КРИПТОГРАФСКИ КЛУЧ

Secure Enclave вклучува единствен коренски криптографски клуч (UID). UID е единствен за секој поединечен уред и не е поврзан со кој било друг идентификатор на уредот.

Случајно генериран UID се создава за време на производството на чипот. Почнувајќи од A9 SoCs, UID се генерира од Secure Enclave TRNG за време на производството и се запишува користејќи софтверски процес што целосно работи во Secure Enclave. Овој процес го штити UID од тоа да биде видлив надвор од уредот за време на производството и затоа

не е достапен за пристап или складирање од страна на Apple или кој било од неговите добавувачи.

sepOS го користи UID за заштита на тајни кои се специфични за уредот. UID дозволува податоците да бидат криптографски врзани за одреден уред. На пример, хиерархијата на клучеви што го штити датотечниот систем го вклучува UID, така што ако внатрешната меморија на SSD е физички преместена од еден уред на друг, датотеките се недостапни. Други заштитени тајни специфични за уредот вклучуваат податоци за Touch ID или Face ID.

Secure Enclave има и групен ID (GID), која е заедничка за сите уреди што користат даден SoC (на пример, сите уреди што користат Apple A14 SoC го делат истиот GID).

3.7. AES ENGINE BO SECURE ENCLAVE

Secure Enclave AES Engine е хардверски блок што се користи за изведување симетрична криптографија базирана на AES шифрирањето. AES Engine е дизајниран да се спротивстави на изложување на приватни информации со користење на тајминг и статичка анализа на моќност (Static Power Analysis). Почнувајќи од A9 SoC, AES Engine вклучува и контрамерки за динамичка анализа на моќност (Dynamic Power Analysis).

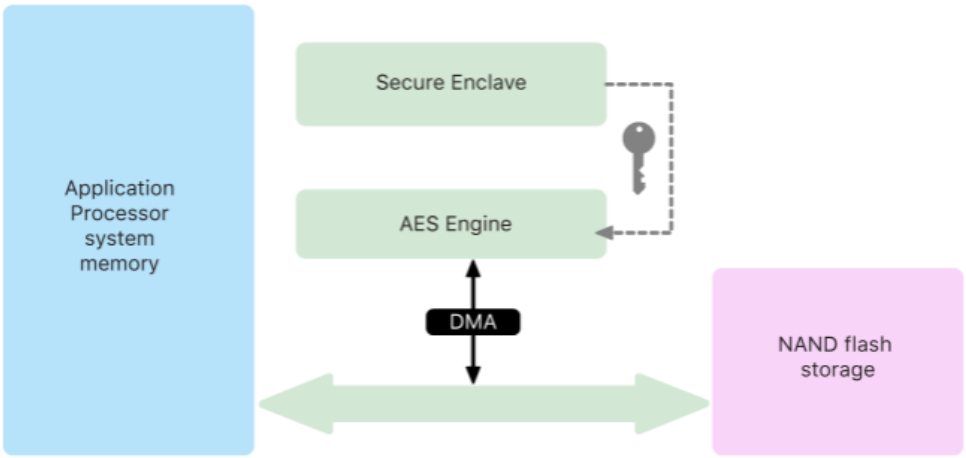
AES Engine поддржува хардверски и софтверски клучеви. Хардверските клучеви се изведени од UID или GID на Secure Enclave. Овие клучеви остануваат во AES Engine и не се видливи дури и за sepOS. Иако софтверот може да бара операции за шифрирање и дешифрирање со хардверски клучеви, тој не може да ги извлече клучевите.

3.8. AES ENGINE

Секој уред на Apple со Secure Enclave има AES256 crypto-engine („AES Engine“) вграден во патеката за директен пристап до меморија (DMA) помеѓу NAND флеш меморијата и главната системска меморија, што го прави шифрирањето на датотеки високо ефикасно. На A9 или понови процесори од серијата A, потсистемот за складирање на флеш е на изолирана магистрала на која и е дозволен пристап само до меморијата што содржи кориснички податоци преку DMA crypto-engine-от.

При вклучување, sepOS генерира ефемерен клуч за завиткување со помош на TRNG. Secure Enclave го пренесува овој клуч до AES engine-от користејќи посебни жици, дизајнирани да спречат пристап до него од кој било софтвер надвор од Secure Enclave. sepOS потоа може да го користи ефемерниот клуч за завиткување за да ги завитка клучевите на датотеките што ќе ги користи драјверот на датотечниот систем на Апликацискиот процесор. Кога драјверот на датотечниот систем чита или запишува датотека, го испраќа завитканиот клуч до AES Engine, кој го одвиткува клучот. AES Engine никогаш не го изложува неотпакуваниот клуч на софтверот.

Забелешка: AES Engine е посебна компонента и од Secure Enclave и Secure Enclave AES Engine, но неговата работа е тесно поврзана со Secure Enclave, како што е прикажано подолу.



Слика 3: AES мотор

3.9. ЗАБРЗУВАЧ НА ЈАВЕН КЛУЧ

Забрзувачот на јавен клуч (Public Key Accelerator) е хардверски блок што се користи за извршување операции на асиметрична криптографија. PKA поддржува RSA (Reed Solomon Algorithm) и ECC (Elliptic Curve Cryptography) алгоритми за потпишување и шифрирање.

PKA поддржува софтверски и хардверски клучеви. Хардверските клучеви се изведени од UID или GID на безбедната енклава. Овие клучеви остануваат во рамките на PKA и не се видливи дури и за sepOS.

На Apple A10 и подоцнежните SoCs, PKA поддржува клучеви поврзани со оперативниот систем, исто така наречени Заштита на запечатен клуч (Sealed Key Protection). Овие клучеви се генерираат со комбинација на UID на уредот и хешот на sepOS што работи на уредот. Хешот е обезбеден од Secure Enclave Boot ROM или од Secure Enclave Boot Monitor на Apple A13 и понови SoCs. Овие клучеви се користат и за потврдување на верзијата на sepOS кога се поднесуваат барања до одредени услуги на Apple и се користат и за подобрување на безбедноста на податоците заштитени со лозинка со тоа што помагаат да се спречи пристапот до материјалот за клучеви доколку се направат критични промени во системот без овластување од корисникот.

3.10. БЕЗБЕДНА НЕИСПАРЛИВА МЕМОРИЈА

Secure Enclave е опремена со безбеден неиспарлив мемориски чип. Овој чип е поврзан со Secure Enclave користејќи посебна I2C магистрала, така што до неа може да пристапи само Secure Enclave. Сите клучеви за шифрирање на кориснички податоци се вкоренети во ентропијата складирана во овој чип.

4. ПРОЦЕС НА СТАРТУВАЊЕ НА IOS УРЕДИ

Секој чекор од процесот на стартување содржи компоненти кои се криптографски потпишани од Apple за да овозможат проверка на интегритетот, така што стартување ќе се одвива само по потврдување на синџирот на доверба.

Кога iOS уредот е вклучен, неговиот Апликациски Процесор веднаш извршува код од меморија наведена како Boot ROM. Овој непроменлив код, познат како хардверски корен на доверба, е поставен за време на создавањето на чипот и е имплицитно веродостоен. Кодот во Boot ROM содржи Apple Root certificate authority (CA) public key - користен за да потврди дека iBoot е потпишан од Apple пред да дозволи да се вчита. Ова е прв чекор во синџирот на доверба, во кој секој чекор проверува дали следниот е потпишан од Apple. Кога iBoot ги завршува своите задачи, го верификува и стартува iOS кернелот. За уреди со A9 или претходен процесор на A-серија, дополнителна фаза на Low Level Bootloader е вчитана и заверена од Boot ROM кој пак самиот го стартува и го потврдува iBoot.

```
336  static bool
337  load_selected_image(struct image_info *loaded_image, u_int32_t type,
     void *load_address, size_t load_length, u_int32_t boot_flag)
338  {
339      /* image epoch must be greater than or equal to the SecureROM's
         epoch */
340      loaded_image->imageOptions |= IMAGE_OPTION_GREATER_EPOCH;
341
342      /* image decode library make use of this information to do various
         validation on image */
343      loaded_image->imageOptions |= IMAGE_OPTION_NEW_TRUST_CHAIN;
344
345      /* if test mode isn't set, we require the image be trusted,
         regardless of the security fuse */
346      if (!(boot_flag & BOOT_FLAG_TEST_MODE))
347          loaded_image->imageOptions |= IMAGE_OPTION_REQUIRE_TRUST;
348
349      /* validate the boot image */
350      if (image_load(loaded_image, &type, 1, NULL, &load_address, &
         load_length)) {
351          dprintf(DEBUG_INFO, "image load failed\n");
352          return false;
353      }
354
355      return true;
356  }
357
```

Слика 4: Дел од кодот во BootRom на iOS9 каде се валидира LLB(BootROM and iBoot source codes of iOS 9 leak online (yalujailbreak.net))

```
259     /* find iBoot and boot it */
260     boot_iboot();
261
262     /* failed to load system */
263     dprintf(DEBUG_CRITICAL, "do_boot: failed to find anything to load\n");
264
265     /* re-init security after failed iBoot load */
266     security_init(false);
267
268     dprintf(DEBUG_CRITICAL, "LLB done, failed to boot asking for DFU...\n");
269     main_dfu();
270 }
271
```

Слика 5: Дел од кодот на LLB каде се стартува iBoot(BootROM and iBoot source codes of iOS 9 leak online (yalujailbreak.net))

```
ROM:84000000 loc_84000000                            ; CODE XREF: ROM:84000078j
ROM:84000000                                         ; DATA XREF: ROM:_reset o ...
ROM:84000000                 B               _reset
ROM:84000004 ; ---------------------------------------------------------------------------
ROM:84000004                 LDR             PC, =_undef
ROM:84000008 ; ---------------------------------------------------------------------------
ROM:84000008                 LDR             PC, =_swi
ROM:8400000C ; ---------------------------------------------------------------------------
ROM:8400000C                 LDR             PC, =_prefabt
ROM:84000010 ; ---------------------------------------------------------------------------
ROM:84000010                 LDR             PC, =_dataabt
ROM:84000014 ; ---------------------------------------------------------------------------
ROM:84000014                 LDR             PC, =_halt
ROM:84000018 ; ---------------------------------------------------------------------------
ROM:84000018                 LDR             PC, =_irq
ROM:8400001C ; ---------------------------------------------------------------------------
ROM:8400001C                 LDR             PC, =_fiq
ROM:8400001C ; ---------------------------------------------------------------------------
ROM:84000020                 DCD _reset
ROM:84000024 off_84000024    DCD _undef              ; DATA XREF: ROM:84000004r
ROM:84000028 off_84000028    DCD _swi                ; DATA XREF: ROM:84000008r
ROM:8400002C off_8400002C    DCD _prefabt            ; DATA XREF: ROM:8400000Cr
ROM:84000030 off_84000030    DCD _dataabt            ; DATA XREF: ROM:84000010r
ROM:84000034 off_84000034    DCD _halt               ; DATA XREF: ROM:84000014r
ROM:84000038 off_84000038    DCD _irq                ; DATA XREF: ROM:84000018r
ROM:8400003C off_8400003C    DCD _fiq                ; DATA XREF: ROM:8400001Cr
ROM:84000040 ; ---------------------------------------------------------------------------
ROM:84000040
ROM:84000040 _reset                                  ; CODE XREF: ROM:loc_84000000j
ROM:84000040                                         ; DATA XREF: ROM:84000020o
ROM:84000040                 ADR             R0, loc_84000000
  ...
```

Слика 6: Првите неколку линии код на LLB сместен во BootRom(LLB - The iPhone Wiki)

При неуспех на вчитување или потврдување на наредните фази во синџирот се постапува различно во зависност од хардверот:

- Boot ROM не може да вчита LLB (постари уреди): Режим за надградба на фирмверот на уредот (Device Ffirmware Upgrade mode)
- LLB или iBoot: Режим за обновување (Recovery mode) каде му се овозможува на корисникот да го реинсталира оперативниот систем

5. БЕЗБЕДНИ АЖУРИРАЊА НА СОФТВЕРОТ

5.1. ПРЕГЛЕД

Заедно со безбедно стартување на iOS мора да постои и механизам за брзо и безбедно добивање на најновите безбедносни ажурирања. Apple редовно објавува софтверски ажурирања за да одговори на новите безбедносни проблеми. Корисниците на уредите со iOS добиваат известувања за ажурирање на уредот. Ажурирањата се испорачуваат безжично, за брзо прифаќање на најновите безбедносни поправки.

Процесот на ажурирање го користи истиот корен на доверба заснован на хардвер што го користи secure boot, кој е дизајниран да инсталира само код потпишан од Apple. Процесот на ажурирање исто така користи софтвер за да провери дали само копии од верзии на оперативниот систем кои се редовно потпишувани од Apple може да се инсталираат на уредите со iOS. Со воспоставување на овие безбедносни процеси, Apple може да престане да потпишува постари верзии на оперативниот систем со познати пропусти и да помогне да се спречат нападите за деградирање.

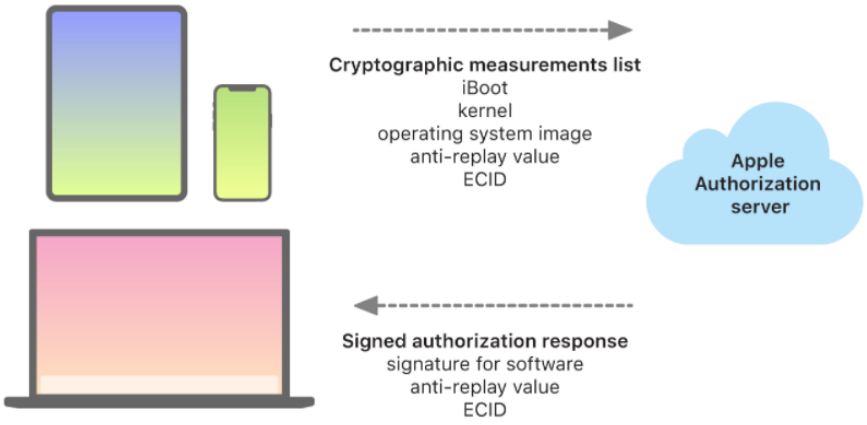
За поголема безбедност на ажурирањето на софтверот, кога уредот што треба да се надгради е физички поврзан со Mac, се презема и инсталира целосна копија од iOS или iPadOS. Но, за надградби на софтверот преку воздух (Over The Air), се преземаат само компонентите потребни за извршување на ажурирањето, со што се подобрува ефикасноста на мрежата со тоа што не се превзема целиот оперативен систем. Понатаму, ажурирањата на софтверот може да се кешираат на Mac со macOS 10.13 или понова верзија со вклучено кеширање на содржината, така што уредите со iOS не треба повторно да го преземаат

потребното ажурирање преку Интернет. (Но сепак треба да контактираат со серверите на Apple за да го завршат процесот на ажурирање.)

5.2. ПЕРСОНАЛИЗИРАН ПРОЦЕС НА АЖУРИРАЊЕ

За време на надградбите и ажурирањата, се воспоставува врска со серверот за овластување за инсталација на Apple, кој вклучува листа на криптографски мерења за секој дел од инсталацискиот пакет што треба да се инсталира, случајна вредност против повторување (nonce) и единствена ексклузивна идентификација на чипови (Exclusive Chip Identification) на уредот.

Серверот за овластување ја проверува претставената листа на мерења со верзии чија инсталација е дозволена и, доколку најде совпаѓање, го додава ECID на мерењето и го потпишува резултатот. Серверот пренесува комплетен сет на потпишани податоци на уредот како дел од процесот на надградба. Додавањето на ECID „го персонализира“ овластувањето за уредот што бара. Со овластување и потпишување само за познати мерења, серверот помага да се осигури дека ажурирањето се одвива точно како што е предвидено од Apple.

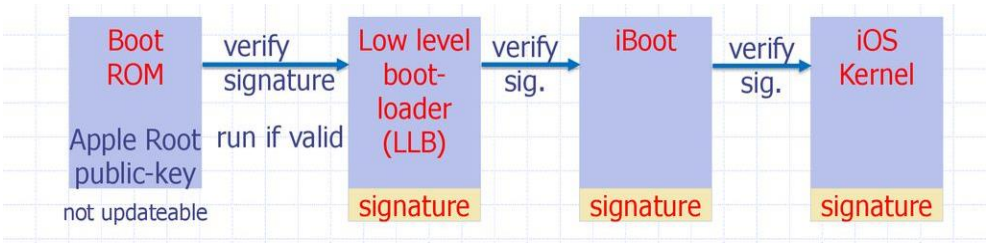


Слика 7: Персонализиран процес на ажурирање

Процес што покажува како уредите на Apple испраќаат информации до серверот за авторизација на Apple, кој пак испраќа одговор.

6. ПРЕГЛЕД НА ШИФРИРАЊЕ И ЗАШТИТА НА ПОДАТОЦИ

Secure boot ланецот, безбедноста на системот и безбедноста на апликациите помагаат да се потврди дека само доверливиот код и доверливи апликации работат на уредот. Уредите на Apple имаат дополнителни функции за шифрирање за заштита на корисничките податоци, дури и кога други делови од безбедносната инфраструктура се компромитирани (на пример, ако уредот е изгубен или користи недоверлив код). Сите овие функции имаат корист и за корисниците и за ИТ администраторите, заштитувајќи ги личните и корпоративните информации и обезбедуваат методи за инстанто и целосно далечинско бришење во случај на кражба или губење на уредот.



Слика 8: Secure boot chain

Уредите со iOS користат методологија за шифрирање на датотеки наречена Data Protection. Секоја датотека е заштитена со единствен клуч по датотека (или по екстензија). Клучот, завиткан со помош на алгоритмот за завиткување NIST AED, дополнително е обвиткан со еден од неколкуте класни клучеви, во зависност од тоа како треба да се пристапи до датотеката. Завиткаииот клуч по датотека потоа се складира во метаподатоците на датотеката.

Кога датотеката е отворена, нејзините метаподатоци се дешифрираат со клучот на датотечниот систем, откривајќи го завиткаииот клуч по датотека и нотација на која класа припаѓа. Клучот по датотека (или по екстензија) се одвиткува со клучот за класа и потоа се

доставува до хардверскиот AES Engine, кој ја дешифрира датотеката додека се чита од флеш меморијата. Целокупното ракување со клучеви со завиткани датотеки се случува во Secure Enclave; клучот за датотека никогаш не е директно изложен на Апликацискиот Процесор. При стартување, Secure Enclave преговара за ефемерен клуч со AES Engine. Кога Secure Enclave ќе ги одвитка клучевите на датотеката, тие повторно се завиткуваат со ефемерниот клуч и се испраќаат назад во Апликацискиот Процесор.

Кога се креира нова датотека, апликацијата што ја создава ѝ доделува класа. Секоја класа користи различни политики за да одреди кога податоците се достапни.

Class	Protection type
Class A: Complete Protection	(NSFileProtectionComplete)
Class B: Protected Unless Open	(NSFileProtectionCompleteUnlessOpen)
Class C: Protected Until First User Authentication *Note:* macOS uses a volume key to recreate FileVault protection characteristics.	(NSFileProtectionCompleteUntilFirstUserAuthentication)
Class D: No Protection	(NSFileProtectionNone)

Слика 9: Class and protection type

Целосна заштита (NSFileProtectionComplete): Клучот за класа е заштитен со клуч што произлегува од корисничката лозинка или лозинка и UID на уредот. Набргу откако корисникот ќе заклучи уред, дешифрираниот клуч за класа се отфрла, што ги прави недостапни сите податоци од оваа класа додека корисникот повторно не ја внесе лозинката или не го отклучи (најави) уредот користејќи Touch ID или Face ID.

Заштитено освен ако не е отворено (NSFileProtectionCompleteUnlessOpen): Некои датотеки можеби ќе треба да се запишат додека уредот е заклучен или кога корисникот е одјавен. Добар пример за ова е електронска пошта што се презема во позадина. Ова

однесување се постигнува со користење на асиметрична криптографија со елипсовидна крива (ECDH преку Curve25519).

Заштитено до автентикација на првиот корисник (NSFileProtectionCompleteUntilFirstUserAuthentication) : Оваа класа се однесува на ист начин како и целосна заштита, освен што дешифрираниот клуч за класа не се отстранува од меморијата кога уредот е заклучен или кога корисникот е одјавен. Ова е стандардна класа за сите податоци за third-party апликации кои инаку не се доделени на класа за заштита на податоци.

Нема заштита (NSFileProtectionNone): Овој клуч за класа е заштитен само со UID и се чува во Effaceable Storage. Бидејќи сите клучеви потребни за дешифрирање датотеки од оваа класа се зачувани на уредот, шифрирањето ја овозможува само користа од брзото далечинско бришење. Ако на датотеката не и е доделена класа за заштита на податоци, таа сè уште се складира во шифрирана форма (како што се сите податоци на уред со iOS и iPadOS).

7. БЕЗБЕДНОСТ НА АПЛИКАЦИЈАТА

Апликациите се едни од најкритичните елементи на безбедносната архитектура, имаат потенцијал негативно да влијаат врз безбедноста, стабилноста и податоците на системот, доколку не се постапува правилно.

Поради ова, Apple обезбедува слоеви на заштита за да се осигура дека апликациите се ослободени од малициозен софтвер и дека не се подложни на манипулација. Дополнителните заштити го ограничуваат пристапот на апликациите до корисничките приватни информации.

На iPhone сите апликации се добиваат од App Store - и сите апликации се „sandboxed“, за да се обезбедат најстрогите контроли. Sandboxing помага да се заштитат податоците на корисниците од неовластен пристап на апликациите.

Native capability	Third-party equivalent
Plug-in unapproved list, Safari extension unapproved list	Virus/Malware definitions
File quarantine	Virus/Malware definitions
XProtect/YARA signatures	Virus/Malware definitions
MRT (Malware Removal Tool)	Endpoint protection
Gatekeeper	Endpoint protection; enforces code signing on apps to help ensure that only trusted software runs
eficheck (Necessary for a Mac without an Apple T2 Security Chip)	Endpoint protection; rootkit detection
Application firewall	Endpoint protection; firewalling
Packet Filter (pf)	Firewall solutions
System Integrity Protection	Built into macOS
Mandatory Access Controls	Built into macOS
Kext exclude list	Built into macOS
Mandatory app code signing	Built into macOS
App notarization	Built into macOS

Слика 10: Безбедност на апликацијата

8. БЕЗБЕДНОСТ НА СЕРВИСИ

Apple има изградено големо множество на сервиси за да им помогне на корисниците да добијат уште поголема корисност и продуктивност од нивните уреди. Овие сервиси на корисниците им нудат функционалности како на пример : складирање во облак (cloud storage), синхронизација, складирање лозинки, автентикација, плаќање, пораки, комуникации и многу повеќе, а сето тоа ја штити приватноста на корисниците и безбедноста на нивните податоци.

Некои од многуте сервиси кои ги нуди iOS се iCloud, Sign in with Apple, Apple Pay, iMessage, Business Chat, FaceTime, Find My и Continuity.

Слика 11: Безбедност на услугите

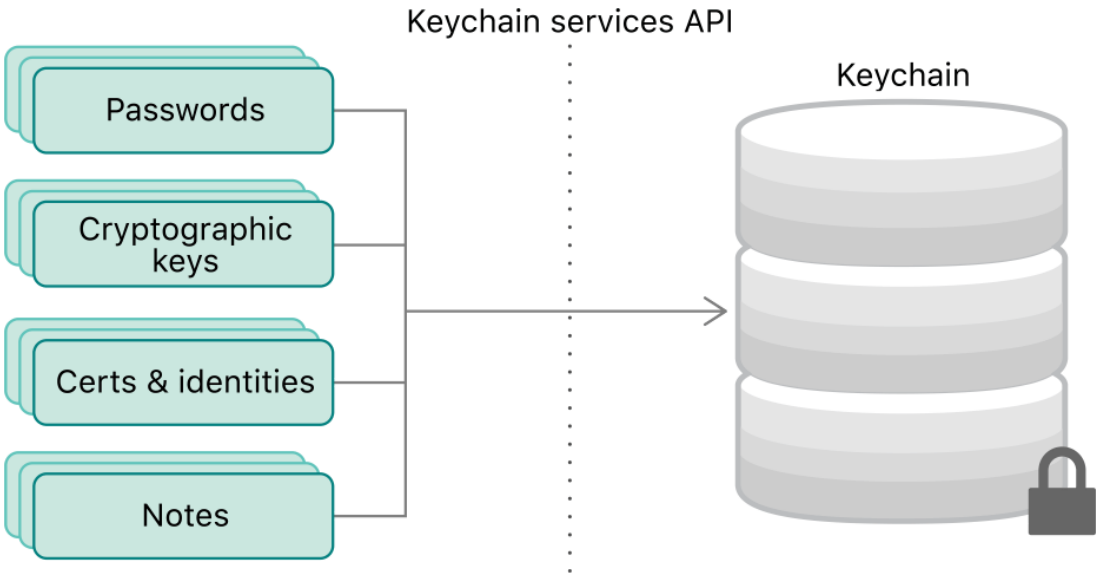
9. KEYCHAIN SERVICES

Корисниците на компјутер често имаат мали тајни што треба безбедно да ги чуваат. На пример, повеќето луѓе управуваат со онлајн сметки. Запомнувањето на сложена, единствена лозинка за секоја од нив е невозможно, но нивното запишување е и несигурно и досадно. Корисниците обично реагираат на оваа ситуација со реискористување на едноставни лозинки на многу сметки, што е исто така небезбедно.

API за keychain services помага да се реши овој проблем со тоа што на апликацијата и дава механизам за складирање на мали делови од кориснички податоци во шифрирана база на податоци наречена keychain. Кога безбедно ќе се запомни лозинката за нив, го ослободувате корисникот да избере комплицирана лозинка.

Keychain не е ограничен на лозинки, како што е прикажано на сликата. Може да се складираат други тајни за кои корисникот експлицитно се грижи, како што се информации за кредитна картичка или дури и кратки белешки. Може исто така да се складираат предмети што му се потребни на корисникот, но можеби не ги знае. На пример, криптографските клучеви, сертификати и доверливи услуги му овозможуваат на

корисникот да се вклучи во безбедна комуникација и да воспостави доверба со други корисници и уреди.



Слика 12: Обезбедување на тајните на корисникот во (keychain)

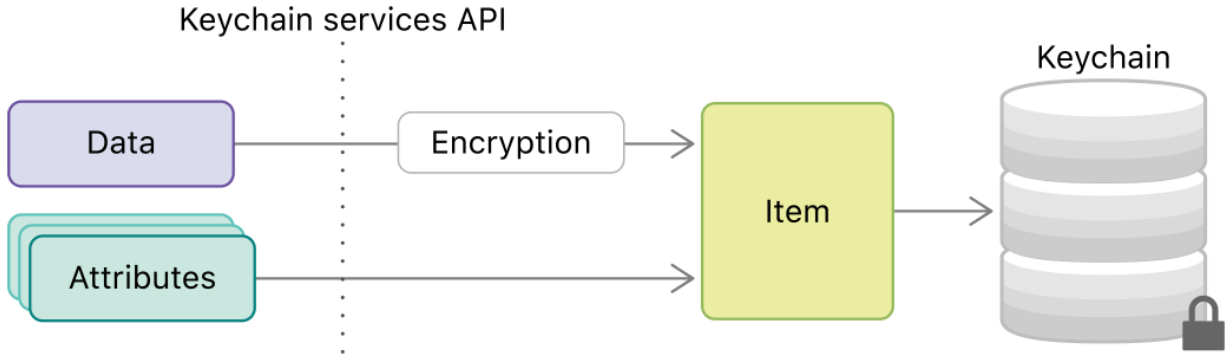
9.1. KEYCHAINS

Во iOS, апликациите имаат пристап до еден keychain (кој логично го опфаќа iCloud keychain). Овој keychain автоматски се отклучува кога корисникот го отклучува уредот, а потоа се заклучува кога уредот е заклучен. Апликацијата може да пристапи само до сопствените ставки на keychain-от или до оние што се споделени со групата на која припаѓа апликацијата. Дополнително, апликацијата не може да управува со самиот keychain container.

9.2. KEYCHAIN ITEMS

Кога корисникот сака да зачува тајна како лозинка или криптографски клуч, ја пакува како keychain item. Заедно со самите податоци, се обезбедува збир на јавно видливи атрибути и за да се контролира пристапноста на ставката и да се направи да може да се

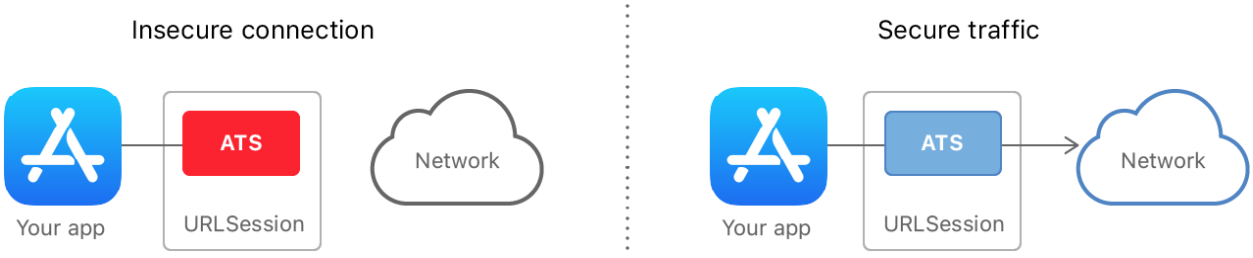
пребарува. Како што е прикажано на сликата, услугите со (keychain) се справуваат со шифрирање и складирање на податоци (вклучувајќи ги атрибутите на податоците) во keychain-от, што е шифрирана база на податоци складирана на дискот. Подоцна, овластените процеси користат keychain сервиси за да го пронајдат предметот и да ги дешифрираат неговите податоци.



Слика 13: Ставање податоци и атрибути во синџир со клучеви

10. СПРЕЧУВАЊЕ НА НЕБЕЗБЕДНИ МРЕЖНИ ВРСКИ

На платформите на Apple, мрежна безбедносна функција наречена Безбедност на транспорт на апликации (Application Transport Security) ја подобрува приватноста и интегритетот на податоците за сите апликации и екстензии на апликации. Тоа го прави со барање мрежните врски направени од апликацијата да бидат обезбедени со протоколот Transport Layer Secutiry со користење на доверливи сертификати и шифри. ATS блокира врски кои не ги исполнуваат минималните безбедносни барања.



Слика 14: Insecure connection vs. Secure traffic

Дијаграм што покажува како ATS блокира небезбедни врски, но овозможува безбеден проток на сообраќај помеѓу апликацијата и мрежата.

11. IOS JAILBREAK

11.1. ШТО Е JAILBREAK?

Jailbreak значи дозволување third-party апликации да се инсталираат на Apple уредот. Спротивно на мислењата на голем дел од корисниците, целосно е легално да се извршуваат third-party апликации Apple уредите. Jailbreaking-от дозволува root пристап до датотечниот систем и менаџер во iOS, овозможувајќи преземање дополнителни апликации, екстензии и теми кои се недостапни преку официјалната продавница за апликации на Apple. Единственото нешто што ги спречува луѓето да направат џеилбрејк е самиот Apple.

11.2. ВИДОВИ ЏЕИЛБРЕЈКОВИ

Кога уредот се вклучува, тој започнува со првично вчитување на кернелот на Apple. Уредот потоа мора да се експлоатира и кернелот да се модификува секој пат кога ќе се вклучи.

- Untethered jailbreak → „untethered“ jailbreak е процес каде што се постигнува jailbreak без потреба од користење на компјутер. Кога корисникот го исклучува и повторно го вклучува уредот, уредот целосно се вклучува, а кернелот се модификува без потреба од компјутер. Иако ова звучи лесно, овој вид на џеилбрејк е потешко да се направи и бара многу големо познавање на хардверската и софтверската архитектура на уредот.
- Tethered jailbreak → Со „врзан“ џеилбрејк, потребен е компјутер за вклучување на уредот секогаш кога ќе се рестартира. Ако уредот се стартува сам, тој повеќе нема да има модификуван кернел и може да се заглави во делумно стартувана состојба.

Во основа, целта на компјутерот е да го „re-jailbreak“-не телефонот секогаш кога ќе се вклучи.

- Semi-tethered jailbreak → Ова во суштина значи дека кога уредот ќе се вклучи, тој повеќе нема да има модификувано јадро, што значи дека нема да може да се изврши изменет код. Но, може да се користи за нормални функции. Кога треба да се користат функции за кои е потребен изменет код за да се стартува, корисникот мора да го стартува уредот со помош на алатка за џеилбрејк.

11.3. КАКО ФУНКЦИОНИРА ЏЕИЛБРЕЈКОТ?

Jailbreak-от овозможува да придобивка на контрола над root и медиумската партиција на уредот. Ова е местото каде што се складираат сите датотеки на iOS. За да се постигне ова, датотеката на локација /private/etc/fstab мора да биде изменета.

fstab е како прекинувач што ги контролира дозволите за root и медиумските партиции. Стандардно, ова е поставено на режим „read-only“. За да може да се прават модификации, треба да се постави fstab во режим „read-write“. Иако ова може да звучи лесно, најголемиот проблем е да се внесат сите датотеки што се потребни на корисникот преку различни контролни точки. Контролните точки се начин на Apple да се осигура дека датотеката е легитимна. Секоја датотека е потпишана со клуч и без него, датотеката ќе биде ставена настрана и ќе биде неупотреблива.

Од каде се добива клучот? Со едноставни зборови, пристапот до вратата може да се обезбеди или ако се одврти бравата (сите контролни точки да се изменат) или да се најде влез на задната врата (bypass). Модификување на контролните точки е тешка задача, затоа најчесто се употребува втората стратегија со користење на задна врата.

11.4. ЏЕЈЛБРЕЈКОТ МОЖЕ ДА ОТВОРИ И МНОГУ БЕЗБЕДНОСНИ ДУПКИ

Third-party апликации можат да бидат опасни. Има причина зошто Apple наметнува повеќе ограничувања од кој било друг мобилен оперативен систем. Злонамерна апликација може да предизвика многу хаос на уредот. Секогаш е можно да се добие лоша апликација, но со преземање апликации што не биле одобрени од Apple за App Store, шансите да се добие малициозен софтвер се зголемуваат.

Безбедносните ажурирања нема да се преземаат. Откако ќе се џеилбрејкне iPhone, нема да може да се ажурира iOS без да се врати на стандардниот режим без џеилбрејк. Иако ова не е голема работа, повеќето луѓе кои направиле џеилбрејк на нивните уреди со iOS ќе чекаат додека не биде достапен нов џеилбрејк.

Сите ја знаат стандардната лозинка. Една од најлошо чуваните тајни за iOS е неговата root лозинка, „alpine“. Сите го знаат тоа, а Apple нема намера да го промени. Имањето root лозинка му дава пристап на корисникот до основните функции на уредот, а тоа може да биде катастрофално ако падне во погрешни раце. Добрата работа е што оваа лозинка може да се смени, но џеилбрејкерите често забораваат да го направат тоа оставајќи ги нивните уреди отворени за пропусти.

12. НЕКОИ ПОПОЗНАТИ ЕКСПЛОАТАЦИИ НА IOS

12.1. PWNAGE/PWNAGE2

Pwnage користи лош синџир на доверба во низата за вклучување на уредот S5L8900. Редоследот на стартување на уредот вклучува Low Level Bootloader и iBoot модули кои се складирани во NOR флеш-меморијата на уредот и обично се шифрирани. Сепак, во тие верзии на хардверот овие процеси не се криптографски потпишани од Apple. Pwnage го искористува овој факт.

Прво, Apple претпоставува дека ако нешто е во меморијата, тоа нужно поминало низ потврда на криптографскиот потпис, и затоа е автентичен код на Apple. Ова е неточно,

бидејќи единствениот механизам што го спречува пишувањето на неовластен код на меморијата е кернелот. Јадрото на iPhone содржи екстензија дизајнирана специјално за пишување на NOR, наречена AppleImage2NORAccess. Оваа екстензија врши проверка на потписот на сите податоци што се обидува да ги напише. Самата верификација ја врши екстензијата FairPlay, која е многу заматена, но деактивација на проверката е многу едноставна. Откако ќе се отстрани проверката, меморијата е подложна на секакви запишувања.

Второ, Apple претпоставува дека деактивацијата на клучевите за шифрирање во „нормална“ средина ќе спречи запишување на датотеки на NOR меморијата. Сепак, бил пронајден начин да се изврши експлоатацискиот код во „безбедна“ средина и да се искористи екстензијата AppleImage2NORAccess на ист начин како што Apple го прави тоа при обновување.

Пред iOS 2.0, NOR бил поставен на начин што кога сликите на софтверот биле запишани во мемријата, криптографските потписи биле исфрлени. Така, иако потписот на iBoot го проверувал кернелот, LLB не го проверувал потписот на iBoot, а VROM не го проверувал потписот на LLB.

Pwnage започнува со вклучување од мемориски уред (ramdisk) во „безбедна“ средина за да се спречи кернелот да ги оневозможи клучевите за шифрирање. Исто така, се додава уште еден мемориски уред, насочен кон просторот за адреси на кернелот, за да овозможи модификување на кернелот во живо. По стартувањето, се деактивира проверката на потписот од екстензијата AppleImage2NorAccess и се продолжува со запишување на експлоатационите датотеки (iBoot, LLB, DeviceTree и слики). Бидејќи проверката на потписот е деактивирана и достапни се клучевите за шифрирање, AppleImage2NORAccess без бунење ги запишува на соодветната локација во меморијата. После тоа, уредот може да се рестартира и ќе ја прифати секоја непотпишана датотека.

12.2. 24KPWN

По стартување, S5L8720 и S5L8920 SoC имаат конфигурација што го мапира Secure ROM-от на адресата 0x0. Оваа конфигурација исто така мапира мала количина на SRAM на 0x22000000 за S5L8720 и 0x84000000 за S5L8920. Статички алоцираните променливи, heap-от и stack-от мора да користат SRAM, бидејќи на „Secure ROM“ не може да се запишува. За таа цел се користи регион на меморија што почнува од (SRAM Start)+0x24000. Регионот на меморија од почетокот на SRAM до (SRAM Start) + 0x24000 се користи како buffer за вчитување на кодот на bootloader-от на следната фаза. LLB кодот се чува во NOR флеш меморијата, заедно со кодот за сите други фази, како и уметничките ресурси (лого) и Device Tree (мапа на сите хардверски компоненти на уредот) што треба да се обезбедат на кернелот. Првиот дел (првите 0x160 бајти) од меморијата на (SRAM Start)+0x24000 се користи за иницијализирани статички алоцирани променливи. Набргу по вклучувањето, вредностите во тој регион се иницијализираат од страна на Secure ROM.

Слабоста во горе опишаната организација е фактот што кодот што ја копира датотеката LLB IMG3 од флеш меморијата, при запишување во SRAM не ја проверува големината на сликата што се вчитува, наместо тоа ја зема големината директно од IMG3 заглавјето кое не е криптографски потпишано. Секоја слика поголема од 0x24000 бајти ќе започне да го покрива делот од меморијата што се користи за складирање на статички алоцирани променливи. Податоци кои при тоа ќе бидат изложени на ризик се USB податочни структури, покажувачи кон разни структури како списоци со задачи за распоредувачот на Secure ROM, како и адреси на хардверските SHA1 регистри. Сите горенаведени се потенцијални патишта за експлоатација.

13. ЗАКЛУЧОК

Apple е посветен на помагање во заштитата на клиентите со водечки технологии за приватност и безбедност - дизајнирани за заштита на личните информации - и сеопфатни методи, за да помогнат во заштитата на корпоративните податоци во претпријатието. Apple

ги наградува истражувачите за работата што ја прават за да ги откријат пропустите нудејќи го Apple Security Bounty.

Тимот на Apple е тим за поддршка на сите производи кои ги има. Тимот обезбедува безбедносна ревизија и тестирање за производите, како во развој, така и објавени. Тимот на Apple, исто така, обезбедува безбедносни алатки и обука, и активно ги следи заканите и извештаите за нови безбедносни проблеми. Apple е член на Форумот на тимови за одговор на инциденти и безбедност (FIRST).

Apple продолжува да ги поместува границите на она што е можно во безбедноста и приватноста.

КОРИСТЕНА ЛИТЕРАТУРА

[1] David Thiel, " *IOS Application Security: The Definitive Guide for Hackers and Developers*" , No starch press, San Francisco ,297 страни

[2] Allister Banks, Charles S. Edge, "*Learning IOS Security*", PACKT Publishing, Bigmingham-Mumbai, 128 страни

[3] Glenn Fleishman,"*Take Control of iOS & iPadOS Privacy and Security*", 2nd Edition, 361 страни