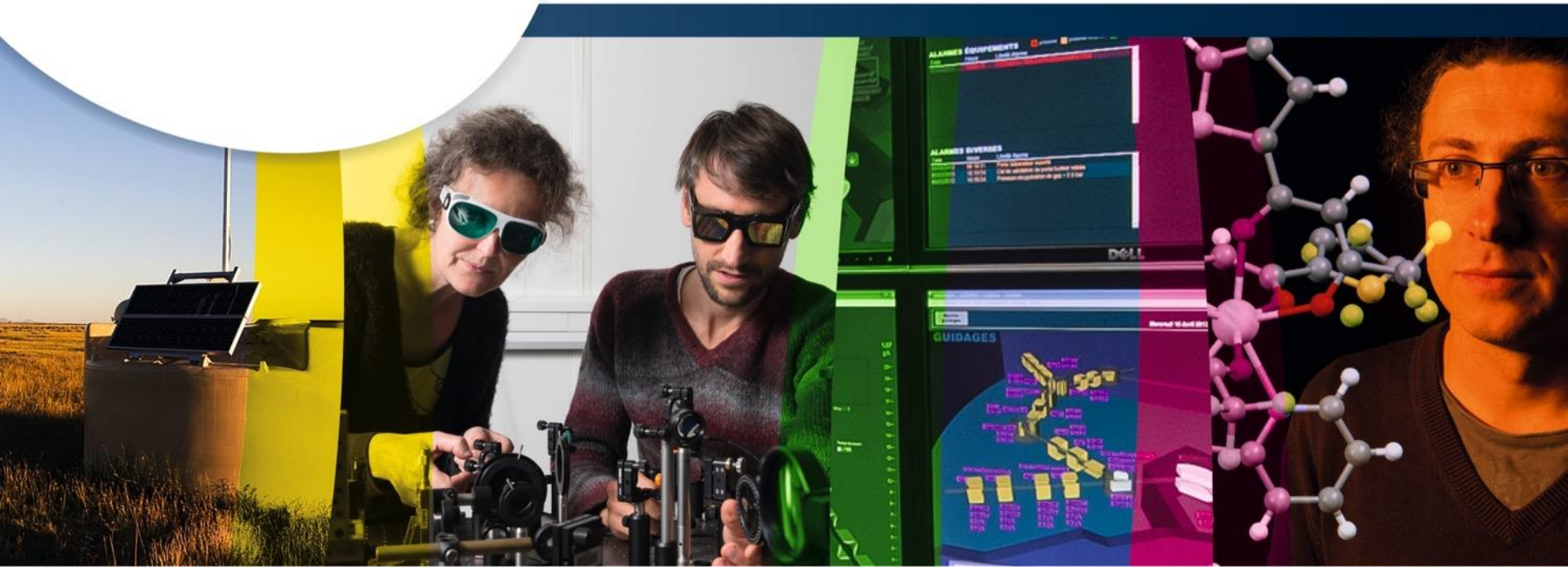
www.cnrs.fr

Mécanismes de chiffrement pour périphériques nomades



www.cnrs.fr

A quoi sert la cryptographie ?

Vocabulaire

- Chiffrer, chiffrement
  - Clé de chiffrement
- Déchiffrer, déchiffrement
  - Clé de déchiffrement
- Décrypter, décryptage
  - On ignore la clé, existe-t-elle ?
- ~~Cryptor~~, ~~cryptage~~
- Cryptologie, étymologiquement la science du secret
  - Cryptographie : s'attache à la protection des messages, assure confidentialité, authenticité et intégrité
  - Cryptanalyse
    - L'art de rendre clair un message secret
    - Attaquer un système cryptographique

Des limites de la cryptographie

- Bruce Schneier, un cryptographe renommé
  - *Applied Cryptography (1994)* : l'utopie mathématique ou la cryptographie est la réponse à tout
    - *It is insufficient to protect ourselves with laws; we need to protect ourselves with mathematics.*
  - *Secrets & Lies (2000) :* aucun système n'est parfait, la technologie n'est pas la réponse
    - *If you think technology can solve your security problems, then you don't understand the problems and you don't understand the technology.*
  - *Liars and Outliers (2012) :* traite de la confiance

Exemples de failles

- Algorithmes cassés
  - RC4
  - MD5
  - SHA1
- Attaques contre SSL/TLS

Quelques vulnérabilités SSL/TLS (d'après O. Levillain)

- 2011 : BEAST (IV implicite dans le mode CBC)
- 2012 : CRIME (utilisation de la compression comme canal auxiliaire)
- 2013 : TIME et BREACH (améliorations/adaptations de CRIME)
- 2013 : Lucky 13(exploitation d'un oracle de padding CBC)
- 2013 : RC4 (exploitation de biais statistiques)
- 2014 : goto failApple (...)
- 2014 : goto fail GnuTLS (True, False, FILE_NOT_FOUND)
- 2014 : Heartbleed (débordement de tampon en lecture)
- 2014 :Triple Handshake (attaque sur la renégociation et la reprise de session)
- 2014 : EarlyCCS (erreur dans l'automate d'état d'OpenSSL)
- 2014 : Universal signature forgery dans NSS (pendant ShellShock)

Quelques vulnérabilités SSL/TLS (d'après O. Levillain)

- 2014 : Exécution de code arbitraire dans Schannel (côté serveur)
- 2014 : POODLE (exploitation d'un oracle de padding CBC avec SSLv3)
- 2015 : SMACK/FREAK (automates d'état déficients + configurations antiques)
- 2015 : LogJam (configurations antiques + mauvaise négociation des groupes DH)
- 2016 DROWN (SSLv2 et clés export)
- 2016 : CVE-2016-0701 (réutilisation du même exposant privé pour DH)
- 2016 : CVE-2016-2108 (erreur de décodage ASN1)

Sécurité de l'information définitions (ISO 27000)

www.cnrs.fr

- Sécurité de l'information
  - Protection de la confidentialité, de l'intégrité et de la disponibilité de l'information
- Disponibilité
  - Propriété d'être accessible et utilisable à la demande par une entité autorisée
- Intégrité
  - Propriété d'exactitude et de complétude
- Confidentialité
  - Propriété selon laquelle l'information n'est pas rendue disponible ni divulguée à des personnes, des entités ou des processus non autorisés
- Auditabilité, authenticité, imputabilité, non-répudiation, fiabilité ➔ preuve

De la nécessité de la cryptographie

- Très difficile d'assurer la confidentialité sans chiffrement
  - Surveillance constante par des personnes de confiance
- Comment détecter qu'un document a été modifié sans avoir un moyen de vérification ?
  - Empreinte (*hash*), signature
- Comment prouver qu'un document a bien été écrit par tel personne sans qu'elle puisse le réfuter ?
  - Cryptographie vient à l'aide

www.cnrs.fr

Rappels de cryptographie

Hachage

- Fonction de hachage (*hash*)
  - Associe à un grand ensemble de données un ensemble beaucoup plus petit
  - Message m → haché ou empreinte h(m), entier de n bits
- Propriétés
  - Facile et rapide à calculer
  - Difficile à inverser (parcours exhaustif)
  - Bien réparti
  - Résistance
    - Préimage : h(m1) connu trouver m2 tel que h(m1) = h(m2)
    - Seconde préimage : m1 connu trouver m2 tel que h(m1) = h(m2)
    - Collisions : trouver m1, m2 tel que h(m1) = h(m2)

Fonctions de hachage

- Quelques fonctions de hachage
  - MD5 définitivement cassée
  - SHA-1 sur le point d'être cassée → ne plus utiliser
  - SHA-2 : 224, 256, 384, 512 bits, construction de Merkle-Damgård
  - SHA-3 (Keccak) nouveau, fonction éponge
- Recommandations RGS annexe B1 : **utiliser SHA-2**
  - Le mécanisme de hachage SHA-1 n'est donc pas conforme au référentiel
  - Le mécanisme de hachage SHA-256 défini dans le FIPS 180-2 est conforme au référentiel
  - Dans l'attente de la publication de nouveaux standards, dont par exemple SHA-3, et de leur mise à l'épreuve, la famille SHA-2 reste utilisable

Mots de passe

- Stockage en clair sur le serveur → catastrophique en cas de fuite
- Stockage du haché h(p)
  - Utilisé par Windows dans NTLM (MD4) et bien des sites
  - Attaque par dictionnaire
  - Attaque par force brute
  - Rainbow table : pré-calcul pour accélérer la recherche
- Utilisation de diversifiant (*salt*) h(salt || password)
  - Obstacle aux *Rainbow tables*
  - Vulnérable à des attaques hors ligne avec de puissants moyens de calcul
- Itérations d'une fonction pour ralentir le calcul → **ce qu'il faut faire**
  - Améliore la robustesse d'un mot de passe court
  - bcrypt
  - sha256crypt,
  - scrypt
  - PBKDF2
  - argon2

Mot de passe et iPhone

- San Bernardino : FBI vs Apple
  - Limite dans l'OS sur le nombre de mots de passe saisis
  - Demande de pouvoir installer un autre OS sans cette limite
    - Image signée par Apple
    - Exploitation d'un faille dans le processus de démarrage sécurisé (peut-être ce qui a été fait)

Intégrité

- Fonction de hachage pour le contrôle d'intégrité
  - La probabilité que 2 messages aient le même haché est très faible
    - $2^{-128} \sim 3 \times 10^{-39}$ pour SHA-256 (paradoxe des anniversaires)
    - On considère que cela n'arrivera jamais
  - Si un message a été modifié son haché est différent
  - Si on connait de façon sûre le haché d'un message on s'assure de son intégrité

HMAC

- HMAC (Keyed-Hash Message Authentication Code)
  - hmac(k, m) = h((k ⊕ ipad) || h(k ⊕ opad) || m)
- Intégrité + authenticité : signature
  - Hachage : intégrité
  - HMAC ajoute l'authenticité
- Exemple d'utilisation : client – serveur
  - Serveur → client : d1, s= hmac(k, d1), d2
    - d1, d2, d3 : données 1, 2, 3
    - k: clé
    - s : signature
  - Client → serveur : d1, s, d3
  - Le client ne peut modifier d1 (utile si cela contient le prix dans le panier)

Cryptographie à clé secrète

- Appelé aussi chiffrement symétrique
- La confidentialité est basée sur l'utilisation d'un secret commun.
  - Cette méthode consiste à utiliser une clé identique pour chiffrer et déchiffrer le message.
  - Il est appelé ainsi car la clé (unique) ne doit être connue que des personnes devant pouvoir accéder au secret.
- Un même algorithme est utilisé pour le chiffrement et le déchiffrement
- Système rapide, et facile à mettre en œuvre
  - Fonctions symétriques :
  - $M' = F_k(M)$ et donc $F_k^{-1}(M') = F_k^{-1}(F_k(M)) = M$
  - Algorithme basé sur des opérations mathématiques simples (substitutions, permutations).

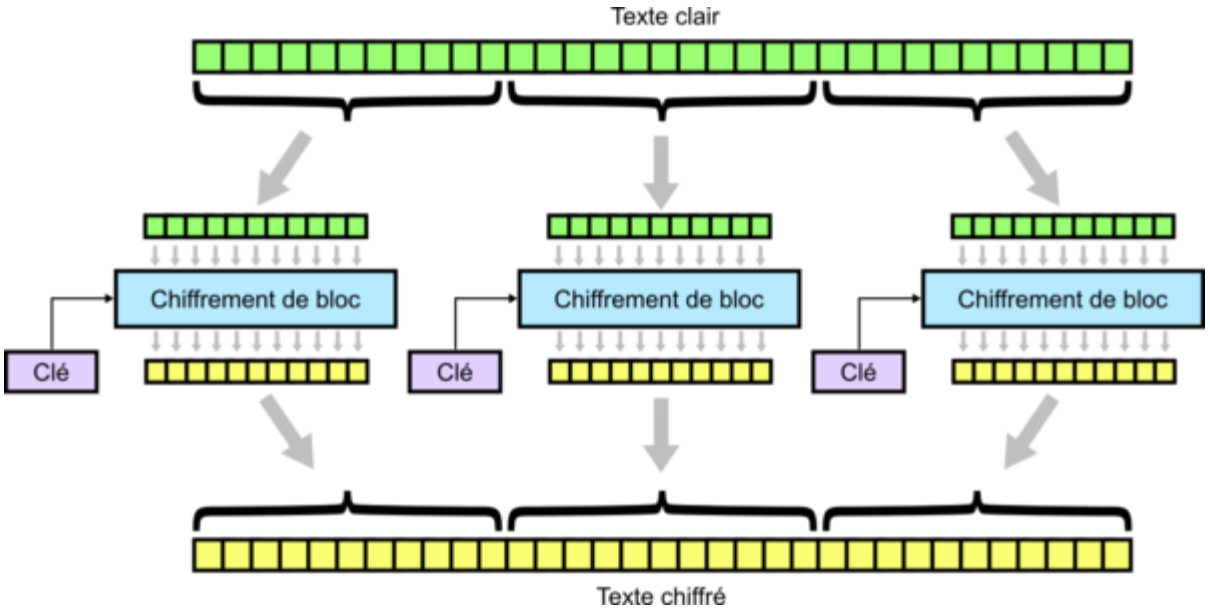
Types de chiffrement

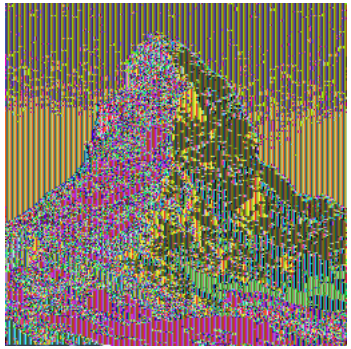
- Masque jetable (One Time Pad) : m' = m ⊕ k
  - Clé aléatoire
  - Taille clé = taille du message
  - **Ne jamais réutiliser la clé**
  - Sécurité prouvée (Shannon)
  - Difficilement praticable
- Par bloc
  - Le plus répandue,
  - Découpe les données en blocs de même taille et les chiffre ensuite les uns après les autres
  - L'opération de chiffrement s'effectue sur des blocs de taille prédéfinie.
- Par flot
  - Chiffre les données bit par bit quelle que soit la longueur du message à coder sans besoin de les découper.
  - L'opération de chiffrement (xor) prend la clé secrète et un nombre (appelé frames) pour générer un flux pseudo aléatoire de bits (appelé keystream).
  - Rapide et prend peu de mémoire
  - Pas de bons algorithmes, quasiment tous cassés (RC4, RC5). *Il est recommandé d'employer des primitives de chiffrement par bloc et non des algorithmes de chiffrement par flot dédiés (RGS).*

Modes de chiffrement

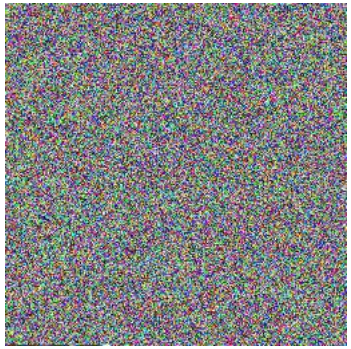
- ECB (Electronic Code Book)
  - Mauvais
  - Utilisé généralement par ceux qui se vantent d'un chiffrement « militaire » 256 bits et qui stockent parfois le mot de passe en clair
- CBC (Cipher Block Chaining)
- XTS (XEX-based tweaked-codebook mode with ciphertext stealing)
- GCM (Galois Counter Mode)
  - Chiffrement + contrôle d'intégrité
- Etc.

ECB (Electronic Code Book)



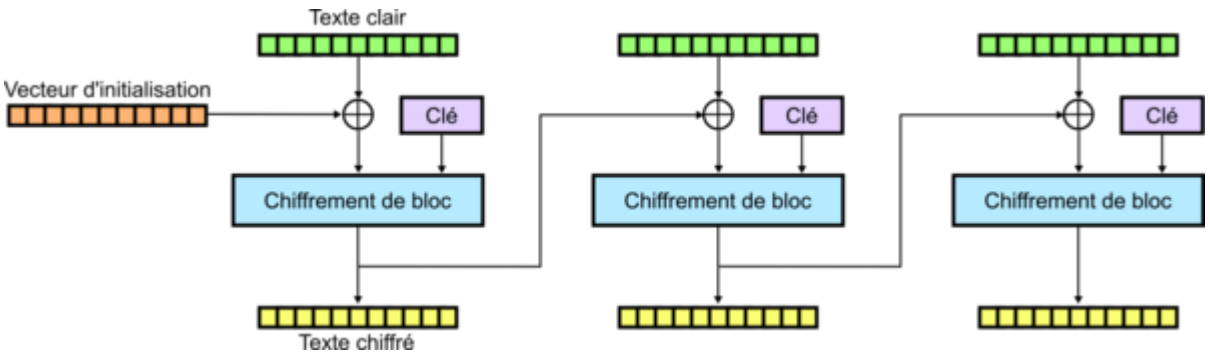


ECB



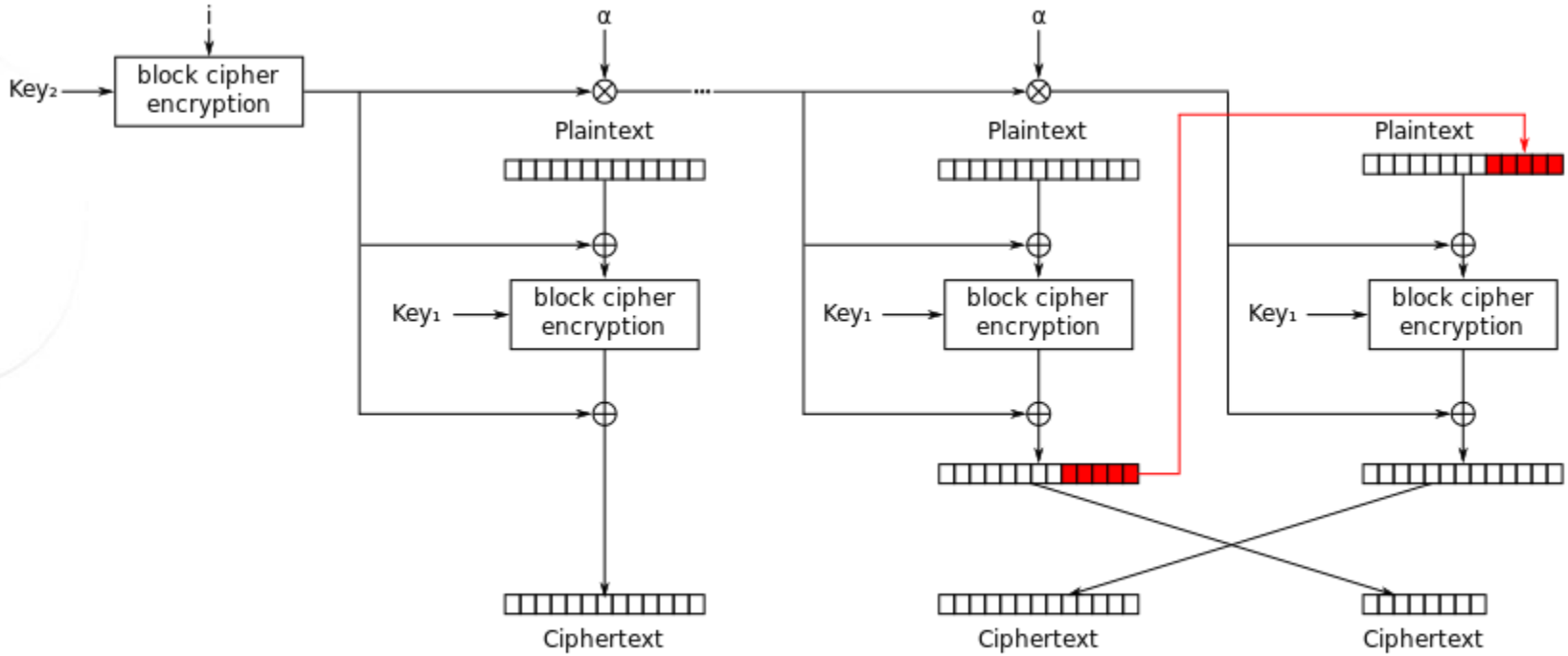
CBC

CBC (Cipher Block Chaining)



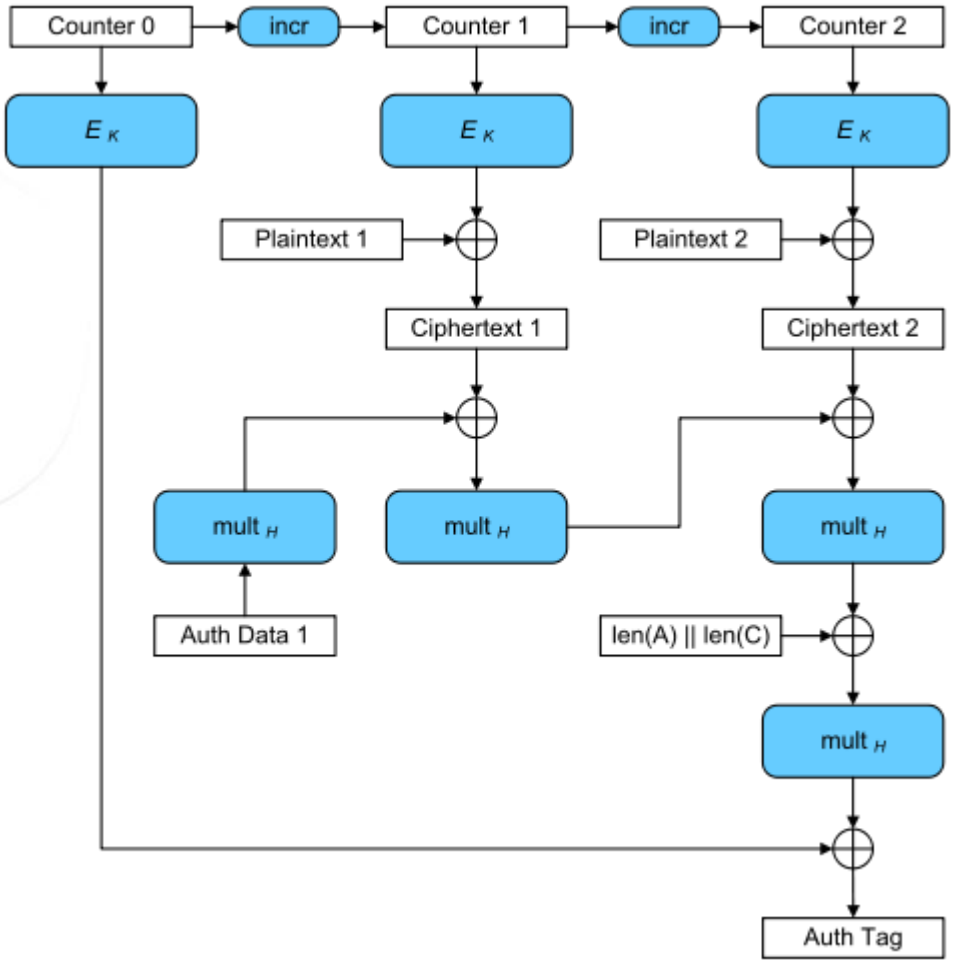
www.cnrs.fr

XTS (XEX-based tweaked-codebook mode with ciphertext stealing)



XEX with tweak and ciphertext stealing (XTS) mode encryption

GCM (Galois Counter Mode)



Confidentialité + **intégrité**
Recommandé (ANSSI)

Générateurs de nombres aléatoires

www.cnrs.fr

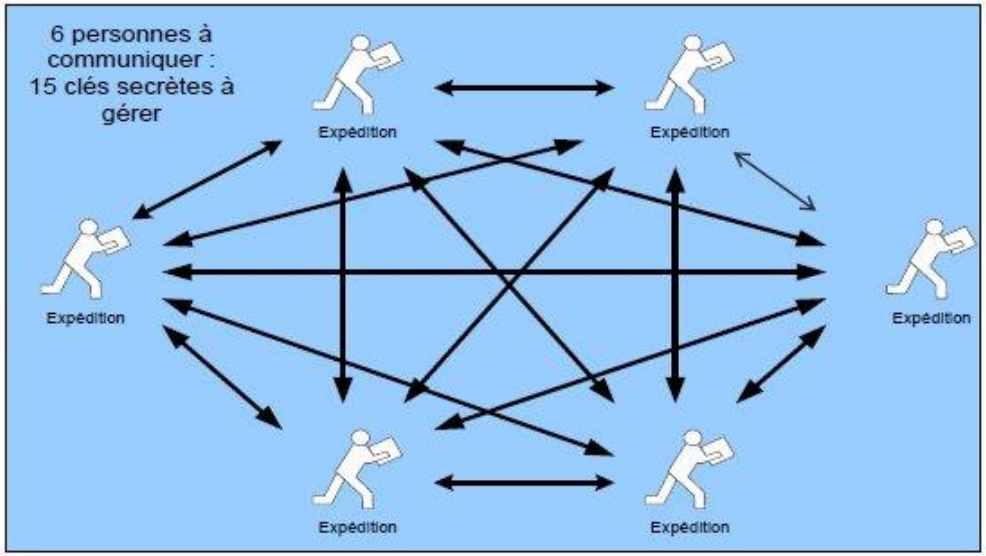
- Générateur aléatoire
  - Phénomène physique : radioactivité, bruit thermique
  - Instructions dans les processeurs Intel récents (bruit thermique)
- Générateur pseudo-aléatoire
  - Lors qu'une personne génère une clé de chiffrement, elle doit faire intervenir le hasard de façon à ajouter de la complexité.
  - De même, certains protocoles cryptographiques nécessitent, pour éviter la re-jouabilité, l'utilisation d'aléas imprévisibles.
  - Mais il est impossible de produire des suites aléatoires uniquement à l'aide d'un ordinateur à cause de sa nature propre.
    - Un générateur sera toujours périodique, donc prévisible
    - On les appelle alors des générateurs pseudo-aléatoires :
      - C'est un algorithme qui génère une séquence de nombres présentant certaines propriétés du hasard.
- Source d'entropie
  - Variations de temps : processus de boot, interruptions
  - Sert à initialiser le générateur pseudo-aléatoire

Générateurs pseudo-aléatoires

- Dual_EC_DRBG basé sur les courbes elliptiques
  - Peu performant
  - Intense lobbying de la NSA
  - Porte dérobée : choix de paramètres et de points particuliers
  - Pare-feu Juniper
    - Utilise Dual_EC_DRBG
    - Découverte d'une porte dérobée qui utilise d'autres paramètres
    - Une porte dérobée peut être utilisée par quelqu'un d'autre
- Confiance limitée dans le générateur matériel des processeurs Intel
  - A utiliser comme source additionnelle d'entropie

Le problème de la distribution des clés

- Il faut pouvoir les transmettre d'une manière sûre.
  - Grand nombre de clés à partager deux à deux entre de nombreuses personnes
  - Nombre de clés à gérer : N(N-1)/2 (N=nb de personnes)



Cryptographie à clé publique

- Appelée aussi cryptographie asymétrique
  - Elle utilise 3 algorithmes :
    - Algorithme de génération des clés
    - Algorithme de chiffrement
    - Algorithme de déchiffrement
- Objectif : résoudre le problème de la distribution des clés
  - Établissement préalable d'un canal pour la transmission de la clef
  - Le principe sera d'utiliser deux clés, une par algorithme (chiffrement/déchiffrement).
  - Et ces deux clés ont la propriété d'être liées l'une à l'autre par un algorithme.

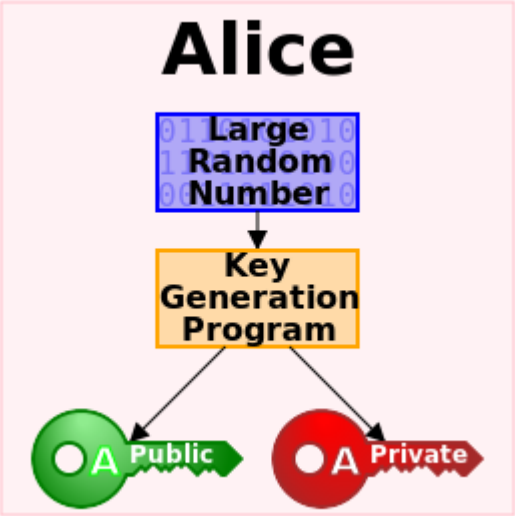
Cryptographie à clé publique

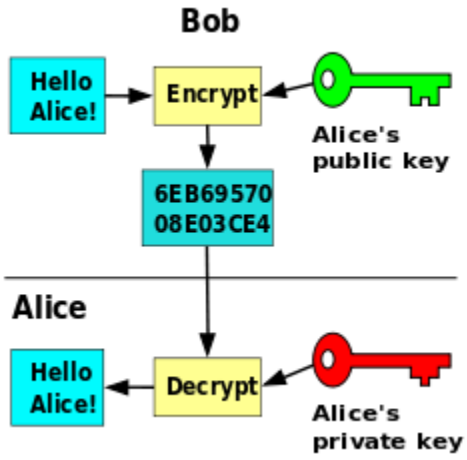
- Algorithmes basés sur des problèmes mathématiques difficiles à résoudre
  - Logarithme discret (Ex : Elgamal)
    - le calcul des logarithmes discrets s'avère difficile, tandis que le problème inverse de l'exponentiation discrète ne l'est pas. C'est une fonction à sens unique.

      Soient g un générateur d'un groupe G d'ordre n (groupe cyclique), et y un élément du groupe G. Le problème du logarithme discret dans G est de trouver $x \in Zn$, tel que $g^x = y$. La valeur x ainsi obtenue est appelée le logarithme discret de y.
  - Factorisation de grands nombres (ex : RSA)
    - Utilisation de fonctions à sens unique;
    - Il est facile de multiplier deux nombres premier pour obtenir un produit, mais difficile de factoriser ce produit afin de retrouver les deux nombres premiers
    - Exemple : quels sont les 2 nombres premiers p et q, dont p x q = 437 ?
  - Courbes elliptiques
    - Plus récent
    - Moins coûteux, clé plus courtes que RSA à robustesse égale
    - NSA et le futur des ordinateurs quantiques

Cryptographie à clé publique

- Fonctionne avec une paire de clés uniques (bi-clés)
  - Une clé privée : connue que du propriétaire de la paire de clé
  - Une clé publique : connue de tous, souvent publié dans un annuaire



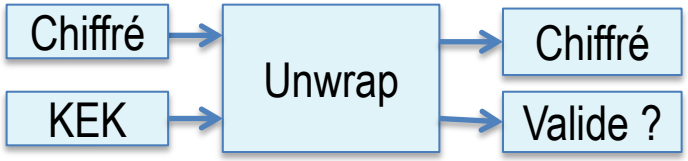


Signature

- Alice calcule l'empreinte du document
- Alice chiffre cette empreinte avec sa clé privée → signature
- Alice transmet à Bob le document avec sa signature
- Bob récupère la clé publique d'Alice
- Bob déchiffre la signature à l'aide de cette publique
- Bob calcule l'empreinte du document et la compare à celle issue du déchiffrement de la signature
- Si identiques le document est intègre et a bien été signé par Alice

AES Key Wrap

- Encapsule en utilisant AES une clé en clair de façon sûre avec contrôle d'intégrité
  - Stockage sur un support non sûr
  - Transfert
- Le taille de la clé encapsulée peut être différente de la taille du bloc
- RFC 3394, spécifications du NIST
- Key encryption key (KEK)



PFS (*Perfect Forward Secrecy*)

- PFS ou confidentialité persistante
  - Propriété qui garantit que la découverte par un adversaire de la clé privée d'un correspondant (secret à long terme) ne compromet pas la confidentialité des communications passées
  - Échanges de clés éphémères par Diffie-Hellmann
- Diffie-Hellmann
  - Dans un groupe fini Alice et Bob se mettent d'accord sur g et p
  - Alice choisit a calcule et envoie à Bob g^a mod p
  - Bob choisit b et calcule et envoie à Alice g^b mod p
  - Bob reçoit g^a et calcule $(g^a)^b$ mod p = g^{ab}
  - Alice reçoit g^b et calcule $(g^b)^a$ mod p = g^{ba} = g^{ab}
  - g^{ab} est utilisé comme clé secrète pour les échanges
  - Vulnérable aux attaques MitM (Man in the Middle)
    - Contremesure : authentification

Certificats

- X509
  - Identité
  - Clé publique
  - Divers attributs
  - Le tout signé par l'autorité de certification
- IGC
  - Hiérarchie d'autorités de certification
    - AC racine ➔ AC 1 ➔ AC 2 ➔ … ➔ utilisateur
    - Transitivité de la confiance

www.cnrs.fr

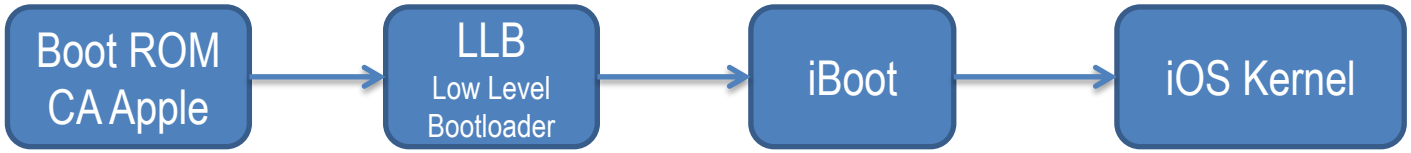
Smartphones

Architecture

- 3 processeurs
  - Chacun a son système
  - Peuvent être sur la même puce (*chip*)
- Processeur baseband
  - Gestion de la radio
  - Premier à démarrer
- Processeur principal
- Processeur sécurité
  - Secure enclave (Apple), versions >= A7 du processeur
    - Générateur de nombres aléatoires matériel
    - Toutes les opérations de gestion de clés
    - Gestion empreintes digitales
  - Trusted Execution Environnment (Android)
    - TrustZone (ARM)
  - TPM (Windows)

Secure boot

- Standard sur les smartphones
- PC : TPM, UEFI, Windows récent
- Vérifie la chaîne de confiance
  - Confiance implicite dans le matériel et le code de la *Boot ROM*
  - Avant d'exécuter une étape on s'assure de l'intégrité du code chargé et de sa validité (signature par le fournisseur)
- Objectifs
  - Interdire attaques : virus MBR, *evil maid*
  - Établir la confiance dans l'OS
- Exemple : Apple



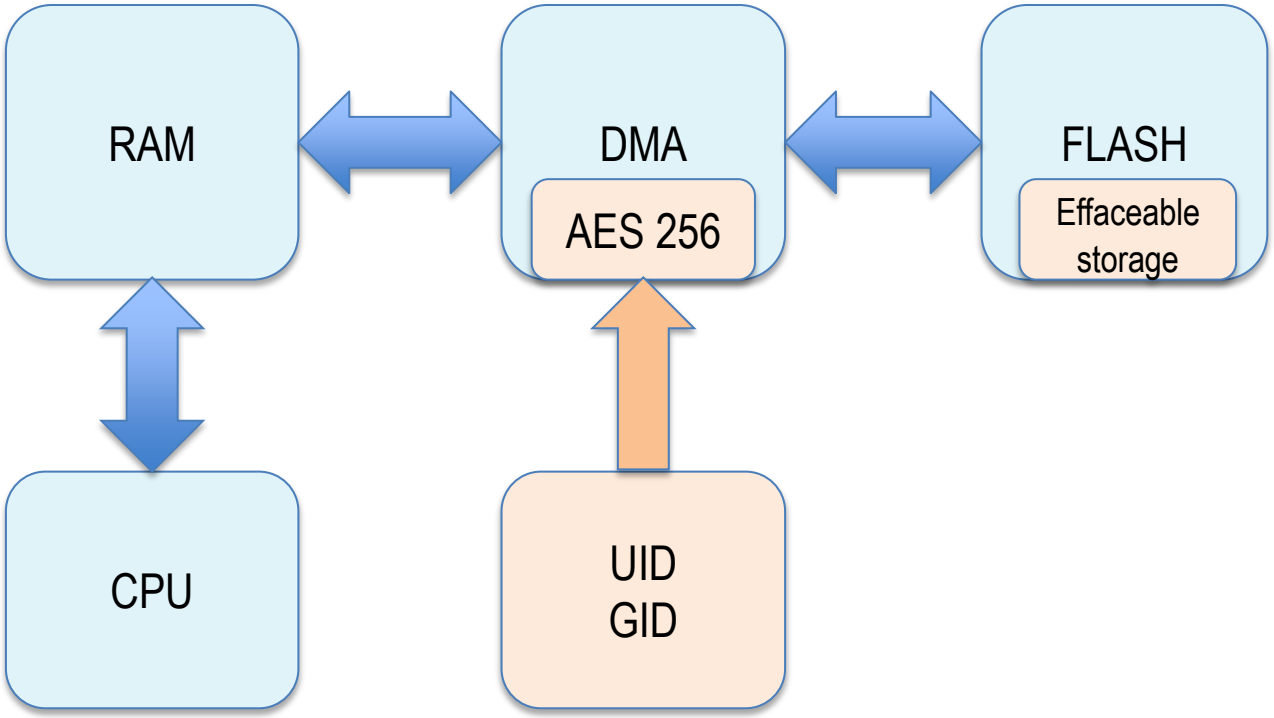
iOS

www.cnrs.fr

iPhone

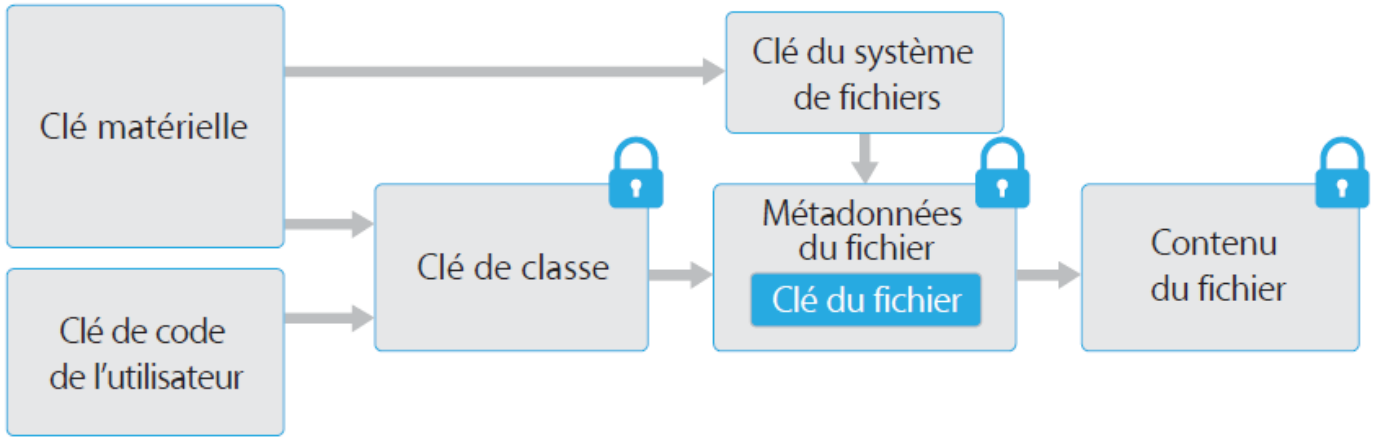
- https://www.apple.com/business/docs/iOS_Security_Guide.pdf
- https://www.apple.com/fr/business/docs/iOS_Security_Guide.pdf

Architecture iPhone



- Clés « gravées » dans le matériel, accessible uniquement par le processeur cryptographique
  - UID : device's unique ID, 256 bits, unique pour chaque iPhone
  - GID : device group ID, 256 bits, unique pour chaque modèle d'iPhone
- *Effaceable storage* : pas de *wear leveling*, un bloc est toujours à la même place

Architecture du chiffrement dans iOS



Chiffrement iOS

- Mémoire flash toujours chiffrée
- AES 256
- Mode CBC ou XTS (processeur A8)
- Vecteur d'initialisation (IV)
- Clé dérivée de la clé matérielle (UID) + …
  - Impossible de lire la flash hors de l'appareil
- File System Key
  - Créée aléatoirement à l'installation d'iOS
  - Stocké en mémoire effaçable
  - Métadonnées des fichiers chiffrée avec cette clé
  - Permet un effacement rapide de la flash

Chiffrement iOS (2)

- File Key
  - Générée aléatoirement à la création du fichier
  - Encapsulé (*wrapped*) avec une class key et stockée dans les métadonnées du fichier
  - Assure une protection en fonctionnement dépendant de la classe
- Class Key
  - Protégée à l'aide de l'UID et du mot de passe, pour certaines classes

Classes de protection

- Complete Protection (NSFileProtectionComplete)
  - Protégée par une clé dérivée du mot de passe et de l'UID
  - Effacée rapidement après verrouillage de l'appareil
  - Il faut entrer le mot de passe ou empreinte digitale pour accéder à nouveau aux informations de cette classe
- Protected Unless Open (NSFileProtectionCompleteUnlessOpen)
  - Permet d'écrire des fichiers alors que l'appareil est verrouillé
  - A la fermeture du fichier, celui-ci devient inaccessible jusqu'au déverrouillage
  - Courbes elliptiques, Diffie-Hellmann

Classes de protection

- Protected Untill First User Authentication (NSFileProtectionCompleteUntilFirstUserAuttentication)
  - Identique à Complete Protection sauf que la clé n'est pas effacée lors du verrouillage
  - Protège contre les attaque qui impliquent un reboot
  - Classe par défaut pour les applications tierces
- No Protection
  - Protégée uniquement par l'UID et stockée dans la mémoire effaçable
  - Permet un effacement rapide

Mot de passe

- Toujours combiné à l'UID
  - Impossible d'effectuer des attaques hors de l'appareil
- PBKDF2
  - Nombre d'itérations tel que 80ms par tentative
  - 6 caractères (minuscules + chiffres) → 5 ans et demi
- Délais croissants entre tentatives infructueuses
  - 1 à 4 : aucun ; 5 : 1 mn ; 6 : 5 mn ; 7 à 8 : 15 mn ; 9 : 1 h
  - Secure Enclave (>= A7) conservé après redémarrage
- Effacement (option) après 10 tentatives infructueuses
- Attaques
  - Manuel ou robot → limites
  - Image spécifique → signée par Apple ou faille (jailbreak)

Trousseau de clés

- Sert à stocker des secrets
  - SQLite, chiffré AES 128 en mode GCM
  - Accessible via le démon *securityd*
    - ACL (groupe d'accès au trousseau, identifiant app, groupe d'app)
    - Partage possible uniquement qu'entre les app d'un même développeur
- Classes, analogues à celles des fichiers
  - Appareil déverrouillé : kSecAttrAccessibleWhenUnklocked
  - Appareil verrouillé : N/D
  - Après authentification initiale : kSecAttrAccessibleAfterFirstUnlock
  - Toujours : kSecAttrAccessibleAlways
  - Code activé : kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly

Conteneurs de clés

- Où son gérés et stockés les clés des classes de protection (données et trousseau)
  - Utilisateur
    - Clés utilisées lors un fonctionnement normal (protégé par mot de passe)
  - Appareil
    - Usage partagé avec plusieurs utilisateurs
    - Clés non protégées par un mot de passe utilisateur
    - Pa défaut non distinct du précédent
  - Sauvegarde
    - Clés utilisées pour les sauvegardes iTune
  - Dépôt
    - Clés utilisées la synchronisation iTune et serveur MDM
  - Sauvegarde iCloud
    - Clés utilisées pour les sauvegardes iTune

Android

www.cnrs.fr

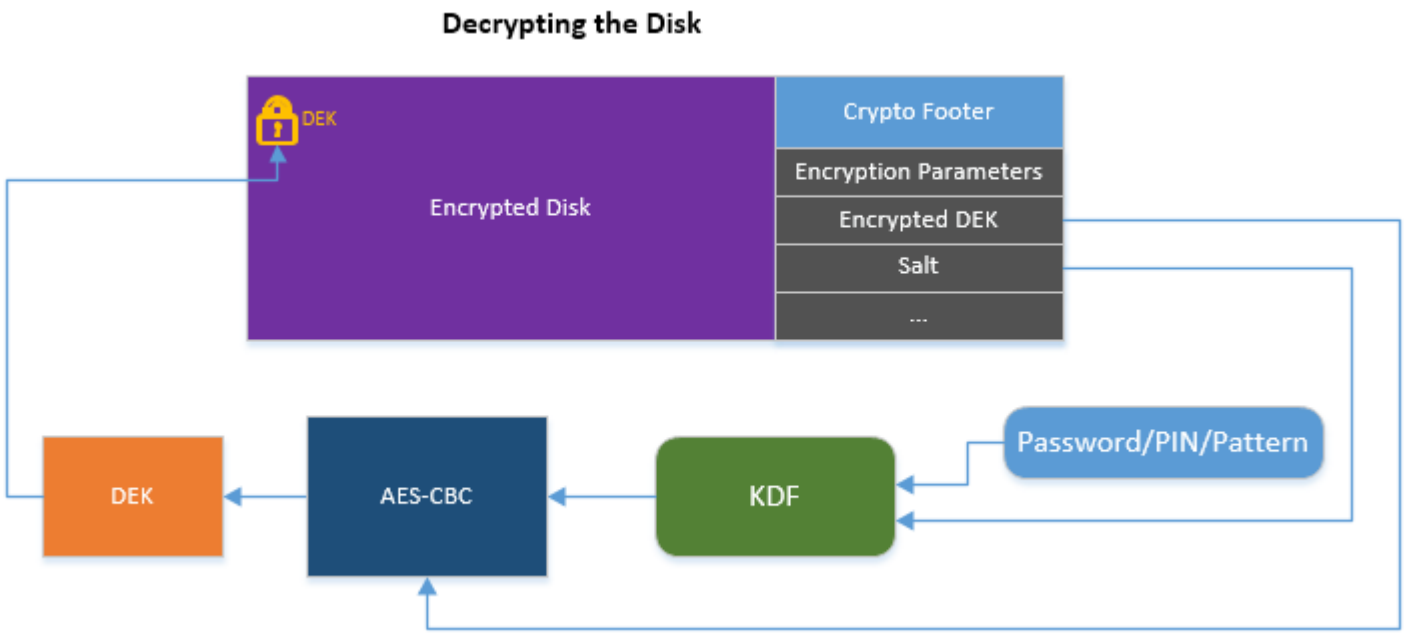
Chiffrement Android

- https://source.android.com/security/encryption/
- https://nelenkov.blogspot.fr/2014/10/revisiting-android-disk-encryption.html#!/2014/10/revisiting-android-disk-encryption.html
- http://www.ssi.gouv.fr/administration/certification_cspn/sous-systeme-de-chiffrement-de-disques-dm-crypt-noyau-linux-4-4-2-cryptsetup-1-7-0/
- http://bits-please.blogspot.fr/2016/06/extracting-qualcomms-keymaster-keys.html

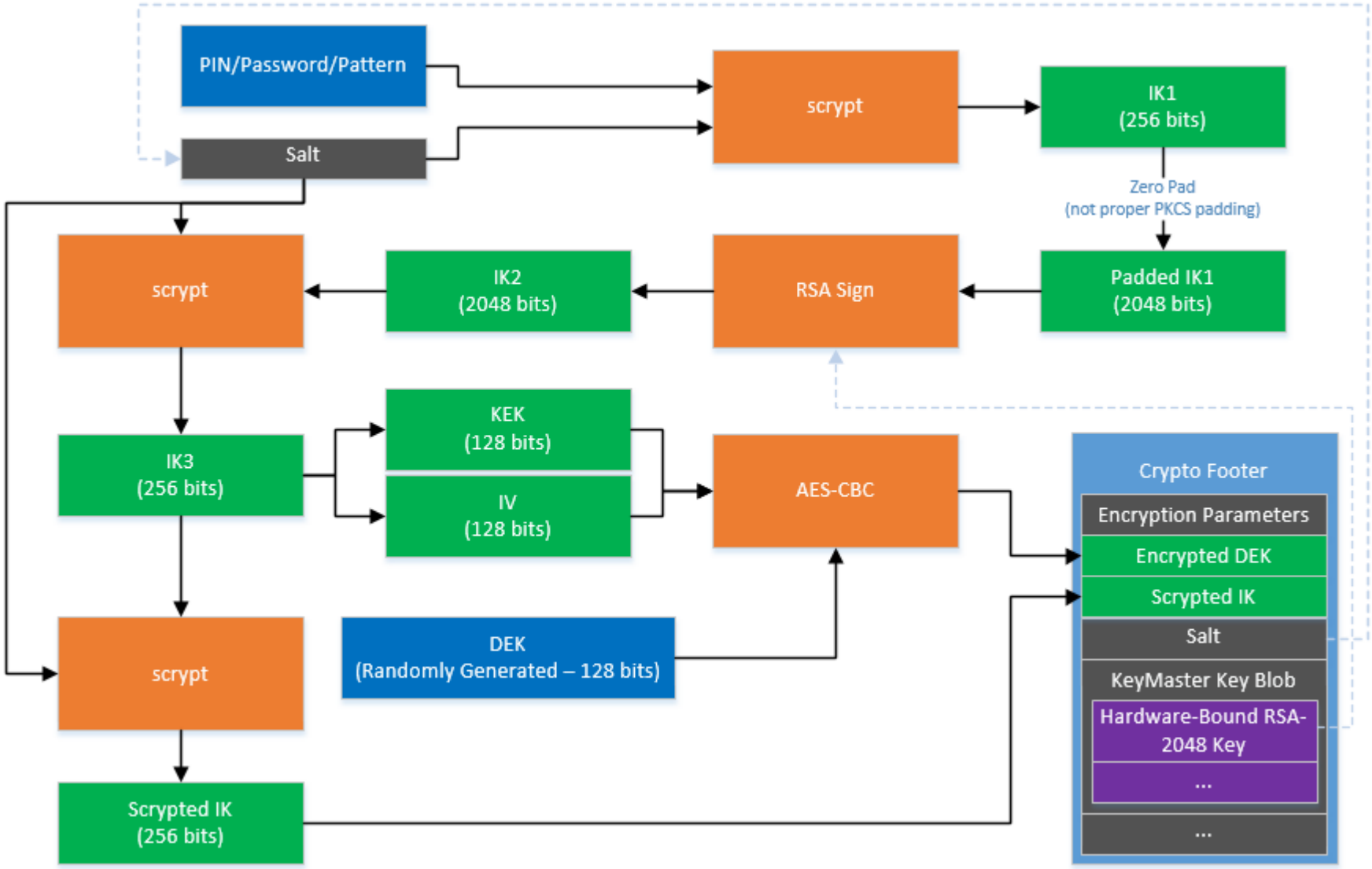
Chiffrement Android

- Basé sur dm-crypt
  - Chiffrement intégral du disque (*full disk encryption* ou FDE)
  - AES 128 bits
    - Mode CBC
    - Vecteur d'initialisation : ESSIV:SHA256
- Clé maître→ DEK (device encryption key)
  - 128 bits généré aléatoirement
  - Chiffrée à l'aide d'une clé obtenue par dérivation
    - Sel (128 bits, généré aléatoirement)
    - Mot de passe
- Prévention des attaques
  - Délai entre tentatives
  - Effacement après un certain nombre de tentatives infructueuses
  - Fonction de dérivation coûteuse à calculer
    - Et liée au matériel

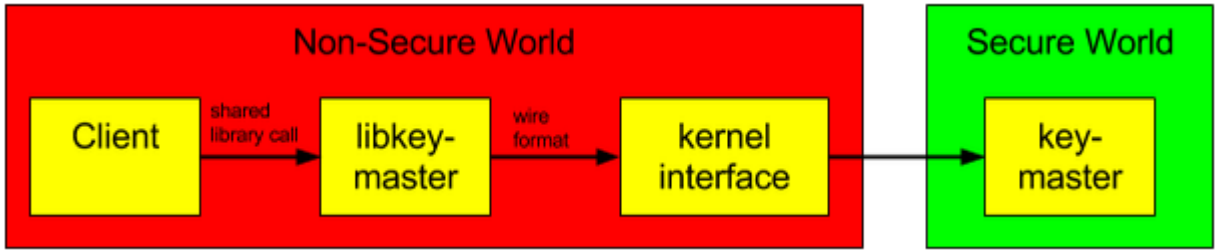
Chiffrement Android



Chiffrement Android



Hardware-backed Keystore



- Les clés sont générées et les opérations cryptographiques sensibles sont effectuées dans le *Secure World* sans jamais révéler les clés au *Non-Secure World*
- SHK clé hardware
- Faille dans la TrustZone et utilisation d'une clé maître dérivée de la SHK visible dans la TrustZone permet une attaque sur le chiffrement

Windows

www.cnrs.fr

Chiffrement Windows

- Bitlocker
  - AES 128 ou 256 bits
  - CBC ou XTS
  - Choisir XTS-AES-256
- Clés
  - FVEK (Full Volume Encryption Key), clé AES de chiffrement du disque
  - VMK (Volume Master Key)
    - Sert à chiffrer la FVEK
    - La FVEK est stockée chiffrée par la VMK dans les métadonnées du disque
    - Rapide de changer la VMK sans chiffrer à nouveau tout le disque
- Keys protectors
  - Servent à protéger la VMK
  - TPM si boot intègre → VMK → FVEK
  - TPM + PIN si boot intègre + PIN valide → VMK → FVEK
  - Recovery key : clé de recouvrement stockée dans un clé USB ➔ à protéger
  - Recovery password : mot de passe de recouvrement ➔ à protéger

Robustesse du chiffrement

- Attaques sur le mot de passe ➔ contre mesures efficaces
  - Délai entre tentative
  - Limite de tentatives
  - Coûteux à calculer
  - Lié à l'appareil
- Attaques envisageables
  - Présence de portes dérobées
    - Initiative de constructeur ➔ quel intérêt ?
    - Initiative d'un service étatique ➔ ?
  - Attaques matérielles
    - Canaux auxiliaires (ex. consommation)
    - Fautes
    - Décorticage du circuit
  - Failles conception, réalisation

Chiffrement

- iOS
  - Protection au niveau du fichier
- Android, Windows
  - FDE
  - Possibilités de chiffrement de fichiers (Android 7.0)

www.cnrs.fr

Applications

Cryptographie dans les applications

- État des lieux ➔ catastrophique
  - Analyse des 1000 applications Android les plus téléchargées
    - 73% de celles qui communiquent avec un serveur ne vérifient pas son certificat
    - 8% ne vérifient pas le nom de l'hôte
    - 77% de celles qui utilisent Webkit ignorent les erreurs SSL
    - 68% ont au moins une de ces 3 failles SSL/TLS
    - Permet des attaques Man in the Middle
- On trouve tout ce qu'il ne faut pas faire
  - Top 10 OWASP
  - Mot de passe en clair
  - Non vérification des certificats
  - Mauvaise utilisation d'un jeton
  - Etc.

Les raisons d'un échec

- La cryptographie et sa mise en œuvre sont intrinsèquement compliquées
- Les API sont complexes, manquent de cohérence, sujettes à erreur
- Les développeurs récupèrent n'importe quel code sur Internet à l'aide d'un moteur de recherche
- Les développeur suppriment les contrôles pour la mise au point et ne les rétablissent pas pour la production
- *Time to market* ➔ impasse sur la sécurité

Bonnes pratiques de développement

- Ne jamais chercher à développer un nouvel algorithme ou protocole cryptographique ➔ laisser cela aux spécialistes
- Ne pas réinventer la roue ➔ utiliser des outils, des bibliothèques éprouvées
- Les systèmes offrent des fonctionnalités intéressantes en matière de sécurité ➔ les utiliser
  - Classes de chiffrement dans iOS
  - Utiliser les mécanismes de gestions des secrets fournis par tous les OS (Trousseau de clés, magasins, etc.)
  - Ne pas se contenter des valeurs par défaut
- Mettre en œuvre les recommandations du RGS
  - Taille de clé
  - Algorithmes
  - Gestion des secrets d'authentification

www.cnrs.fr

Perspectives

Ouverture vers l'Internet des objets

www.cnrs.fr

- On ne peut ignorer les objets connectés
  - De plus en plus présents
  - On en utilise
  - On en produit
- Contraintes spécifiques
  - Puissance de calcul limitée
  - Consommation électrique (batterie)
  - Réseau à faible débit
  - Maintenance, mises à jour, fin de vie
- Besoins de sécurité
  - D'abord intégrité et authenticité
  - Avant confidentialité
  - Risques humains
- Un défi qu'il faut relever

Futur de la cryptographie

www.cnrs.fr

- Chiffrement homomorphe
  - Permet d'effectuer des opérations sur les données chiffrées
  - Encore très peu performant (CPU, mémoire)
- Cryptographie post ordinateurs quantiques
  - Logarithme discret, RSA, courbes elliptiques ne résistent pas : algorithme de Shor
  - Mise en garde NSA
  - AES : taille de la clé divisée par 2
  - Algorithmes à l'étude