M(alléable)EGA

SSE - Séminaire en sécurité

Alexis Martins

HEIG-VD

1. Background

Histoire

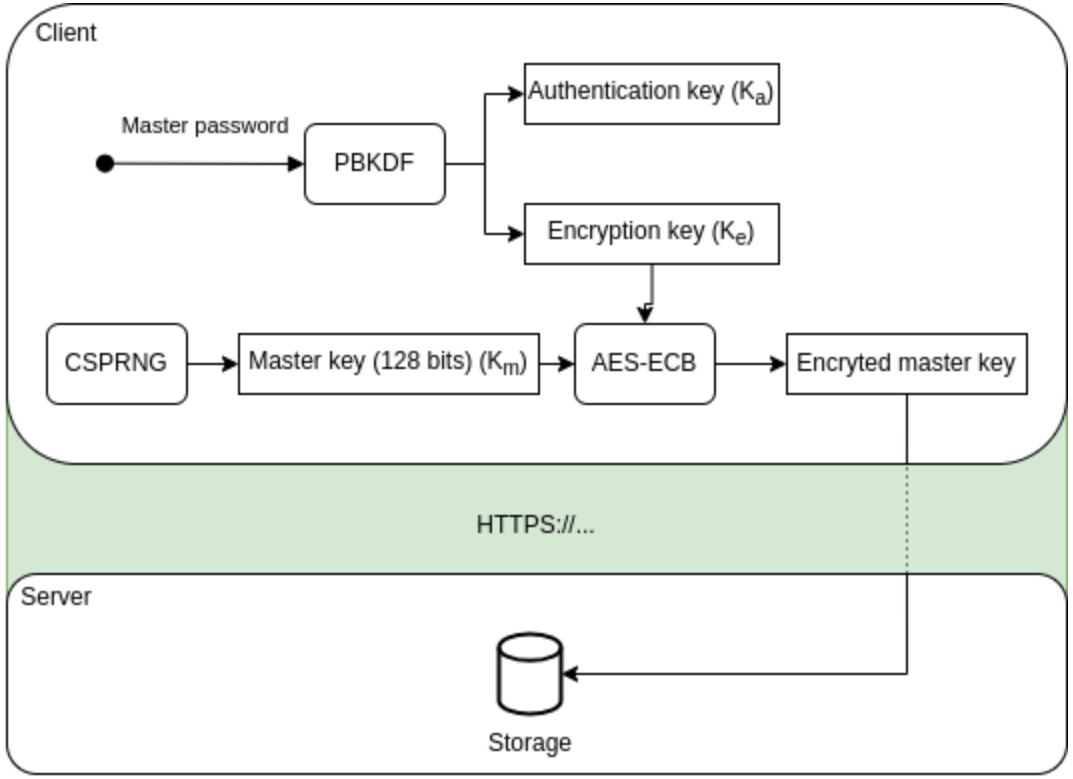
- Fournisseur de services cloud (stockage)
- Fondé en 2013
- "Privacy company"
  - Chiffrement de bout en bout

Quelques chiffres

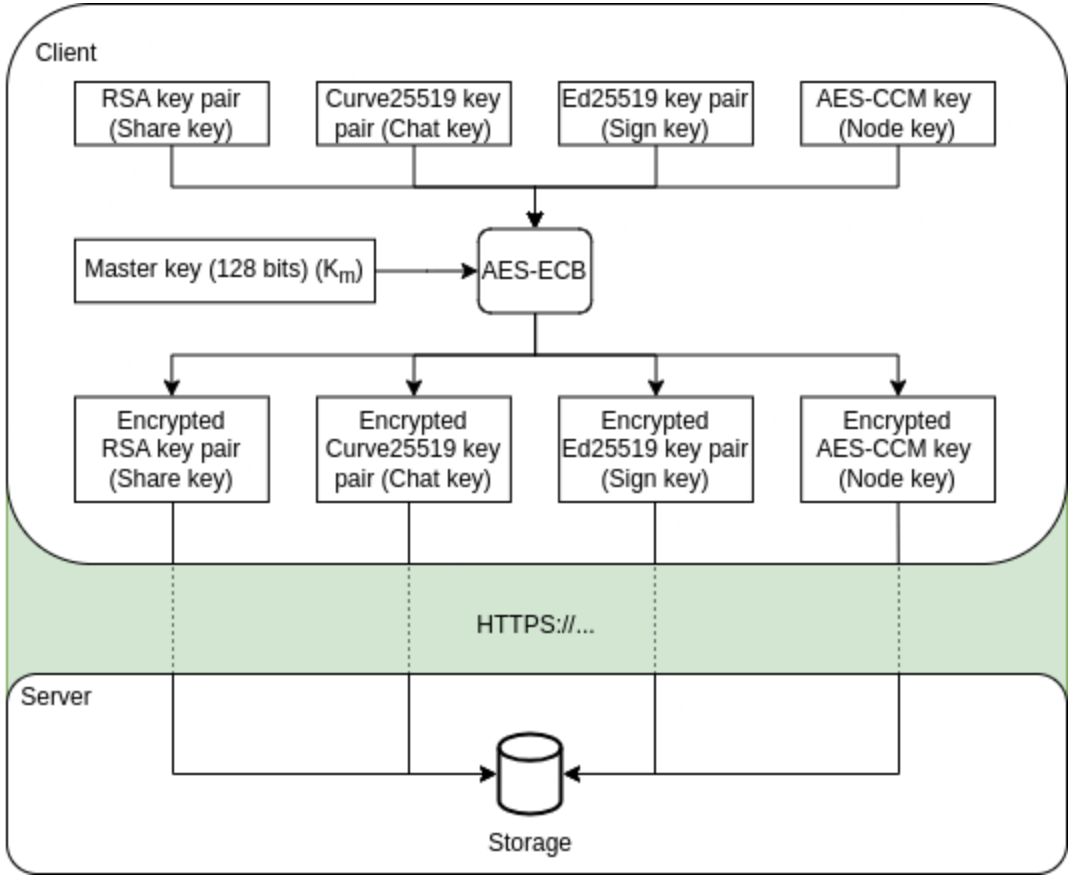
- 250 millions d'utilisateurs au total
- 10 millions d'utilisateurs quotidiens
- 1000 PB de données stockées

2. Cryptographie

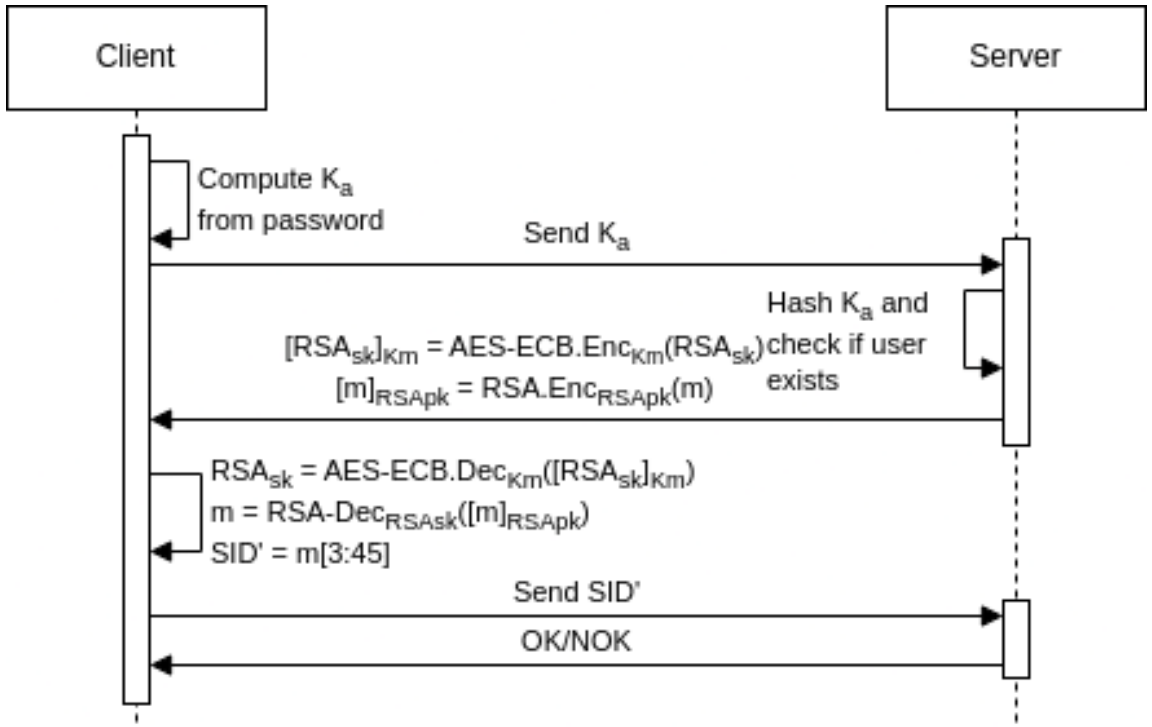
Clés (1)


Client
Master password
PBKDF
Authentication key (K_a)
Encryption key (K_e)
CSPRNG
Master key (128 bits) (K_m)
AES-ECB
Encryted master key
HTTPS://...
Server
Storage

Clés (2)


Client
RSA key pair (Share key)
Curve25519 key pair (Chat key)
Ed25519 key pair (Sign key)
AES-CCM key (Node key)
Master key (128 bits) (K_m)
AES-ECB
Encrypted RSA key pair (Share key)
Encrypted Curve25519 key pair (Chat key)
Encrypted Ed25519 key pair (Sign key)
Encrypted AES-CCM key (Node key)
HTTPS://...
Server
Storage

Connexion (1)


Client
Server
Compute K_a from password
Send K_a
Hash K_a and check if user exists
$[RSA_{sk}]_{Km} = AES\text{-}ECB.Enc_{Km}(RSA_{sk})$
$[m]_{RSApk} = RSA.Enc_{RSApk}(m)$
$RSA_{sk} = AES\text{-}ECB.Dec_{Km}([RSA_{sk}]_{Km})$
$m = RSA\text{-}Dec_{RSAsk}([m]_{RSApk})$
SID' = m[3:45]
Send SID'
OK/NOK

Connexion (2)

Récupération de la clé privée :

$$RSA_{sk}^{encoded} = \text{AES-ECB.Dec}_{K_m}([RSA_{sk}]_{K_m})$$
$$RSA_{sk}^{encoded} = len(q)||q||len(p)||p||len(d)||d||len(u)||u||padding$$

Connexion (3)

Récupération du SID' (RSA-CRT) :

DecSid($[sk_{share}^{encoded}]_{k_M}$, $[m]_{pk_{share}}$):

Given: encrypted RSA private key $[sk_{share}^{encoded}]_{k_M}$, encrypted message $[m]_{pk_{share}}$
Returns: decrypted and unpadded SID sid'

1 $sk_{share}^{encoded} \leftarrow$ AES-ECB.Dec(k_M, $[sk_{share}^{encoded}]_{k_M}$)
2 $N, e, d, p, q, d_p, d_q, u \leftarrow$ DecodeRsaKey($sk_{share}^{encoded}$)
3 $m'_p \leftarrow ([m]_{pk_{share}})^{d_p} \bmod p$
4 $m'_q \leftarrow ([m]_{pk_{share}})^{d_q} \bmod q$
5 $t \leftarrow m'_p - m'_q \bmod p$
6 $h \leftarrow t \cdot u \bmod p$
7 $m' \leftarrow h \cdot q + m'_q$
8 $sid' \leftarrow m'[3{:}45]$ // Unpad 43 B SID.
9 **return** sid'

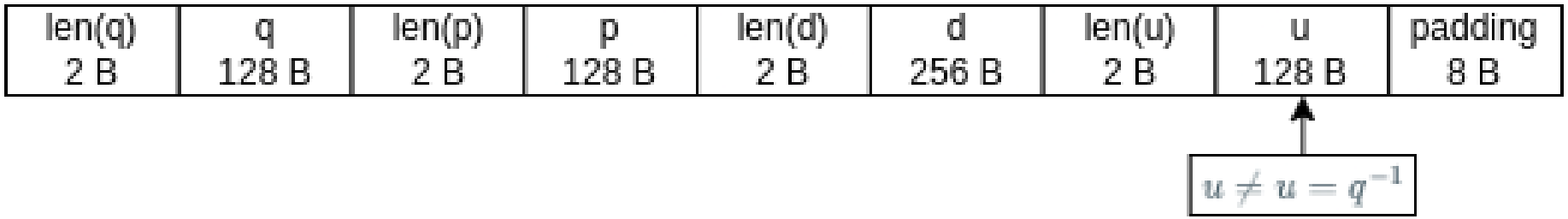
3. Attaques

Récupération de la clé privée RSA

Récupération de la clé privée RSA

- Attaquant contrôle le serveur
- Case distinction oracle
- But : Récupérer la composante `q` pour retrouver la clé privée

Key overwritting



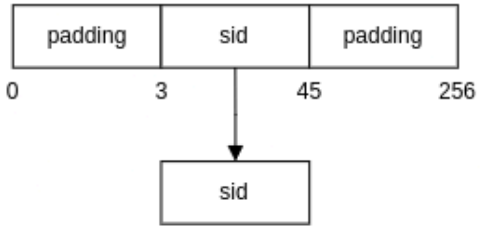
- Chiffré avec AES-ECB
- Pas de contrôles d'intégrité

RSA-CRT

Normal case :

$m_p = [m]_{pk}^{dp} \bmod p$
$m_q = [m]_{pk}^{dq} \bmod q$

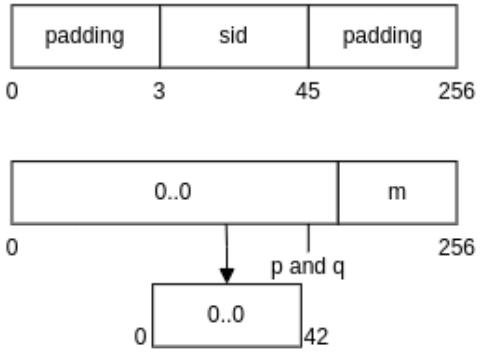
$t = m_p - m_q \bmod p$
$h = t * u \bmod p$
$m = h * q + m_q$



Case m < q :

$m_p = [m]_{pk}^{dp} \bmod p = m$
$m_q = [m]_{pk}^{dq} \bmod q = m$

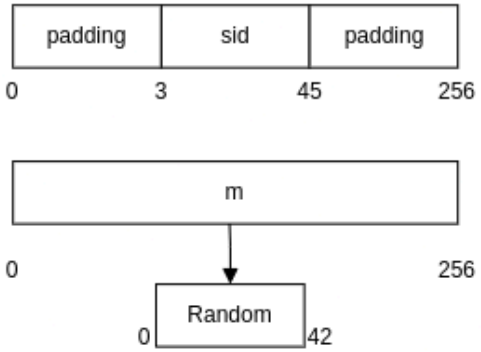
$t = m_p - m_q \bmod p = 0$
$h = 0 * u \bmod p = 0$
$m = 0 * q + m_q = m_q$



Case m >= q:

$m_p = [m]_{pk}^{dp} \bmod p$
$m_q = [m]_{pk}^{dq} \bmod q$

$t = m_p - m_q \bmod p$
$h = t * u \bmod p$
$m = h * q + m_q$



Fonctionnement

- Recherche dichotomique sur q
  - Si sid est nul, alors m < q
  - Si sid est aléatoire, alors m >= q
- Avec q, on récupère p
  - $p = N/q$
- Environ 1023 connexions de l'utilisateur

Impacts

- Confidentialité des partages compromise
  - Partage de nodes
  - Certaines clés du MegaChat
- Porte d'entrée pour d'autres attaques

Améliorations

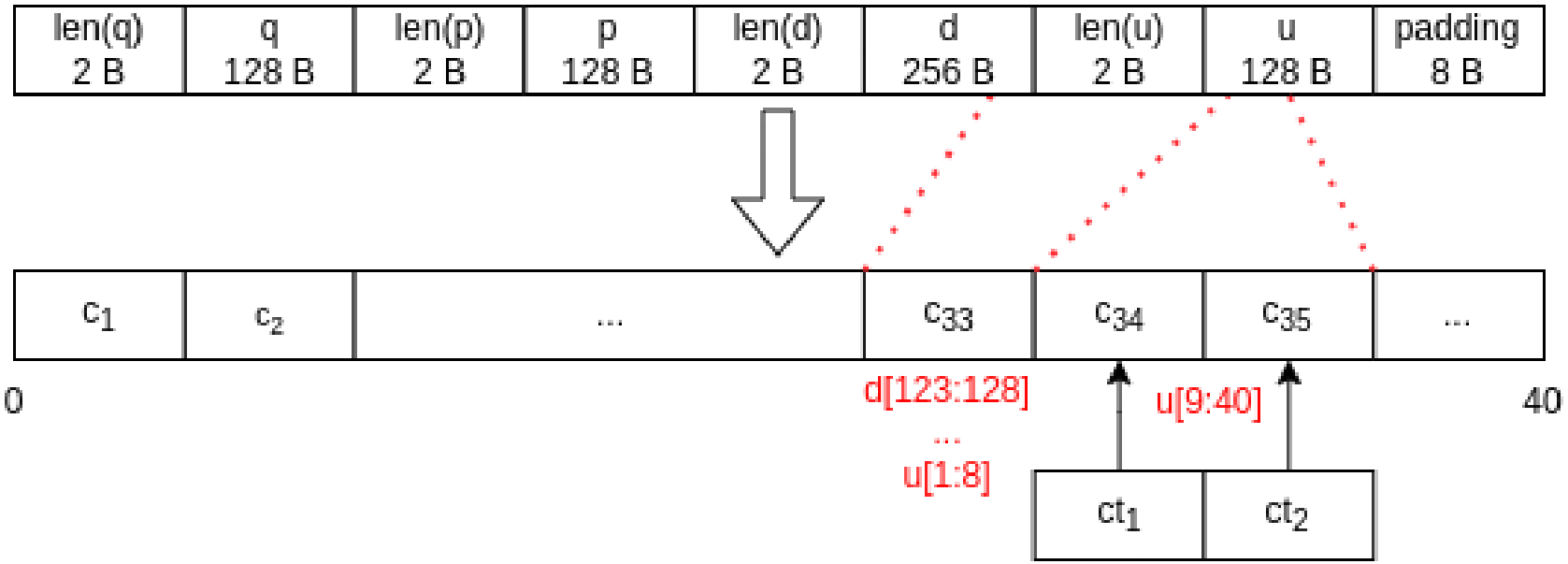
- 512 connexions : Utilisation de Lattice
- 6 connexions : Utilisation de Lattice plus poussée
- 2 connexions : Ajout d'un second vecteur d'attaque

Récupération de textes clairs

Récupération de textes clairs

- Attaquant contrôle le serveur + possède RSA_{sk}
- Rappel : Toutes les clés sont chiffrées avec la `même` clé et AES-ECB
- But : Récupération des autres clés stockées chiffrées sur le serveur
  - L'attaquant demande au client de déchiffrer ces clés
  - Une clé récupérée par connexion

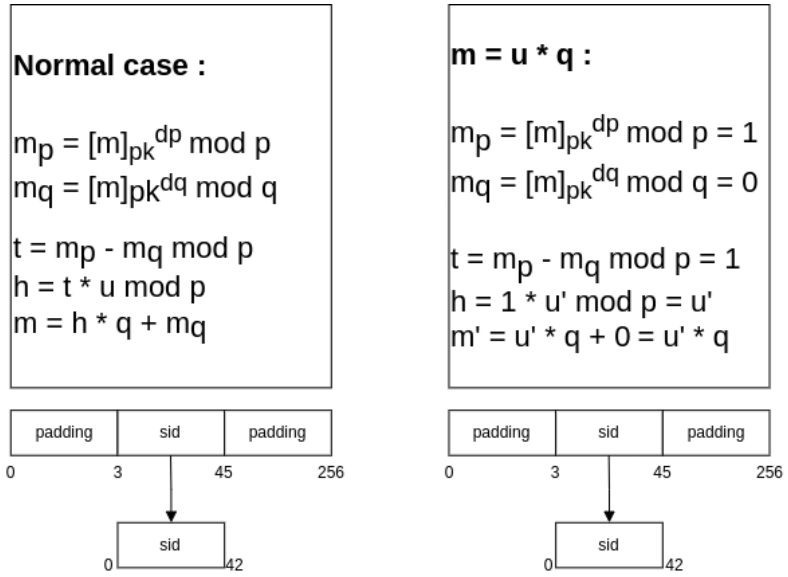
Key overwritting



- ct_1 et ct_2 sont des clés récupérées sur le serveur
- $u \rightarrow u' = u_1||x||u_2$, où x sont les 32 bytes de clés et u_x sont le reste des bits de u

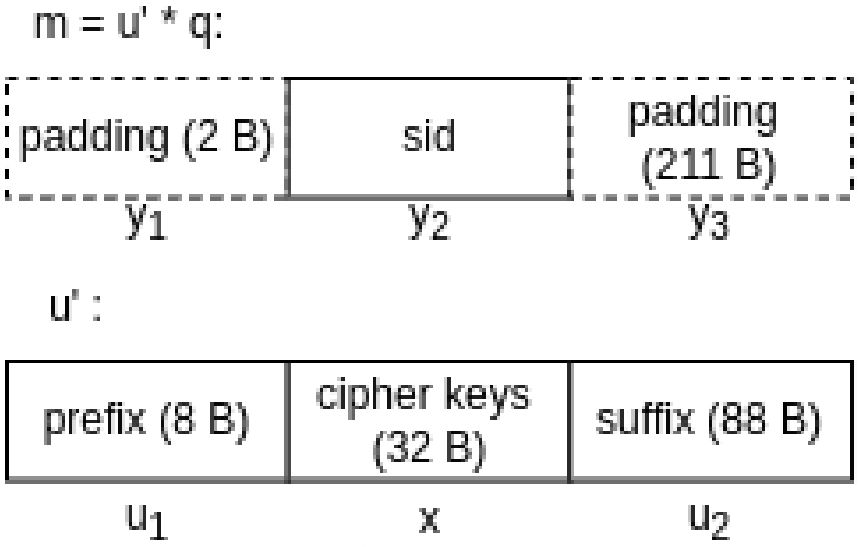
RSA-CRT simplifié

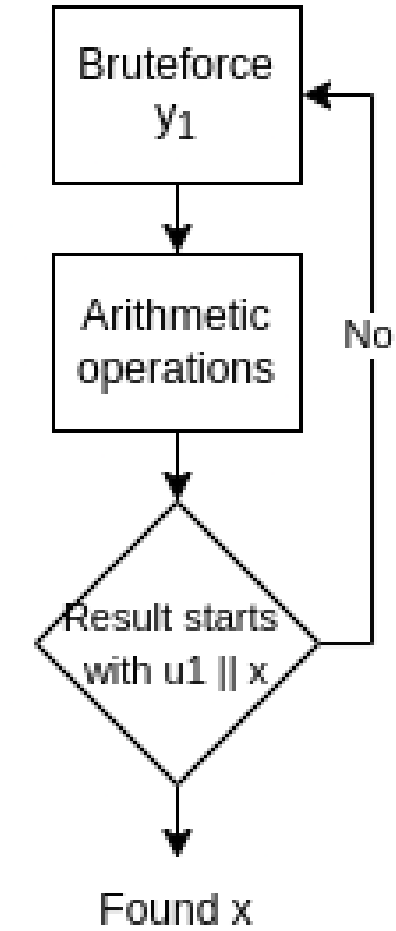
Choisir un m spécifique, tel que $m = u \cdot q$



Le but de l'attaquant est d'avoir la valeur de m' pour récupérer u' qui contient les blocs modifiés et déchifrrés

Récupérer le texte clair





Récupérer le texte clair (détails)

Les variables du sytème :

$$m' = y_1 \| y_2 \| y_3 = u' \cdot q$$

$$u' = u_1 \| x \| u_2$$

Les transformations arithmétiques:

$$\hat{y}_1^* \| y_2 \cdot 256^{211} + y_3 = (u_1 \| x \cdot 256^{88} + u_2) \cdot q$$

$$\frac{\hat{y}_1^* \| y_2 \cdot 256^{211}}{q} = u_1 \| x \cdot 256^{88} + u_2 - \frac{y_3}{q}$$

$$u_1 \| x = \left\lfloor \frac{\hat{y}_1^* \| y_2 \cdot 256^{211}}{q} \cdot 256^{-88} \right\rfloor$$

Négligeable

Impacts

- Permet à l'attaquant d'accéder à `tout` :
  - Fichiers
  - Conversations
  - Partages

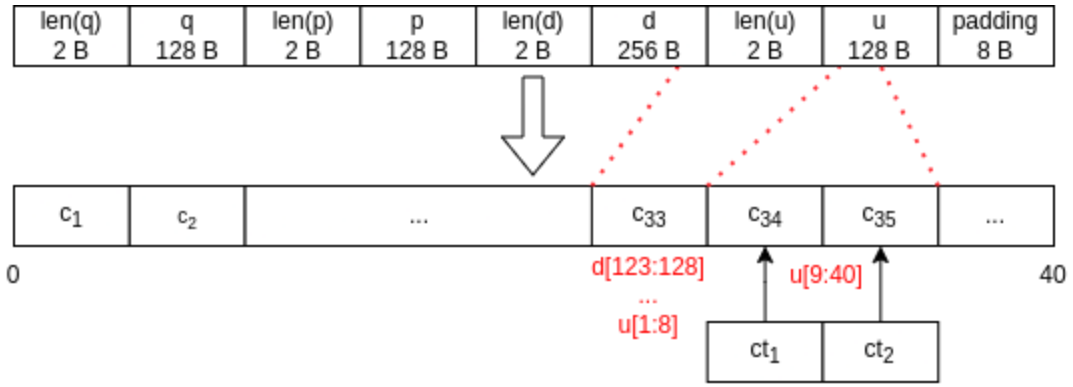
Attaques sur l'intégrité

Attaques sur l'intégrité

- Attaquant contrôlant le serveur + possède un oracle de déchiffrement
- But : Pouvoir ajouter des nouveaux fichiers
- 2 scénarios possibles
  - Oracle de déchiffrement
  - Plaintext-Ciphertext

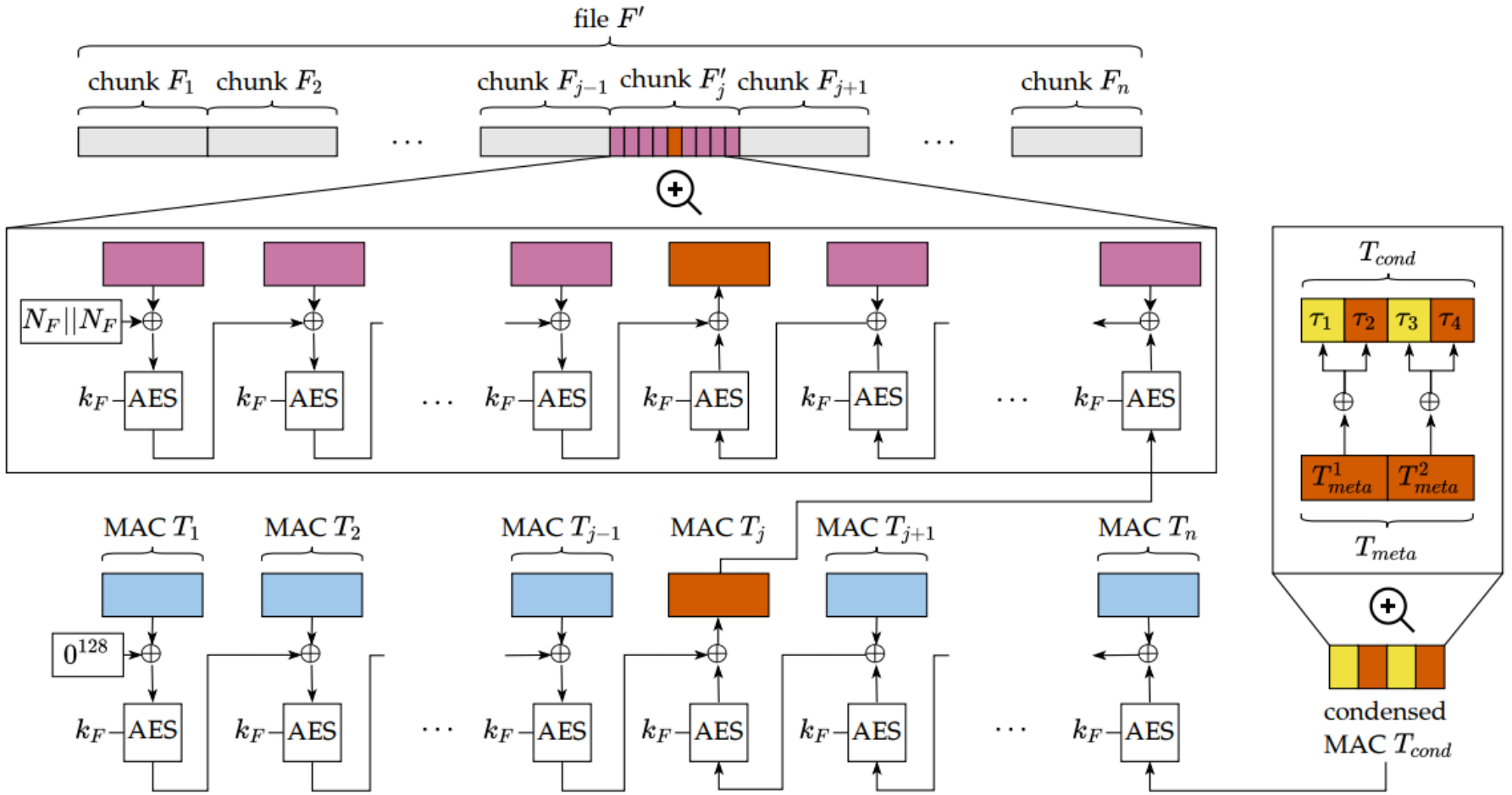
Oracle de déchiffrement

- Création de nouvelles clés de chiffrement "chiffrées"
- Demander au client de les déchiffrer



Nouvelle clé (k_F), nonce(N_F) et tag(T_{meta}) similaires aux originaux que l'on obtient après désobfuscation des valeurs récupérées

Reconstruction de fichiers



Reconstruction de fichiers (détails)

- L'attaquant veut reconstruire un fichier pour qu'il soit valide pour le tag T_{meta}
- Modification du fichier en partant du MAC final T_{cond}
  - L'attaquant identifie un endroit dans le fichier où il peut ajouter un bloc AES à un chunk sans rendre le fichier illisible
  - Meet-in-the-middle réalisé en calculant le tag condensé de T_1 à T_{j-1} ($T_{cond,j-1}$) et celui de T_n à T_j ($T_{cond,j}$)
  - Calcul de la valeur de T_j pour que $T_j = T_{cond,j-1} \oplus \text{AES-ECB.Dec}(T_{cond,j+1})$, puis adapté bloc AES et son chunk pour avoir ce tag

Impacts

- Ajout de fichiers arbitraires
  - Fichiers peuvent être compromettants afin d'incriminer la victime

GaP-Bleichenbacher

Attaque de Bleichenbacher (1)

- Attaque sur RSA PKCS#1 v1.5
- Adaptative chosen-ciphertext
- Format d'un message : 00 || 02 || RAND_PAD || 00 || MESSAGE

Attaque de Bleichenbacher (2)

- $c = m^e \mod N$ où m = 0x0002...
- Attaquant récupère c et le multiplie par s^e, $c' = (m \cdot s)^e \mod N$
  - $m \cdot s \in [0x0002\ldots; 0x0003\ldots)$

$$O(c) = \begin{cases} 1 & \text{if } m = c^d \mod N \text{ starts with } 0x0002 \\ 0 & \text{otherwise} \end{cases}$$

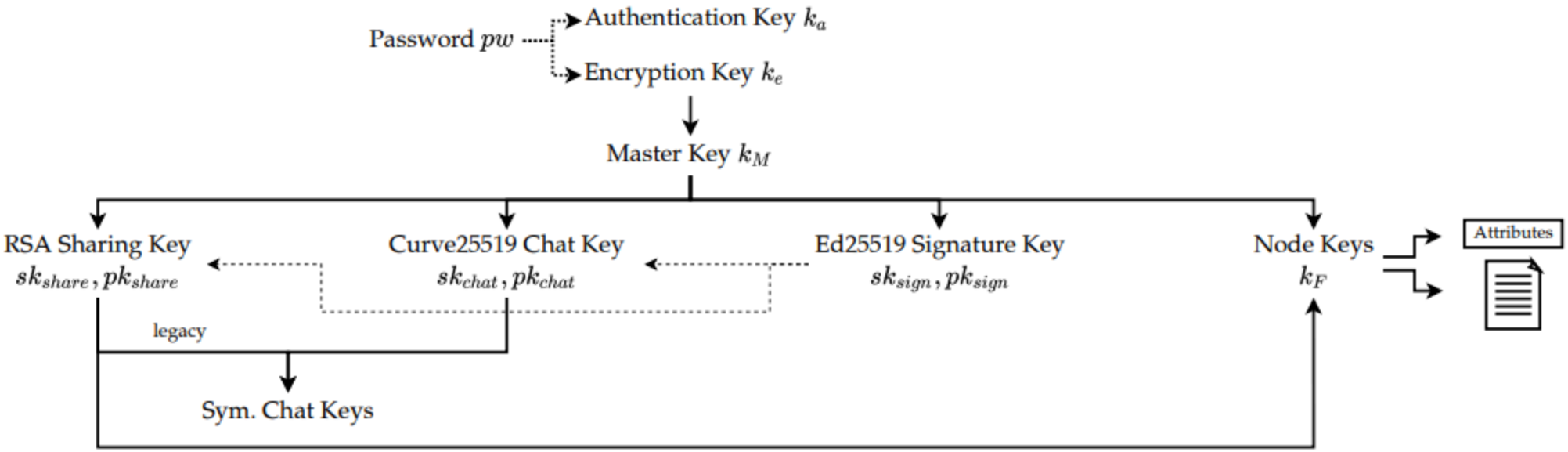
- Après environ 1000 messages, il est possible de retrouver `m`

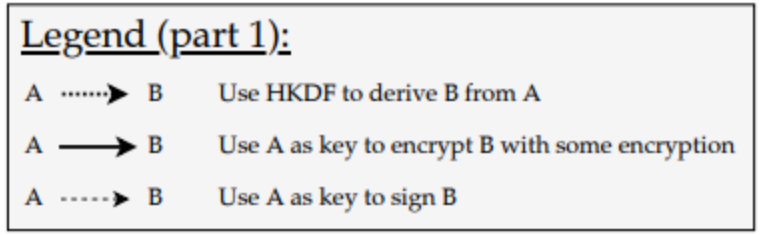
GaP-Bleichenbacher

- Deux scénarios :
  - Attaquant se faisant passer pour le fournisseur de service
  - Attaquant étant un utilisateur ayant un chat avec la victime
- But : Récupérer les clés partagées à l'aide de RSA
- Padding des messages moins rigide.
  - Déchiffrement réussi pour différents padding (intervalles)
  - Avec plusieurs requêtes, on élimine les intervalles sans solutions

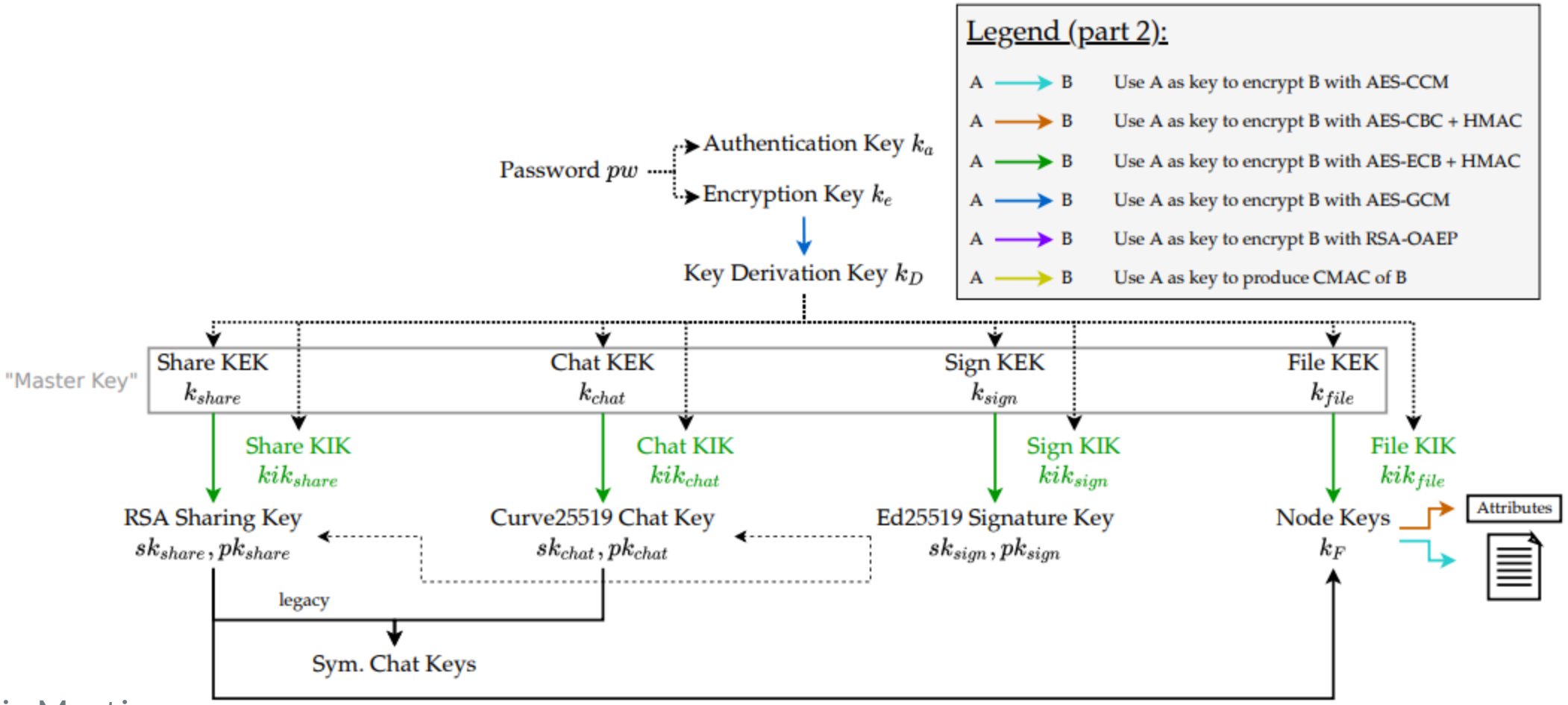
4. Contre-mesures

Etat actuel

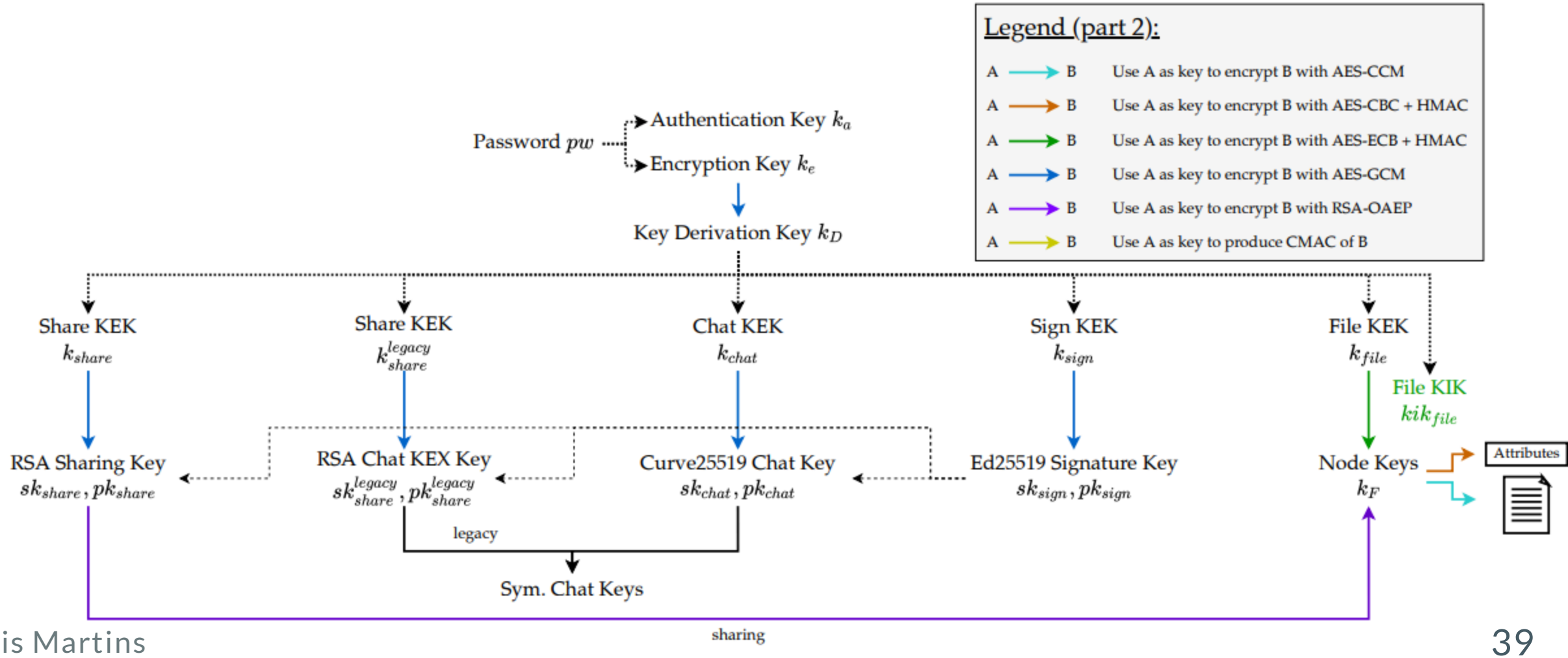

Password pw
Authentication Key k_a
Encryption Key k_e
Master Key k_M
RSA Sharing Key
sk_{share}, pk_{share}
Curve25519 Chat Key
sk_{chat}, pk_{chat}
Ed25519 Signature Key
sk_{sign}, pk_{sign}
Node Keys
k_F
Attributes
legacy
Sym. Chat Keys


Legend (part 1):
A ⋯▶ B Use HKDF to derive B from A
A ⟶ B Use A as key to encrypt B with some encryption
A ⋯▶ B Use A as key to sign B

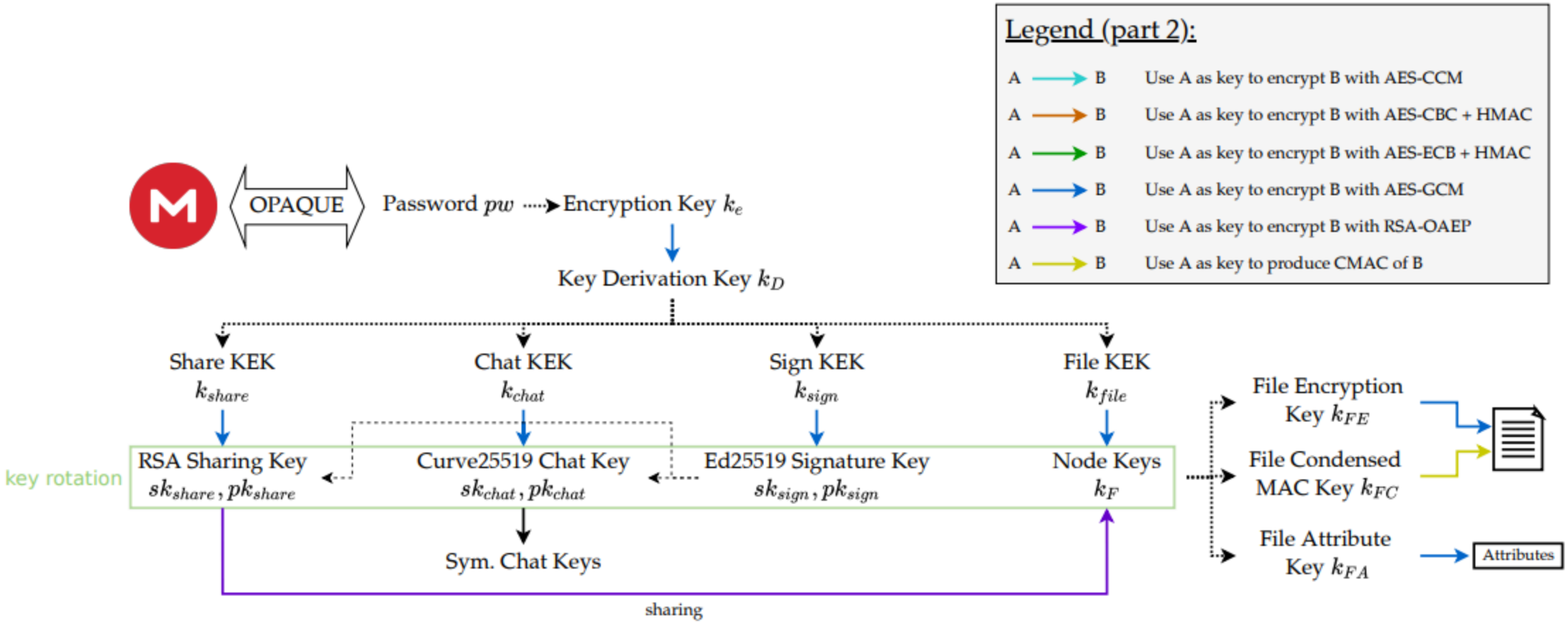
Contre-mesures immédiates



Contre-mesures minimales



Contre-mesures recommandées



5. Conclusion

Conclusion

- Prioriser le chiffrement authentifié
- Séparation des clés
- Penser à un système évolutif
- Don't roll your own crypto


HACKING
MEGA
imgflip.com

Références

- Articles résumé sur le site
- Papier scientifique avec tous les détails
- Vidéo YouTube résumant l'attaque