# Bootloader

4.

- Un bootloader este un program specializat, proprietar şi specific pentru hardware-ul pe care este executat (pentru un anumit System on Chip - SoC).
- Acesta rulează atunci când dispozitivul este pornit (pe dispozitivele ARM asta înseamnă atunci când dispozitivul iese din reset).
- Are scopul de a iniţializa hardware-ul, apoi de a găsi şi a porni sistemul de operare.
- Boot-area include de obicei mai multe etape şi există un bootloader pentru fiecare etapă. Dar în curs ne vom referi la un singur bootloader care include toate etapele.

5.

- Majoritatea bootload-erelor oferă un mod numit fastboot sau download care permite scrierea (flashing) partiţiilor raw în spaţiul de stocare permanent al dispozitivului, dar şi bootarea unor imagini de sistem temporare (fără scrierea lor pe dispozitiv).
- Modul fastboot este activat de o combinaţie specială de taste hardware în timpul boot-ării, dar şi prin trimiterea comenzii “adb reboot bootloader”.

6.

- Dispozitivele de pe piaţă vin cu un bootloader blocat (locked) pentru a asigura integritatea imaginilor ce rulează pe ele.
- Atunci când bootloader-ul este blocat, nu se va putea scrie sau boota alte imagini de sistem
- În cel mai bun caz se vor putea scrie imagini de sistem care au fost semnate de către producătorul dispozitivului.
- Cele mai multe dispozitive vor permite deblocarea bootloader-ului, ceea ce dezactivează restricţiile fastboot şi verificările de semnătura a imaginilor.
- De obicei, deblocarea bootloader-ului necesită formatarea partiţiei userdata, pentru a nu permite unui sistem de operare maliţios să aibă acces la datele curente ale utilizatorului.

# Recovery

8.

- O metodă mai flexibilă de a actualiza un dispozitiv este prin Recovery OS.
- Acesta este un sistem de operare minimal bazat pe Linux, care include un kernel, un RAM disk cu diferite utilitare de nivel scăzut şi un UI minimal.
- Este stocat pe partiţia de recovery şi de obicei folosit pentru a aplica pachete Over-the-air (OTA).

- Pachetele OTA includ de obicei noi versiuni (patch-uri binare) ale unor fişiere de sistem şi un script care aplică modificările.
- Fişierele din OTA sunt semnate folosind cheia privată a producătorului
- Imaginea de recovery include şi cheia publică pentru verificarea fişierelor OTA înainte de a face modificările. Acest lucru ne garantează faptul că fişierele OTA provin dintr-o sursă de încredere.

9.

- Imaginea de recovery este scrisă pe partiţia de recovery şi poate fi suprascrisă atunci când dispozitivul este în modul fastboot/download.
- Atunci când se scrie o imagine de recovery custom, este posibilă dezactivarea verificării semnăturilor OTA. Acest lucru va putea permite modificarea sistemului de operare principal.
- De asemenea, un recovery custom poate permite accesul la root prin ADB.
- Mai poate permite obţinerea citirea şi salvarea datelor de pe partiţiile existente.

10.

- Dacă partiţia de date este criptată, atunci nu se pot accesa datele în mod direct dintr-un custom recovery.
    - Dar se poate instala un rootkit (un program maliţios) pe partiţia system din custom recovery.
    - Rootkit-ul va permite accesul remote la dispozitiv în timp ce acesta este în sistemul de operare principal.
    - Astfel se vor putea accesa datele necriptate (ele sunt decriptate în mod transparent de către sistemul de operare la bootare).
- Mecanismul verified boot poate preveni acest atac dacă este verificată partiţia de boot folosind o cheie nealterabilă, stocată în hardware.
- Astfel, mecanismele de securitate verified boot şi criptarea disk-ului vor putea limita daunele făcute de o imagine de sistem maliţioasă (scrisă atunci când bootloader-ul este deblocat)

# Verified Boot

12.

- Mecanismul de verified boot este bazat pe device mapper.
- Acesta este un framework din kernelul de Linux, care oferă un mod generic de implementare a dispozitivelor bloc virtuale.
    - Acest framework este baza pentru Logical Volume Manager (LVM) în Linux şi este folosit pentru a implementa criptarea întregului disc (dm-crypt), RAID arrays şi storage replicat distribuit.

- Device mapper funcţionează prin maparea unui dispozitiv bloc virtual peste unul sau mai multe dispozitive fizice, şi prin modificarea datelor în tranzit.

13.
- Mecanismul de verified boot din Android este bazat pe dm-verity.
- Dm-verity este un target de device mapper de verificare a integrităţii blocurilor. Asta înseamnă că verifică integritatea fiecărui bloc atunci când este citit de pe disc.
- Dacă verificarea are succes atunci blocul de date este citit. Dacă nu, atunci citirea se va întoarce cu o eroare I/O (ca şi cum blocul a fost corupt fizic).

14.
- Dm-verity este implementat folosind un arbore Merkle care include toate hash-urile blocurilor de pe dispozitiv.
- Nodurile frunză ale arborelui includ hash-urile blocurilor fizice, iar nodurile intermediare sunt hash-uri ale nodurilor copil (hash-uri de hash-uri).
- Nodul rădăcină (numit şi hash-ul rădăcină) este bazat pe toate hash-urile de la nivelurile inferioare.
- O singură modificare asupra unui bloc va duce la modificarea hash-ului rădăcină.
- Prin urmare, pentru a verifica dacă un arbore hash este nemodificat, e nevoie doar de verificarea hash-ului rădăcină.

15.
- În timpul rulării, dm-verity calculează hash-ul fiecărui bloc atunci când este citit şi îl verifică prin traversarea arborelui de hash-uri precalculat.
- Citirea datelor de pe dispozitivul fizic este deja o operaţie ce consumă timp, iar latenţa introdusă de hashing şi verificare este neglijabilă.
- Odată verificat, un bloc este cache-uit şi la citirile ulterioare ale aceluiaşi bloc nu se va mai face nici o verificare de integritate.
- Dm-verity depinde de arborele de hash-uri precalculat ale tuturor blocurilor unui dispozitiv, de aceea dispozitivul trebuie să fie montat read-only pentru ca verificarea să fie posibilă.
- Montarea unui dispozitiv read-write va duce automat la eşuarea verificării de integritate.
    - Chiar dacă fişierele nu sunt modificate la rulare, se vor modifica anumite metadate din superbloc.

16.
- De aceea mecanismul dm-verity functionează bine cu partiţia system, deoarece aceasta este modificată doar prin actualizarea sistemului de operare.
    - Orice altă modificare va însemna coruperea sistemului de operare sau a discului, sau poate indica faptul că un program maliţios încearcă să modifice sistemul de operare (un fişier de sistem).

    - Dm-verity se potriveste bine cu modelul de securitate al Android-ului, care foloseşte o partiţie read-write doar pentru datele aplicaţiilor, şi stochează fişierele sistemului de operare pe partiţia system care este montată read-only.

17.
- Din Android 4.4 este folosit target-ul dm-verity, dar mecanismul este implementat un pic diferit fată de cel din kernelul de Linux.
    - Mai exact ce e implementat diferit: verificarea hash-ului rădăcină şi montarea partiţiilor verificate
    - Cheia publică RSA folosită pentru verificare este stocată pe partiţia de boot (fişierul “verity_key”), şi este folosit pentru a verifica tabela de mapare dm-verity care include: locaţia dispozitivului, offset-ul tabelei de hash-uri, hash-ul rădăcină şi salt-ul.

18.
- Verity metadata block include tabela de mapare şi semnătura, şi este scris pe disc imediat după ultimul bloc al sistemului de fişiere.
- Partiţia este marcată verificabilă prin adăugarea unui verity flag în fişierul fstab.

19.
- Atunci când managerul sistemului de fişiere întâlneşte acest verity flag, încarcă metadatele verity de pe dispozitiv şi verifică semnătura folosind cheia verity.
- Dacă verificarea semnăturii se face cu succes, managerul sistemului de fişiere parsează tabela de mapare dm-verity şi o pasează mai departe la device mapper-ul din Linux.
- Device mapper va folosi informaţiile din tabela de mapare pentru a crea dispozitivul bloc virtual dm-verity.

20.
- Acest dispozitiv bloc virtual este montat în punctul de montare specificat în fstab, în locul dispozitivului fizic.
- Astfel, toate citirile de bloc de pe dispozitivul fizic vor fi verificate în mod transparent folosind arborele de hash-uri precalculat.
- Dacă sunt modificate sau adăugate fişiere, sau remontată partiţia ca read-write, se va face o verificare de integritate care va eşua şi va fi generată o eroare I/O.

21.
- Pentru a asigura protecţia integrităţii, trebuie să avem încredere în kernelul care conţine dm-verity. Prin urmare partiţia de boot trebuie să fie de încredere.
- Pe Android, acest lucru necesită verificarea partiţiei de boot, care conţine kernelul, RAM disk-ul şi cheia verity.
- Această verificare depinde de acel dispozitiv şi este implementată  de obicei în bootloader.

- Verificarea se face folosind o cheie nealterabilă de verificare a semnăturii care este stocată în hardware.

22.

- Procedura pentru activarea verified boot pe Android include următorii paşi:
    - Generarea arborelui de hash-uri
    - Crearea tabelului de mapare dm-verity
    - Semnarea tabelului
    - Generarea şi scrierea blocului de metadate verity pe dispozitiv.

23.

- Arborele de hash-uri dm-verity este generat folosind programul veritysetup, care este parte din cryptsetup.
    - Care este un pachet de utilitare de gestiune a discurilor.
- Programul veritysetup poate acţiona direct asupra dispozitivelor bloc şi genera un arbore de hash-uri pe baza unei imagini de sistem de fişiere, şi poate scrie tabela hash într-un fişier.
- Arborele trebuie scris pe acelaşi dispozitiv, de aceea, atunci când se apelează veritysetup, trebuie specificat offset-ul - o locaţie după blocul de metadate verity.

24.

- Paşii parcurşi pentru generarea arborelui sunt:
- Alegerea unui salt random
- Împărţirea imaginii de sistem în blocuri de 4k
- Pentru fiecare bloc se calculează un hash SHA256 (se ia salt-ul în calcul).
- Nivelul 1 este format din concatenarea hash-urior în blocuri de 4k.
- Se face padding cu 0 pentru a ajunge la un multiplu de 4k.
- Se adaugă nivelul în arbore.
- Se repetă paşii 2-6 pe baza nivelului tocmai generat, până când se obţine un singur hash (rădăcină).

25.

- Aici este o ilustrare a arborelui de hash-uri.

26.

- Atunci când se generează arborele de hash-uri, se generează hash-ul rădăcină
    - Acesta este folosit pentru a tabela de mapare dm-verity pentru acel dispozitiv
- Tabela de mapare conţine versiunea de dm-verity, dispozitivul ce stochează datele şi hash-urile, dimensiunile blocurilor de date şi hash-uri, locaţia pe disk a datelor şi a arborelui de hash-uri, algoritmul de hash-ing, hash-ul rădăcină şi salt-ul.

27.

- Tabela de mapare este semnată folosind o cheie RSA pe 2048 biţi, în format mincrypt (o bibliotecă criptografică minimalistă folosită şi pentru verificarea semnăturilor OTA)
    - Formatul mincrypt este o serializare a structurii RSAPublickey.
    - Cheia se găseşte pe partiţia de boot, în fişierul /verity_key.
- Semnătura rezultată este în format PKCS#1 v1.5.
- Blocul de metadate verity are 32 KB şi include tabela de mapare plus semnătura.

28.

- Mai exact blocul de metadate verity include:
- Magic number - folosit de managerul sistemului de fisiere pentru verificarea consistenței tabelei (4B)
- Versiunea - deoarece blocul se poate extinde pentru a permite diferite modificări (4B)
- Signature - semnătura tabelei în format PKCS1.5 în forma padded (256B)
- Lungimea tabelului (4B)
- Tabelul în sine
- Padding - de zerouri până la 32k

29.

- Ultimul pas din procesul de activare al verified boot, este modificarea fişierului fstab pentru a activa verificarea integrităţii blocurilor pentru partiţia system.
- Pentru aceasta este nevoie doar de adăugarea flag-ului “verify”.
- Aveţi aici un exemplu de fişier fstab de pe un dispozitiv Pixel XL.
- Atunci când dispozitivul bootează, Android-ul crează automat dispozitivul virtual dm-verity pe baza intrării din fstab şi a informaţiei din tabela de mapare (conţinută în blocul de metadate).
- Dispozitivul virtual va fi montat ca /system, în locul dispozitivului fizic.

30.

- Orice modificare ulterioară a partiţiei system va cauza o eroare de verificare a integrităţii
- Aplicarea unui pachet OTA care modifică blocurile fără a modifica metadatele verity, va invalida arborele de hash-uri.
- Un OTA compatibil cu verified boot trebuie să opereze la nivel bloc şi să actualizeze atât blocurile de date cât şi arborele de hash-uri şi metadatele verity.

31.

- Verified boot verifică integritatea dispozitivului, de la rădăcina de încredere hardware până la partiţia de sistem.
- La fiecare etapa a boot-ării, se verifică integritatea şi autenticitatea următoarei etape, înainte de a se executa.
- Starea boot-ării se referă la nivelul de protecţie oferit utilizatorului atunci când dispozitivul bootează. Avem 4 stări: GREEN, YELLOW, ORANGE, şi RED.
- Starea dispozitivului se referă la posibilitatea de a rescrie imaginile (flashing). Stările posibile sunt: LOCKED, UNLOCKED.

32.

- Integritatea bootloader-ului este verificată folosind rădăcina de încredere hardware.
- Partiţiile boot şi recovery sunt verificate folosind cheia OEM (de la producător). Întotdeauna încearcă să verifice partiţia de boot cu această cheie înainte de a încerca alte chei (dacă verificarea failează).
- Dacă dispozitivul este LOCKED, se încearcă întâi cheia OEM, si apoi se încearcă verificarea folosind certificatul integrat în semnătura partiţiei.
- Dacă dispozitivul este UNLOCKED atunci este posibil ca utilizatorul să scrie imagini semnate cu alte chei.

33.

- Un dispozitiv verificat va boota într-una dintre următoarele 4 stări:
- GREEN - a fost verificat întregul lanţ de încredere, de la bootloader, la partiţia de boot şi alte partiţii verificate (system).
- YELLOW - partiţia boot a fost verificată folosind certificatul integrat şi semnătura este validă. Bootloader-ul va afişa un warning şi fingerprint-ul cheii publice înainte de a permite continuarea boot-ării.
- ORANGE - dispozitivul nu a fost verificat si poate fi modificat în mod liber. Bootloader-ul afişază un warning înainte de a continua boot-area.
- RED - verificarea dispozitivului a eşuat. Bootloader-ul va afişa un warning şi va opri boot-area.

34.

- Există două stări posibile ale dispozitivului:
- LOCKED - dispozitivul nu poate fi rescris. Un dispozitiv LOCKED poate boota doar în stările GREEN, YELLOW, sau RED.
- UNLOCKED - Imaginile dispozitivului pot fi rescrise în mod liber. Dispozitivul nu va fi verificat. Va boota întotdeauna în starea ORANGE.

35.

- Aici avem o schema cu verificarea boot-ării şi cele 4 stări de bootare.