

# Aionda Mail Security Whitepaper

Zero-Knowledge · Post-Quantum · Made in Germany

---

**Version 1.0** — March 2026

**Aionda GmbH**  
Stuttgart, Germany

[contact-46epp9ba@contact.aionda.com](mailto:contact-46epp9ba@contact.aionda.com)  
<https://mail.aionda.com>

PUBLIC DOCUMENT

## Contents

---

|          |                                                                      |          |
|----------|----------------------------------------------------------------------|----------|
| <b>1</b> | <b>Livre blanc de securite Aionda Mail</b>                           | <b>5</b> |
| 1.1      | Table des matieres . . . . .                                         | 5        |
| 1.2      | 1. Resume executif . . . . .                                         | 5        |
| 1.3      | 2. Modele de menace . . . . .                                        | 6        |
| 1.3.1    | 2.1 Contre quoi Aionda Mail protege . . . . .                        | 6        |
| 1.3.2    | 2.2 Contre quoi Aionda Mail ne protege PAS . . . . .                 | 7        |
| 1.3.3    | 2.3 Principe de conception . . . . .                                 | 7        |
| 1.4      | 3. Vue d'ensemble de l'architecture . . . . .                        | 7        |
| 1.4.1    | 3.1 Vue d'ensemble des composants . . . . .                          | 8        |
| 1.5      | 4. Authentification Zero-Knowledge (OPAQUE) . . . . .                | 8        |
| 1.5.1    | 4.1 Pourquoi pas le hachage de mot de passe ? . . . . .              | 8        |
| 1.5.2    | 4.2 Fonctionnement d'OPAQUE . . . . .                                | 9        |
| 1.5.3    | 4.3 Implementation . . . . .                                         | 9        |
| 1.5.4    | 4.4 Migration SRP . . . . .                                          | 10       |
| 1.6      | 5. Cle maitre du coffre-fort & Shamir Secret Sharing . . . . .       | 10       |
| 1.6.1    | 5.1 Generation de la cle maitre . . . . .                            | 10       |
| 1.6.2    | 5.2 Shamir Secret Sharing (2-of-3) . . . . .                         | 10       |
| 1.6.3    | 5.3 Protection des parts . . . . .                                   | 10       |
| 1.6.4    | 5.4 Protection du stockage de session . . . . .                      | 11       |
| 1.7      | 6. KEM hybride post-quantique . . . . .                              | 11       |
| 1.7.1    | 6.1 Pourquoi le post-quantique ? . . . . .                           | 11       |
| 1.7.2    | 6.2 Approche hybride . . . . .                                       | 11       |
| 1.7.3    | 6.3 Processus d'encapsulation . . . . .                              | 11       |
| 1.7.4    | 6.4 Processus de decapsulation . . . . .                             | 12       |
| 1.7.5    | 6.5 Tailles des cles . . . . .                                       | 12       |
| 1.7.6    | 6.6 Bibliotheque . . . . .                                           | 12       |
| 1.8      | 7. Pipeline de chiffrement des e-mails . . . . .                     | 13       |
| 1.8.1    | 7.1 E-mail entrant (SMTP → Stockage chiffre) . . . . .               | 13       |
| 1.8.2    | 7.2 Lecture d'un e-mail (dechiffrement dans le navigateur) . . . . . | 14       |
| 1.8.3    | 7.3 Envoi d'un e-mail . . . . .                                      | 14       |
| 1.8.4    | 7.4 Pieces jointes . . . . .                                         | 14       |
| 1.8.5    | 7.5 Regroupement des e-mails (Zero-Knowledge) . . . . .              | 15       |
| 1.9      | 8. Couche de transport API chiffee . . . . .                         | 15       |
| 1.9.1    | 8.1 Problematique . . . . .                                          | 15       |
| 1.9.2    | 8.2 Solution : API chiffee de bout en bout . . . . .                 | 15       |
| 1.9.3    | 8.3 Proprietes essentielles . . . . .                                | 16       |
| 1.9.4    | 8.4 Ce que voit CloudFlare . . . . .                                 | 16       |
| 1.10     | 9. Cryptographie du partage de dossiers . . . . .                    | 16       |
| 1.10.1   | 9.1 Modele de partage . . . . .                                      | 16       |
| 1.10.2   | 9.2 Flux de partage . . . . .                                        | 16       |
| 1.10.3   | 9.3 Modele de permissions . . . . .                                  | 17       |
| 1.10.4   | 9.4 Copie inter-coffres (Re-Wrapping) . . . . .                      | 17       |
| 1.11     | 10. Protections contre les canaux auxiliaires . . . . .              | 17       |
| 1.11.1   | 10.1 Bucket Padding . . . . .                                        | 17       |
| 1.11.2   | 10.2 Compression avant chiffrement . . . . .                         | 18       |
| 1.11.3   | 10.3 Confidentialite du regroupement . . . . .                       | 18       |
| 1.12     | 11. Gestion et cycle de vie des cles . . . . .                       | 18       |

|        |      |                                                               |    |
|--------|------|---------------------------------------------------------------|----|
| 1.12.1 | 11.1 | Hierarchie des clés                                           | 18 |
| 1.12.2 | 11.2 | Stockage des clés                                             | 19 |
| 1.12.3 | 11.3 | Empreintes de clés                                            | 19 |
| 1.13   | 12.  | Mécanisme de récupération                                     | 19 |
| 1.13.1 | 12.1 | Clé de récupération (mnémonique BIP39)                        | 19 |
| 1.13.2 | 12.2 | Dérivation de la clé de récupération                          | 19 |
| 1.13.3 | 12.3 | Ce que le serveur stocke                                      | 20 |
| 1.13.4 | 12.4 | Processus de récupération                                     | 20 |
| 1.13.5 | 12.5 | Limitation de débit                                           | 20 |
| 1.13.6 | 12.6 | Pas de récupération de mot de passe                           | 20 |
| 1.14   | 13.  | Authentification sans mot de passe (Passkeys)                 | 20 |
| 1.14.1 | 13.1 | Extension WebAuthn PRF                                        | 20 |
| 1.14.2 | 13.2 | Fonctionnement                                                | 21 |
| 1.14.3 | 13.3 | Passkeys multiples                                            | 21 |
| 1.15   | 14.  | Guardian : protection MITM & signature des réponses           | 21 |
| 1.15.1 | 14.1 | Le problème des applications web                              | 21 |
| 1.15.2 | 14.2 | Vue d'ensemble de l'architecture                              | 21 |
| 1.15.3 | 14.3 | Vérification de la signature des réponses (Ed25519)           | 22 |
| 1.15.4 | 14.4 | Gestion des clés publiques Ed25519                            | 22 |
| 1.15.5 | 14.5 | Vérification des certificats TLS (Firefox)                    | 23 |
| 1.15.6 | 14.6 | Récupération automatique du certificat (anti-usurpation)      | 23 |
| 1.15.7 | 14.7 | Indicateurs d'état de sécurité                                | 24 |
| 1.15.8 | 14.8 | Couverture des menaces                                        | 24 |
| 1.15.9 | 14.9 | Limites                                                       | 24 |
| 1.16   | 15.  | Archive e-mail d'entreprise (Blockchain)                      | 25 |
| 1.16.1 | 15.1 | Vue d'ensemble                                                | 25 |
| 1.16.2 | 15.2 | Architecture de la chaîne de hachage                          | 25 |
| 1.16.3 | 15.3 | Détection d'altération                                        | 26 |
| 1.16.4 | 15.4 | Clé d'archive d'entreprise (CAK) — Chiffrement Zero-Knowledge | 26 |
| 1.16.5 | 15.5 | Ce qui est chiffré                                            | 27 |
| 1.16.6 | 15.6 | Piste d'audit                                                 | 28 |
| 1.16.7 | 15.7 | Mise en suspens juridique & rétention                         | 28 |
| 1.16.8 | 15.8 | Export forensique                                             | 28 |
| 1.17   | 16.  | Ce que le serveur voit — et ce qu'il ne voit pas              | 29 |
| 1.17.1 | 16.1 | Le serveur PEUT voir                                          | 29 |
| 1.17.2 | 16.2 | Le serveur NE PEUT PAS voir                                   | 29 |
| 1.17.3 | 16.3 | Garantie cryptographique                                      | 30 |
| 1.18   | 17.  | Référence des algorithmes                                     | 30 |
| 1.18.1 | 17.1 | Tableau complet des algorithmes                               | 30 |
| 1.18.2 | 17.2 | Niveaux de sécurité                                           | 31 |
| 1.19   | 18.  | Comparaison avec d'autres fournisseurs                        | 31 |
| 1.20   | 19.  | Limites & transparence                                        | 32 |
| 1.20.1 | 19.1 | Modèle de confiance des applications web                      | 32 |
| 1.20.2 | 19.2 | Visibilité des métadonnées                                    | 33 |
| 1.20.3 | 19.3 | Journal de traitement des e-mails                             | 33 |
| 1.20.4 | 19.4 | Sécurité des e-mails externes                                 | 34 |
| 1.20.5 | 19.5 | Pas de sequestre de clés                                      | 34 |
| 1.21   | 20.  | Feuille de route                                              | 34 |
| 1.22   |      | Historique du document                                        | 34 |

1.23 Contact . . . . . 34

# 1. Livre blanc de securite Aionda Mail

---

Version 1.0 — Mars 2026 Aionda GmbH, Stuttgart, Allemagne

---

## 1.1 Table des matieres

1. Resume executif
  2. Modele de menace
  3. Vue d'ensemble de l'architecture
  4. Authentification Zero-Knowledge (OPAQUE)
  5. Cle maitre du coffre-fort & Shamir Secret Sharing
  6. KEM hybride post-quantique
  7. Pipeline de chiffrement des e-mails
  8. Couche de transport API chiffree
  9. Cryptographie du partage de dossiers
  10. Protections contre les canaux auxiliaires
  11. Gestion et cycle de vie des cles
  12. Mecanisme de recuperation
  13. Authentification sans mot de passe (Passkeys)
  14. Guardian : protection MITM & signature des reponses
  15. Archive e-mail d'entreprise (Blockchain)
  16. Ce que le serveur voit — et ce qu'il ne voit pas
  17. Reference des algorithmes
  18. Comparaison avec d'autres fournisseurs
  19. Limites & transparence
  20. Feuille de route
- 

## 1.2 1. Resume executif

Aionda Mail est un service de messagerie chiffre Zero-Knowledge et post-quantique, opere par Aionda GmbH a Stuttgart, en Allemagne. Le service combine des adresses e-mail jetables (DEAs) avec une boite aux lettres entierement chiffree — une combinaison qu'aucun autre fournisseur ne propose.

### Proprietes de securite fondamentales :

- **Architecture Zero-Knowledge** : Tout le chiffrement et le dechiffrement s'effectue exclusivement dans le navigateur de l'utilisateur. Le serveur n'a jamais acces au contenu des e-mails dans la boite aux lettres securisee, aux mots de passe ou aux cles de chiffrement.
- **Securite post-quantique** : Le mecanisme hybride d'encapsulation de cles (X25519 + ML-KEM-1024) protege toutes les donnees contre les attaques des ordinateurs classiques et quantiques.
- **Authentification Zero-Knowledge** : Le protocole OPAQUE (RFC 9807) garantit que les mots de passe ne sont jamais transmis ni stockes sur le serveur — meme pas sous forme de hachages.

- **Shamir Secret Sharing (2-of-3)** : La cle maitre du coffre-fort est divisee en trois parts protegees par le mot de passe, la passkey et la cle de recuperation. Deux parts quelconques suffisent a reconstruire la cle maitre.
- **Perfect Forward Secrecy** : Chaque requete API utilise une paire de cles cryptographiques unique et a usage unique. La compromission d'une requete n'affecte aucune autre.
- **Protection MITM (Guardian)** : L'extension de navigateur verifie independamment toutes les reponses du serveur via des signatures Ed25519 et detecte les attaques de type « homme du milieu » grace a la verification des certificats TLS — meme face aux proxys d'entreprise et aux CDN compromis.
- **Archive e-mail conforme GoBD** : Les comptes d'entreprise beneficent d'une chaine de hachage inviolable (blockchain SHA3-256) avec contenu chiffre de bout en bout (Hybrid KEM), une piste d'audit complete, une mise en suspens juridique et une retention configurable — conforme aux reglementations allemandes GoBD.
- **Pas de recuperation de mot de passe** : En cas de perte du mot de passe et de toutes les methodes de recuperation, les donnees sont irrecoverables. C'est un choix de conception — cela prouve que le serveur ne peut pas acceder aux donnees.

**Jurisdiction** : Droit allemand (RGPD/DSGVO), aucun partage de donnees avec des services de renseignement etrangers.

## 1.3 2. Modele de menace

### 1.3.1 2.1 Contre quoi Aionda Mail protege

| Menace                                                                                   | Protection                                                                                                                                                                                    |
|------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Compromission du serveur</b> (fuite de base de donnees, acces interne)                | Tout le contenu des e-mails est chiffre avec des cles que le serveur ne possede jamais                                                                                                        |
| <b>Ecoute reseau</b> (FAI, Wi-Fi, CDN)                                                   | Transport API chiffre de bout en bout via Hybrid KEM                                                                                                                                          |
| <b>Inspection CloudFlare</b>                                                             | Les requetes API sont chiffrees avant de quitter le navigateur ; CloudFlare ne voit que du texte chiffre. L'extension Guardian detecte les alterations de reponses via des signatures Ed25519 |
| <b>Proxys MITM d'entreprise</b> (ZScaler, Fortinet, etc.)                                | L'extension Guardian detecte les certificats de proxy via une liste de blocage des emetteurs (Firefox)                                                                                        |
| <b>Attaques par ordinateur quantique</b> (« recolter maintenant, dechiffrer plus tard ») | ML-KEM-1024 (NIST FIPS 203) offre une resistance post-quantique                                                                                                                               |
| <b>Vol de base de donnees de mots de passe</b>                                           | OPAQUE ne stocke que des enregistrements cryptographiques, pas de hachages de mots de passe                                                                                                   |
| <b>Attaques hors ligne par force brute sur les mots de passe</b>                         | OPAQUE empeche les attaques hors ligne ; la limitation de debit cote serveur empeche les attaques en ligne                                                                                    |
| <b>Analyse de la taille des e-mails</b>                                                  | Le Bucket Padding masque les tailles reelles des e-mails                                                                                                                                      |
| <b>Canaux auxiliaires de compression</b> (CRIME/BREACH)                                  | Le Bucket Padding est applique apres la compression                                                                                                                                           |

|                                   |                                                             |
|-----------------------------------|-------------------------------------------------------------|
| <b>Enumeration d'utilisateurs</b> | Reponses factices deterministes pour les comptes inexistant |
|-----------------------------------|-------------------------------------------------------------|

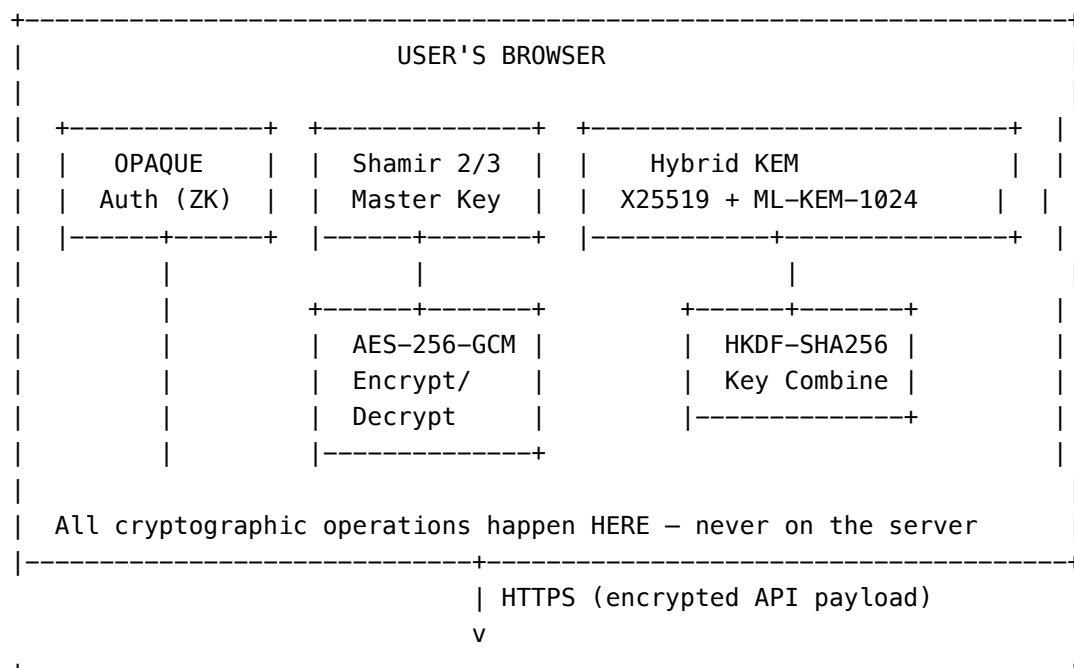
### 1.3.2 2.2 Contre quoi Aionda Mail ne protege PAS

| Limitation                                         | Explication                                                                                                                             |
|----------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <b>Appareil compromis</b>                          | Si un logiciel malveillant controle le navigateur, il peut lire le contenu déchiffre                                                    |
| <b>Metadonnees vers des destinataires externes</b> | Les e-mails vers Gmail/Outlook circulent en clair apres avoir quitte les serveurs d'Aionda (sauf si PGP est utilise)                    |
| <b>Metadonnees des e-mails sur les serveurs</b>    | Les horodatages, les adresses IP et les tailles des e-mails chiffrés sont visibles par le serveur                                       |
| <b>Confiance dans la livraison web</b>             | Le navigateur telecharge du JavaScript depuis les serveurs d'Aionda a chaque visite (voir la section 17 pour les mesures d'attenuation) |
| <b>Cryptanalyse sous contrainte physique</b>       | Aucun systeme cryptographique ne protege contre la coercion physique                                                                    |

### 1.3.3 2.3 Principe de conception

Aionda Mail suit le modele du « **serveur honnete** » : le systeme est concu de sorte que meme un serveur totalement compromis — ou un operateur malveillant — ne puisse dechiffrer les donnees des utilisateurs. La securite ne repose pas sur la confiance envers l'operateur. Elle repose sur les mathematiques.

### 1.4 3. Vue d'ensemble de l'architecture



| AIONDA SERVER                                                      |
|--------------------------------------------------------------------|
| Stores ONLY:                                                       |
| • Encrypted email blobs (ciphertext)                               |
| • Encrypted Shamir shares (XOR'd with derived keys)                |
| • Hybrid KEM ciphertexts (useless without private keys)            |
| • OPAQUE records (encrypted at rest with AES-256-GCM)              |
| • Public keys (X25519 + ML-KEM-1024)                               |
| CANNOT access: passwords, master keys, email content, private keys |

### 1.4.1 3.1 Vue d'ensemble des composants

| Composant                         | Objectif                                                   | Emplacement       |
|-----------------------------------|------------------------------------------------------------|-------------------|
| OPAQUE (RFC 9807)                 | Authentification par mot de passe<br>Zero-Knowledge        | Client + Serveur  |
| Shamir Secret Sharing             | Protection de la cle maitre du coffre-fort (seuil 2-of-3)  | Client uniquement |
| Hybrid KEM (X25519 + ML-KEM-1024) | Encapsulation de cles post-quantique pour les e-mails      | Client + Serveur  |
| AES-256-GCM                       | Chiffrement symetrique authentifie                         | Client + Serveur  |
| HKDF-SHA256                       | Derivation de cles a partir de secrets partages hybrides   | Client + Serveur  |
| BIP39                             | Encodage de la cle de recuperation (mnemonique de 24 mots) | Client uniquement |
| WebAuthn PRF                      | Deverrouillage du coffre-fort par passkey                  | Client uniquement |
| Bucket Padding                    | Protection contre les canaux auxiliaires                   | Client + Serveur  |

## 1.5 4. Authentification Zero-Knowledge (OPAQUE)

### 1.5.1 4.1 Pourquoi pas le hachage de mot de passe ?

Les services traditionnels stockent des hachages de mots de passe (bcrypt, Argon2). Bien que meilleure que le texte en clair, cette approche presente des faiblesses fondamentales :

- Le serveur voit le mot de passe lors de la connexion (meme brievement en RAM)
- Les hachages de mots de passe peuvent etre attaques par force brute hors ligne si la base de donnees est volee
- Le serveur pourrait etre modifie pour enregistrer les mots de passe

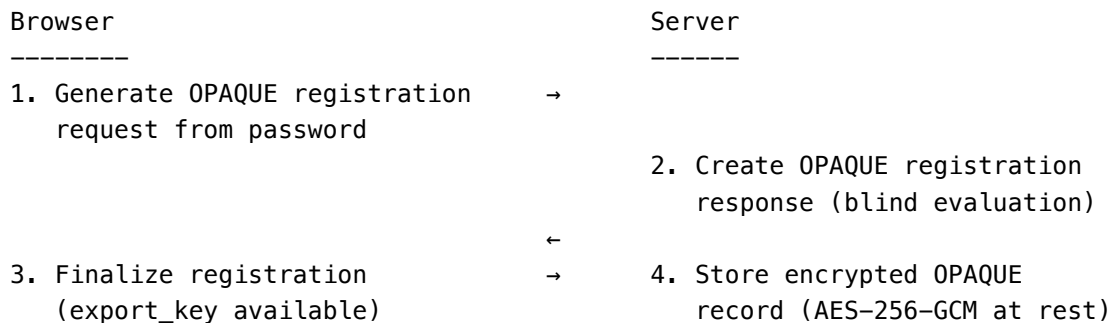


**OPAQUE elimine ces trois problemes.** Le mot de passe ne quitte jamais le navigateur — ni en clair, ni sous forme de hachage, sous aucune forme.

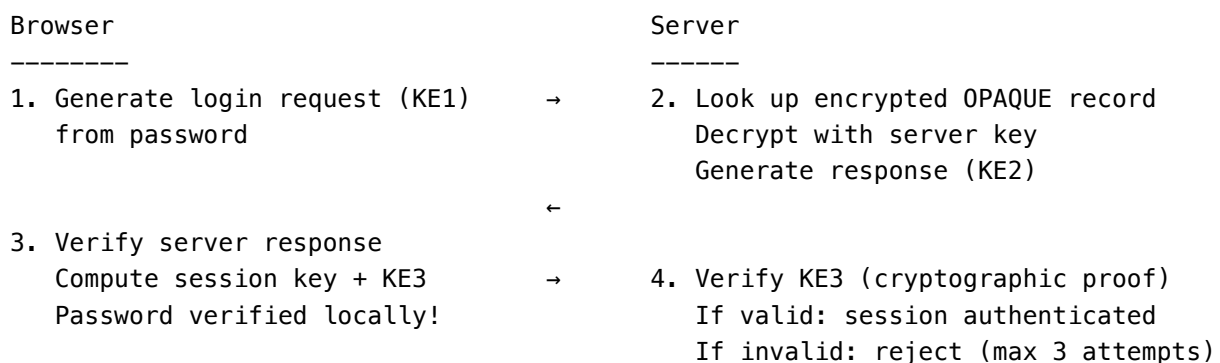
### 1.5.2 4.2 Fonctionnement d'OPAQUE

OPAQUE (RFC 9807) est un protocole d'échange de clés authentifié par mot de passe asymétrique (aPAKE). Il utilise un mécanisme de défi-réponse cryptographique où le serveur peut vérifier que l'utilisateur connaît le bon mot de passe sans jamais apprendre ce qu'il est.

#### Inscription (une seule fois) :



#### Connexion (a chaque session) :



#### Propriétés essentielles :

- Le mot de passe est vérifié **côté client** à l'étape 3 — le serveur ne le voit jamais
- Le serveur stocke un **enregistrement OPAQUE**, qui n'est pas un hachage de mot de passe et ne peut pas être attaqué par force brute hors ligne
- Les enregistrements OPAQUE sont en outre **chiffres au repos** avec AES-256-GCM à l'aide d'une clé côté serveur
- **Protection contre l'énumération d'utilisateurs** : les comptes inexistants reçoivent des réponses factices déterministes avec un timing identique
- **Limitation de débit** : maximum 3 tentatives d'authentification par session, délai d'expiration de session de 120 secondes

### 1.5.3 4.3 Implementation

- **Bibliothèque** : @serenity-kit/opaque (basée sur WASM, qualité production)
- **Composant serveur** : microservice dédiée aux opérations cryptographiques OPAQUE
- **Format base64** : base64url (compatible URL, sans remplissage) pour la compatibilité du protocole
- **Journalisation d'audit** : tous les événements d'authentification sont journalisés avec

horodatages et adresses IP

#### 1.5.4 4.4 Migration SRP

Les comptes existants utilisant SRP-6a sont automatiquement migres vers OPAQUE lors de la prochaine connexion. Apres la migration, le verificateur SRP est definitivement supprime. La migration est unidirectionnelle — les comptes ne peuvent pas revenir a SRP.

## 1.6 5. Cle maitre du coffre-fort & Shamir Secret Sharing

### 1.6.1 5.1 Generation de la cle maitre

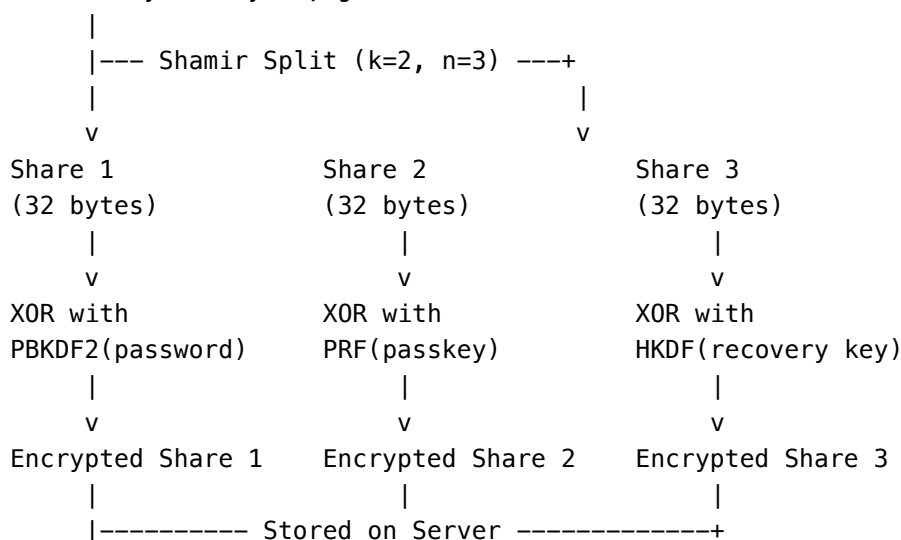
Lorsqu'un utilisateur active la boite aux lettres chiffree, une **cle maitre de 256 bits (32 octets)** est generee a l'aide du generateur de nombres aleatoires cryptographiquement sur du navigateur (`crypto.getRandomValues`).

Cette cle maitre est la racine de tout le chiffrement. Elle ne quitte jamais le navigateur en clair. Elle n'est stockee nulle part — ni dans le navigateur, ni sur le serveur, sous aucune forme.

### 1.6.2 5.2 Shamir Secret Sharing (2-of-3)

La cle maitre est divisee en trois parts a l'aide du schema **Shamir's Secret Sharing** sur le corps de Galois  $GF(2^8)$  avec le polynome irreductible AES ( $x^8 + x^4 + x^3 + x + 1$ ).

Master Key (32 bytes, generated once)



**Propriete de seuil :** deux quelconques des 3 parts suffisent a reconstruire la cle maitre par interpolation de Lagrange. Le serveur ne stocke que les parts chifrees — et ne peut en dechiffrer aucune.

### 1.6.3 5.3 Protection des parts

Chaque part est combinee par XOR avec une cle derivee d'un facteur d'authentification different :

| Part   | Protegee par        | Derivation de cle                                             |
|--------|---------------------|---------------------------------------------------------------|
| Part 1 | Mot de passe        | PBKDF2-SHA256, 600 000 iterations, sel aleatoire de 32 octets |
| Part 2 | Passkey (FIDO2)     | Extension WebAuthn PRF, liee au materiel                      |
| Part 3 | Cle de recuperation | HKDF-SHA3-256 avec sel lie au compte                          |

### Scenarios de reconstruction :

- **Connexion normale** : Mot de passe (Part 1) + Passkey (Part 2) → Cle maitre
- **Passkey perdue** : Mot de passe (Part 1) + Cle de recuperation (Part 3) → Cle maitre
- **Changement de mot de passe** : Passkey (Part 2) + Cle de recuperation (Part 3) → Cle maitre

### 1.6.4 5.4 Protection du stockage de session

Meme au sein d'une session de navigateur, la cle maitre n'est jamais stockee en clair :

1. Une cle AES-256 ephemere est generee
2. La cle maitre est chiffree avec cette cle ephemere
3. Seul le blob chiffre est place dans le sessionStorage
4. La cle ephemere n'existe qu'en memoire JavaScript (liberee par le ramasse-miettes a la fermeture de l'onglet)

## 1.7 6. KEM hybride post-quantique

### 1.7.1 6.1 Pourquoi le post-quantique ?

Les ordinateurs quantiques executant l'algorithme de Shor pourraient casser la cryptographie a cle publique classique (RSA, ECDH, X25519) en temps polynomial. Bien que des ordinateurs quantiques a grande echelle n'existent pas encore, la menace des attaques « **recolter maintenant, dechiffrer plus tard** » est reelle : des adversaires pourraient stocker des donnees chiffrees aujourd'hui et les dechiffrer lorsque les ordinateurs quantiques seront disponibles.

### 1.7.2 6.2 Approche hybride

Aionda Mail utilise un **mecanisme hybride d'encapsulation de cles** combinant :

- **X25519** (Curve25519 ECDH) — securite classique eprouvee, niveau de securite de 128 bits
- **ML-KEM-1024** (NIST FIPS 203, anciennement Kyber-1024) — securite post-quantique, niveau de securite NIST 5

L'approche hybride offre une **defense en profondeur** : la cle combinee est sure tant qu'**au moins un** des deux algorithmes reste incasse.

### 1.7.3 6.3 Processus d'encapsulation

Sender (encrypting an email):

1. Generate ephemeral X25519 keypair
2. X25519 key agreement with recipient's public key

- x25519SharedSecret (32 bytes)
- 3. ML-KEM-1024 encapsulation with recipient's public key
  - mlKemSharedSecret (32 bytes) + mlKemCiphertext (1568 bytes)
- 4. Combine secrets:
 

```
combinedSecret = x25519SharedSecret || mlKemSharedSecret (64 bytes)
```
- 5. Derive final key:
 

```
sharedSecret = HKDF-SHA256(
  ikm = combinedSecret,
  salt = nil,
  info = "trashmail-hybrid-kem-v1",
  length = 32
)
```
- 6. Use sharedSecret to wrap the email's ephemeral AES-256 key

#### 1.7.4 6.4 Processus de decapsulation

Recipient (decrypting an email):

- 1. X25519 key agreement:
 

```
x25519Shared = X25519(recipientPrivateKey, ephemeralPublicKey)
```
- 2. ML-KEM-1024 decapsulation:
 

```
mlKemShared = ML-KEM-1024.Decapsulate(mlKemCiphertext, recipientPrivateKey)
```
- 3. Combine and derive (identical to sender):
 

```
sharedSecret = HKDF-SHA256(x25519Shared || mlKemShared, "trashmail-hybrid-kem-v1")
```
- 4. Unwrap email's ephemeral AES-256 key using sharedSecret
- 5. Decrypt email content with ephemeral key

#### 1.7.5 6.5 Tailles des cles

| Parametre               | Taille       | Standard           |
|-------------------------|--------------|--------------------|
| X25519 public key       | 32 octets    | RFC 7748           |
| X25519 private key      | 32 octets    | RFC 7748           |
| ML-KEM-1024 public key  | 1 568 octets | NIST FIPS 203      |
| ML-KEM-1024 private key | 3 168 octets | NIST FIPS 203      |
| ML-KEM-1024 ciphertext  | 1 568 octets | NIST FIPS 203      |
| Secret partage combine  | 32 octets    | Sortie HKDF-SHA256 |

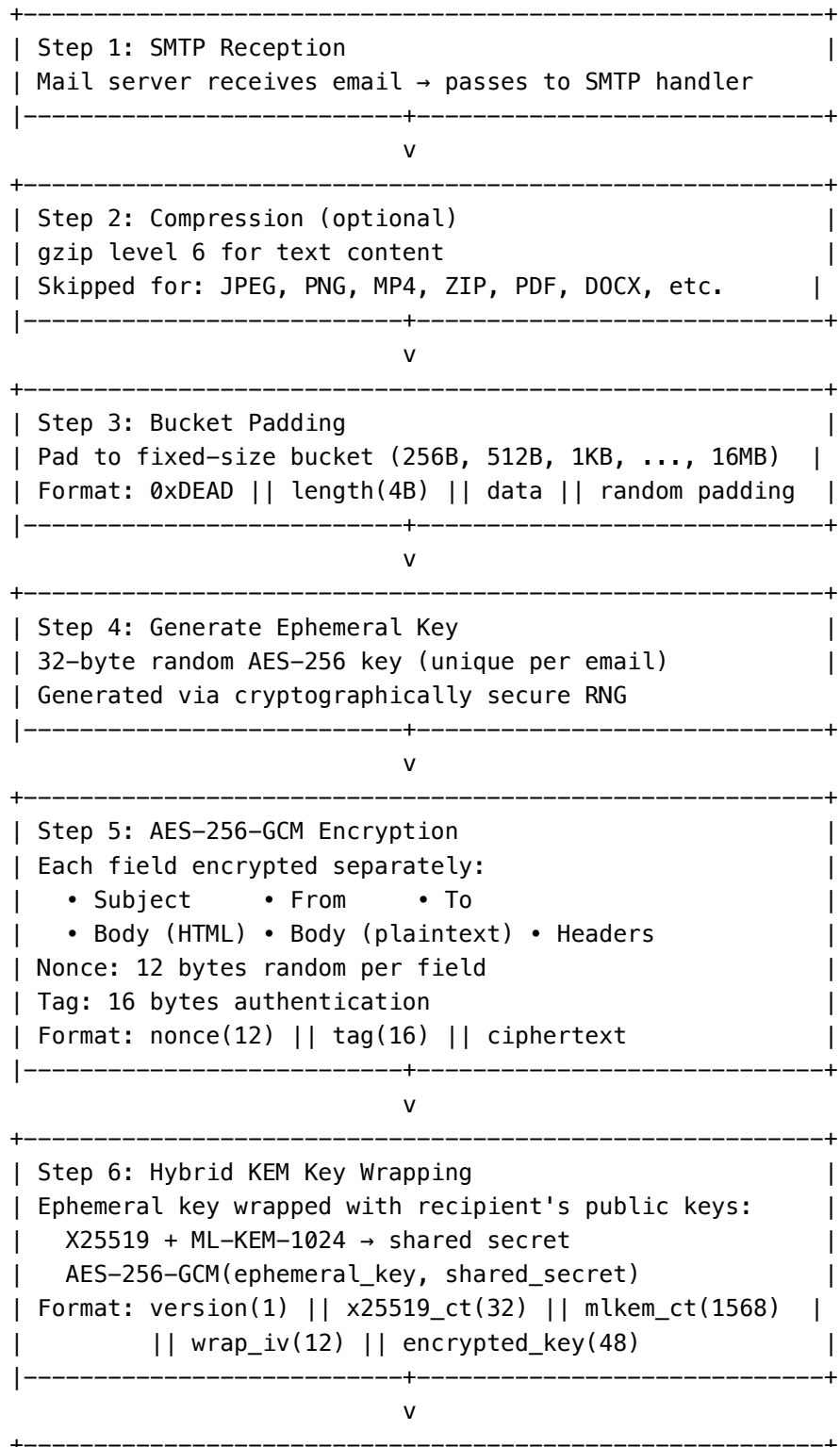
#### 1.7.6 6.6 Bibliotheque

- **ML-KEM-1024** : @noble/post-quantum (auditee, implementation en JavaScript pur)
- **X25519** : API WebCrypto (crypto.subtle.deriveBits)
- **HKDF** : @noble/hashes (conforme RFC 5869)

## 1.8 7. Pipeline de chiffrement des e-mails

### 1.8.1 7.1 E-mail entrant (SMTP → Stockage chiffré)

Lorsqu'un e-mail arrive sur le serveur SMTP d'Aionda Mail :



```
| Step 7: Secure Erasure |
| sodium_memzero() clears ephemeral key from RAM |
| Only encrypted blobs remain |
|-----+-----|
|                               v |
|-----+-----|
| Step 8: Database Storage |
| Stored in vault_emails table: |
|   encrypted_subject, encrypted_from, encrypted_to, |
|   encrypted_body, encrypted_body_text, |
|   encrypted_headers, wrapped_ephemeral_key |
| Threading: SHA-256 hashes of Message-ID/In-Reply-To |
|           (not plaintext – zero-knowledge threading) |
|-----+-----|
```

### 1.8.2 7.2 Lecture d'un e-mail (dechiffrement dans le navigateur)

Le processus inverse se deroule entierement dans le navigateur :

1. Recuperation de l'e-mail chiffré depuis le serveur via l'API chiffrée
2. Analyse de la cle ephémère encapsulée (extraction du texte chiffré X25519 + texte chiffré ML-KEM)
3. **Decapsulation Hybrid KEM** à l'aide des clés privées du coffre-fort → secret partagé
4. Dechiffrement de la cle ephémère AES-256
5. **Dechiffrement AES-256-GCM** de chaque champ (objet, expéditeur, destinataire, corps, en-têtes)
6. Reordonnancement des octets : format serveur (nonce || tag || ct) → format WebCrypto (nonce || ct || tag)
7. Suppression du Bucket Padding (détection des octets magiques 0xDEAD)
8. Décompression si gzip (détection des octets magiques 0x1F8B)
9. Decodage UTF-8 vers texte en clair

### 1.8.3 7.3 Envoi d'un e-mail

Lors de la composition et de l'envoi d'un e-mail :

1. Le client chiffre le contenu de l'e-mail avec un protocole de defi-reponse
2. Le serveur recoit le payload chiffre, le dechiffre de maniere ephemere (en memoire uniquement), et l'envoie via SMTP
3. Le serveur retourne les en-tetes MIME generes au client (chiffres)
4. Le client chiffre une copie avec la cle maitre du coffre-fort et la stocke dans le dossier Envoyes
5. Le texte en clair ephemere cote serveur est immediatement detruit — jamais ecrit sur disque

### 1.8.4 7.4 Pieces jointes

Chaque piece jointe est chiffree independamment :

- Cle ephemere AES-256 separee par piece jointe
- Encapsulation Hybrid KEM separee par piece jointe

- Nom de fichier et type MIME chiffres separement
- Pas de compression pour les formats deja compresses (JPEG, ZIP, PDF, etc.)

### 1.8.5 7.5 Regroupement des e-mails (Zero-Knowledge)

Le regroupement des e-mails (classement des e-mails lies) utilise uniquement des **hachages SHA-256** des en-tetes Message-ID et In-Reply-To. Le serveur ne voit jamais les chaines Message-ID reelles — il peut regrouper les e-mails par egalite de hachage sans en connaitre le contenu.

## 1.9 8. Couche de transport API chiffee

### 1.9.1 8.1 Problematique

Meme avec HTTPS, certains intermediaires peuvent inspecter le trafic :

- **CloudFlare** (CDN/protection DDoS) termine le TLS et peut voir les requetes en clair
- **Les proxys d'entreprise** peuvent effectuer une inspection TLS
- **Les parametres API** (comme ?cmd=read\_email&id=123) exposent des metadonnees

### 1.9.2 8.2 Solution : API chiffee de bout en bout

Toutes les communications API sont en outre chifrees de bout en bout entre le navigateur et le serveur applicatif, a l'interieur du tunnel HTTPS :

Browser  
-----

Server  
-----

Phase 1: Key Exchange (once per session)

```

GET /get_encryption_keys      →      Return 20 pre-generated
                                Hybrid KEM keypairs
                                {uuid, x25519_pub, mlkem_pub}
                                ←

```

Phase 2: Encrypted Request (every API call)

```

1. Pick random keypair from cache
2. Hybrid KEM encapsulate → shared secret
3. gzip compress request payload
4. Bucket-pad compressed data
5. AES-256-GCM encrypt with shared secret
6. Generate ephemeral response keypair
7. POST /e {                      →      8. Validate key ownership
    encrypted_payload,            9. Hybrid KEM decapsulate
    key_uuid,                     10. AES-256-GCM decrypt
    x25519_ciphertext,            11. Decompress
    mlkem_ciphertext,             12. Route to API controller
    response_x25519_pub,          13. Execute business logic
    response_mlkem_pub            14. Encrypt response with
}                                  client's response keys
                                ←      15. Return encrypted response

```

16. Hybrid KEM decapsulate response
17. AES-256-GCM decrypt
18. Decompress → plaintext response

### 1.9.3 8.3 Propriétés essentielles

- **Usage unique** : chaque paire de clés API est utilisée exactement une fois, puis définitivement invalidée
- **Perfect Forward Secrecy** : la compromission d'une clé de requête n'affecte aucune autre requête
- **Liées à la session** : les clés sont revendiquées par une session spécifique et ne peuvent pas être réutilisées par une autre
- **Pool de clés** : le serveur maintient environ 100 000 paires de clés pré-générées
- **Reapprovisionnement automatique** : le client demande automatiquement de nouvelles clés lorsque le cache tombe en dessous de 10
- **Durée de vie des clés** : les clés revendiquées expirent après 24 heures
- **Bidirectionnel** : la requête ET la réponse sont chiffrées — le serveur ne retourne jamais de texte en clair

### 1.9.4 8.4 Ce que voit CloudFlare

Avec cette architecture, CloudFlare (ou tout proxy terminant le TLS) ne voit que :

- POST /e — un point d'entrée unique et opaque
- Un blob binaire de données chiffrées
- Aucun nom de commande API, aucun paramètre, aucun identifiant d'e-mail, aucune donnée utilisateur

## 1.10 9. Cryptographie du partage de dossiers

### 1.10.1 9.1 Modèle de partage

Les utilisateurs peuvent partager des dossiers chiffrés avec d'autres utilisateurs d'Aionda Mail. Le mécanisme de partage utilise le Hybrid KEM pour chiffrer une clé spécifique au dossier pour chaque destinataire.

### 1.10.2 9.2 Flux de partage

Folder Owner

-----

Recipient

-----

1. Derive folder key from master key:  
`folderKey = HKDF-SHA256(masterKey, folderUuid)`
2. Fetch recipient's public keys:  
`recipient.x25519_pub (32 bytes)`  
`recipient.mlkem_pub (1568 bytes)`
3. Hybrid KEM encapsulate:  
`hybridEncapsulate(recipient.x25519_pub, recipient.mlkem_pub)`  
→ {x25519Ciphertext, mlKemCiphertext, sharedSecret}



## 4. Encrypt folder key:

```
wrappedKey = AES-256-GCM(folderKey, sharedSecret, nonce)
```

## 5. Store on server:

```
{x25519_ct, mlkem_ct, nonce, wrappedKey, permissions}
```

## 6. Fetch sharing record from server

## 7. Hybrid KEM decapsulate:

```
hybridDecapsulate(x25519_ct, mlkem_ct,
  own_x25519_priv, own_mlkem_priv)
→ sharedSecret
```

## 8. Decrypt folder key:

```
folderKey = AES-GCM-decrypt(
  wrappedKey, sharedSecret, nonce)
```

## 9. Decrypt emails in folder using folderKey

## 1.10.3 9.3 Modele de permissions

| Permission  | Capacite                                               |
|-------------|--------------------------------------------------------|
| readonly    | Lire les e-mails du dossier (dechiffrement uniquement) |
| einliefern  | Soumettre de nouveaux e-mails dans le dossier          |
| bearbeiten  | Modifier le contenu du dossier                         |
| antworten   | Repondre aux e-mails du dossier                        |
| vollzugriff | Acces complet, y compris le transfert de propriete     |

## 1.10.4 9.4 Copie inter-coffres (Re-Wrapping)

Lorsqu'un destinataire copie un e-mail d'un dossier partage vers son propre coffre-fort, la cle de l'e-mail doit etre **re-encapsulee** pour sa propre paire de cles Hybrid KEM. Cette operation est realisee entierement cote client :

1. Dechiffrement de la cle du dossier partage a l'aide des cles privees du destinataire
2. Dechiffrement de la cle ephemere de l'e-mail a l'aide de la cle du dossier
3. Re-encapsulation de la cle ephemere avec les propres cles publiques du destinataire
4. Stockage de la copie re-encapsulee dans le coffre-fort du destinataire

Le serveur facilite le transfert mais ne voit jamais aucune cle en clair.

## 1.11 10. Protections contre les canaux auxiliaires

## 1.11.1 10.1 Bucket Padding

**Problematiche :** les tailles des e-mails chiffres peuvent reveler des informations. Un attaquant observant les longueurs de texte chiffre pourrait en deduire le contenu (par exemple, un e-mail de 50 octets est probablement « OK, merci » tandis qu'un e-mail de 500 Ko contient des pieces

jointes).

**Solution :** toutes les données sont remplies jusqu'à des « buckets » de taille fixe avant le chiffrement :

Bucket sizes: 256B, 512B, 1KB, 2KB, 4KB, 8KB, 16KB, 32KB,  
64KB, 128KB, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB, 16MB

Format: [0xDEAD magic][4-byte length][actual data][random padding to bucket boundary]

**Exemple :** un e-mail de 523 octets est rempli jusqu'à 1 024 octets. Un observateur ne voit qu'un « e-mail de 1 Ko » — pas la taille réelle de 523 octets.

### 1.11.2 10.2 Compression avant chiffrement

Les données sont compressées avec gzip (niveau 6) **avant** le chiffrement. C'est le seul ordre correct :

- La compression après le chiffrement échouerait (les données chiffrées ont une entropie maximale)
- Le Bucket Padding après la compression empêche les attaques de type CRIME/BREACH qui exploitent les taux de compression

### 1.11.3 10.3 Confidentialité du regroupement

Les fils d'e-mails utilisent des hachages SHA-256 des en-têtes Message-ID au lieu d'identifiants en clair. Le serveur peut regrouper les e-mails liés par égalité de hachage sans connaître les identifiants de messages réels.

## 1.12 11. Gestion et cycle de vie des clés

### 1.12.1 11.1 Hierarchie des clés

Vault Master Key (32 bytes, generated once per account)

```
|
|--- Vault Keypair (Hybrid KEM)
|   |-- X25519 public key (32 bytes) – stored plaintext on server
|   |-- X25519 private key (32 bytes) – encrypted with master key
|   |-- ML-KEM-1024 public key (1568 bytes) – stored plaintext on server
|   |-- ML-KEM-1024 private key (3168 bytes) – encrypted with master key
|
|--- Per-Email Ephemeral Keys (32 bytes each)
|   |-- Wrapped with recipient's Hybrid KEM public keys
|
|--- Per-Attachment Ephemeral Keys (32 bytes each)
|   |-- Wrapped independently per attachment
|
|--- Folder Keys (derived via HKDF per folder)
|   |-- Shared via Hybrid KEM encapsulation per recipient
|
|--- Signature Encryption Key (derived from master key)
|   |-- Encrypts email signature templates
```

### 1.12.2 11.2 Stockage des clés

| Cle                           | Emplacement de stockage                                          | Protection                                                      |
|-------------------------------|------------------------------------------------------------------|-----------------------------------------------------------------|
| Cle maitre                    | Nulle part (reconstruite a la demande a partir des parts Shamir) | Shamir 2-of-3                                                   |
| Cles privees du coffre-fort   | Serveur (chiffrees)                                              | AES-256-GCM avec cle maitre                                     |
| Cles publiques du coffre-fort | Serveur (en clair)                                               | Non sensibles — publiques par definition                        |
| Cles ephemerer d'e-mail       | Serveur (encapsulees)                                            | Encapsulation Hybrid KEM                                        |
| Enregistremes OPAQUE          | Serveur (chiffres au repos)                                      | AES-256-GCM avec cle serveur                                    |
| Parts Shamir chiffrees        | Serveur                                                          | XOR avec les cles derivees du mot de passe/passkey/recuperation |
| Cles de transport API         | Serveur (pool pre-genere)                                        | Usage unique, duree de vie 24h                                  |

### 1.12.3 11.3 Empreintes de clés

Chaque paire de clés du coffre-fort possède une empreinte SHA-256 stockée sur le serveur. Cela permet :

- Une piste d'audit des rotations de clés
- La detection de changements de clés non autorises
- La verification de l'integrite des clés cote client

## 1.13 12. Mecanisme de recuperation

### 1.13.1 12.1 Cle de recuperation (mnemonique BIP39)

Lors de la configuration du coffre-fort, l'utilisateur recoit une **phrase de recuperation de 24 mots** generee a partir de 256 bits d'entropie, encodee selon le standard BIP39 :

Example: apple river mountain sunset golden bridge falcon ocean  
 crystal thunder meadow silver dolphin forest marble castle  
 velvet compass harbor window ancient pepper rocket shield

### 1.13.2 12.2 Derivation de la cle de recuperation

1. Generate: 256 bits random entropy
2. Encode: BIP39 mnemonic (24 words, 11 bits per word)
3. Derive: verificationKey = HKDF-SHA3-256(entropy,

```

        salt = accountId,
        info = "trashmail-recovery-verify"
    )
4. Hash:    verificationHash = SHA3-256(verificationKey)
5. Store:    Server stores ONLY verificationHash (32 bytes)

```

### 1.13.3 12.3 Ce que le serveur stocke

Le serveur ne stocke **que le hachage SHA3-256** d'une cle de verification derivee. Il ne stocke pas :

- Les mots de recuperation
- L'entropie
- La cle de verification elle-meme

### 1.13.4 12.4 Processus de recuperation

1. L'utilisateur saisit la phrase de recuperation de 24 mots
2. Le client derive l'entropie  $\square$  HKDF-SHA3-256  $\square$  SHA3-256  $\square$  hachage de verification
3. Le client envoie le hachage de verification au serveur (jamais la cle en clair)
4. Le serveur compare avec le hachage stocke
5. En cas de correspondance : toutes les methodes 2FA sont desactivees, l'utilisateur configure une nouvelle authentication
6. La cle de recuperation est revoquee apres un seul usage

### 1.13.5 12.5 Limitation de debit

- Maximum 3 tentatives de verification par heure
- Verrouillage de 60 minutes apres depassement de la limite
- Usage unique : la cle de recuperation est definitivement revoquee apres une utilisation reussie

### 1.13.6 12.6 Pas de recuperation de mot de passe

**Il n'y a pas de reinitialisation du mot de passe par e-mail.** Si un utilisateur perd son mot de passe ET tous les autres facteurs d'authentification (passkey + cle de recuperation), ses donnees sont definitivement inaccessibles. C'est une decision de conception intentionnelle qui prouve l'integrite du modele Zero-Knowledge — si le serveur pouvait recuperer les donnees, il pourrait aussi les lire.

---

## 1.14 13. Authentication sans mot de passe (Passkeys)

### 1.14.1 13.1 Extension WebAuthn PRF

Aionda Mail prend en charge les passkeys FIDO2 (cles de securite materielles, authenticateurs biometriques) pour la connexion sans mot de passe et le deverrouillage du coffre-fort.

L'**extension WebAuthn PRF (Pseudo-Random Function)** fournit une sortie deterministe de 32 octets liee a la passkey et au credential specifiques. Cette sortie est utilisee pour proteger la Part Shamir 2.

### 1.14.2 13.2 Fonctionnement

#### 1. Registration:

- User creates passkey via `navigator.credentials.create()`
- PRF extension generates hardware-bound output
- Output XOR'd with Shamir Share 2 → encrypted share stored on server

#### 2. Authentication:

- User authenticates with passkey (biometric/PIN)
- PRF extension reproduces same 32-byte output
- Output XOR'd with encrypted share → Shamir Share 2 recovered
- Combined with Share 1 (password) → Master Key reconstructed

#### 3. Vault Unlock (passwordless):

- If both passkey (Share 2) and password (Share 1) available → immediate unlock
- Password verified via OPAQUE (separate from passkey auth)

### 1.14.3 13.3 Passkeys multiples

Les utilisateurs peuvent enregistrer plusieurs passkeys (par exemple, Touch ID MacBook, Face ID iPhone, YubiKey). Chaque passkey protege independamment sa propre copie de la Part 2. Une seule passkey combinee au mot de passe suffit a deverrouiller le coffre-fort.

## 1.15 14. Guardian : protection MITM & signature des reponses

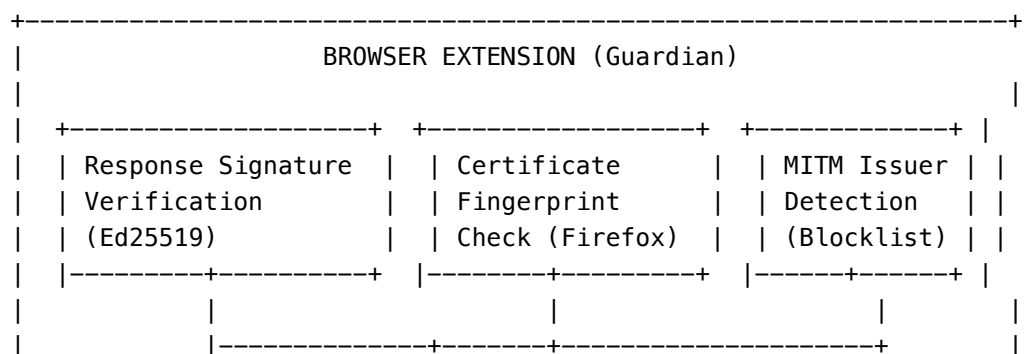
### 1.15.1 14.1 Le probleme des applications web

Toute application web a un probleme de confiance inherent : le navigateur telecharge du JavaScript depuis le serveur a chaque visite. Un attaquant de type « homme du milieu » (MITM) — qu'il s'agisse d'un CDN compromis, d'un proxy d'entreprise ou d'un FAI malveillant — pourrait theoriquement injecter du code modifie qui exfiltre les cles de chiffrement.

Aionda Mail repond a ce probleme avec le **module Guardian**, un composant d'extension de navigateur (disponible pour Chrome et Firefox) qui verifie independamment l'integrite du serveur.

### 1.15.2 14.2 Vue d'ensemble de l'architecture

Le module Guardian fonctionne au sein du Service Worker de l'extension de navigateur — completement independant du JavaScript de l'application web. Il effectue trois types de verification :



|  |                            |  |
|--|----------------------------|--|
|  | v                          |  |
|  | Security Status Badge      |  |
|  | Yes Green = Verified       |  |
|  | No Red = MITM Detected     |  |
|  | ! Red = Missing Signatures |  |
|  | -----+                     |  |

### 1.15.3 14.3 Verification de la signature des reponses (Ed25519)

Chaque reponse API du serveur d'Aionda Mail est signee cryptographiquement a l'aide d'**Ed25519** (Edwards-curve Digital Signature Algorithm).

#### Processus de signature (cote serveur) :

1. Server generates API response body (JSON)
2. Construct signing input: responseBody + "|" + unixTimestamp
3. Sign with Ed25519 private key → 64-byte signature
4. Attach HTTP headers:
  - X-Aionda-Signature: <base64(signature)>
  - X-Aionda-Timestamp: <unix\_timestamp>
  - X-Aionda-Key-Id: <key\_identifieur>

#### Processus de verification (extension de navigateur) :

1. Extract signature, timestamp, and key ID from HTTP headers
2. Look up Ed25519 public key by key ID (bundled in extension)
3. Verify key has not expired (valid\_from / valid\_until)
4. Check timestamp freshness:  $|\text{now} - \text{timestamp}| \leq 300 \text{ seconds}$
5. Reconstruct signing input: responseBody + "|" + timestamp
6. `crypto.subtle.verify("Ed25519", publicKey, signature, data)`
7. If invalid → MITM alert, red badge

#### Proprietes essentielles :

- **Protection contre la relecture** : la fenetre de 5 minutes sur l'horodatage empeche la relecture d'anciennes reponses
- **Detection d'alteration** : toute modification du corps de la reponse invalide la signature
- **Isolation des cles** : les cles publiques sont integrees dans l'extension (non telechargees depuis le serveur)
- **Separation des environnements** : les cles de developpement (dev-2026-01) ne peuvent pas etre utilisees sur les URL de production et vice versa

### 1.15.4 14.4 Gestion des cles publiques Ed25519

Les cles publiques sont distribuees avec l'extension de navigateur dans `public_key.json` :

```
{
  "keys": {
    "prod-2026-01": {
      "algorithm": "Ed25519",
      "public_key": "<base64 SPKI DER>",
      "valid_from": "2026-01-13T00:00:00Z",
      "valid_until": "2027-01-13T00:00:00Z"
    }
  }
}
```

}

- **Format de cle** : SPKI DER (Subject Public Key Info, Distinguished Encoding Rules)
- **Taille de cle** : 32 octets (cle publique Ed25519 de 256 bits)
- **Taille de signature** : 64 octets (fixe)
- **Rotation** : de nouvelles cle sont ajoutes avant l'expiration des anciennes ; les mises a jour de l'extension livrent les nouvelles cle
- **Aucune confiance envers le serveur** : les cle sont integrees dans le binaire de l'extension, non recuperees depuis le serveur

### 1.15.5 14.5 Verification des certificats TLS (Firefox)

Sur Firefox, le module Guardian effectue une verification supplementaire des certificats TLS a l'aide de l'API `browser.webRequest.getSecurityInfo()` (non disponible sur Chrome en raison des limitations de Manifest V3).

#### Flux de verification :

1. Browser extension intercepts HTTPS response
2. Extract TLS certificate chain from browser's security info:
  - Leaf certificate fingerprint (SHA-256)
  - Issuer Distinguished Name (O=, CN=)
  - Subject (CN=)
3. Check against known MITM issuers (hardcoded blocklist):  
ZScaler, Netskope, Fortinet, Palo Alto, Blue Coat, Check Point, Barracuda, Sophos, WatchGuard, Cisco Umbrella  
→ If match: MITM detected, show warning
4. Check against trusted issuers:  
Google Trust Services, Cloudflare, Let's Encrypt, DigiCert, Sectigo  
→ If match AND subject matches expected domain: OK
5. If unknown issuer: Fetch server's own certificate fingerprint
  - Server connects to itself via external routing (prevents spoofing)
  - Response is Ed25519 signed (prevents MITM from lying about cert)
  - Compare issuer organization with browser's certificate issuer
  - If mismatch: MITM suspected, show warning

**Pourquoi la validation basee sur l'emetteur plutot que le pinning ?** CloudFlare (utilise comme CDN) effectue une rotation des certificats feuille entre les serveurs de bordure. Le pinning de certificat traditionnel (correspondance exacte des empreintes) provoquerait des faux positifs. La validation basee sur l'emetteur est plus robuste : l'autorite de certification emettrice est stable meme lorsque les certificats feuille changent.

### 1.15.6 14.6 Recuperation automatique du certificat (anti-usurpation)

Le point d'entree de certificat du serveur utilise une technique anti-usurpation ingenieuse :

Server connects to `cert.trashmail.com` (or `cert-subdomain.domain`)  
with SNI = `mail.aionda.com`

- Forces external routing through CloudFlare
- Receives the actual certificate that users see
- Prevents localhost spoofing
- Response signed with Ed25519 to prevent tampering

Le serveur demande essentiellement « quel certificat le monde extérieur voit-il pour mon domaine ? » — et signe la réponse pour que l'extension puisse lui faire confiance.

### 1.15.7 14.7 Indicateurs d'état de sécurité

L'extension affiche un badge dans la barre d'outils du navigateur :

| Badge  | Couleur | Signification                                                     |
|--------|---------|-------------------------------------------------------------------|
| Yes    | Vert    | Toutes les réponses vérifiées — signatures valides                |
| !      | Orange  | Utilisation d'une clé de signature obsolète (rotation en attente) |
| No     | Rouge   | MITM détecté — échec de la vérification de signature              |
| !      | Rouge   | Signatures manquantes — réponses non signées                      |
| Shield | Bleu    | Mode protégé — aucune vérification effectuée pour l'instant       |

### 1.15.8 14.8 Couverture des menaces

| Attaque                                     | Méthode de détection                                     | Navigateur       |
|---------------------------------------------|----------------------------------------------------------|------------------|
| Proxy MITM d'entreprise (ZScaler, Fortinet) | Liste de blocage des émetteurs de certificats            | Firefox          |
| Réponses API modifiées                      | Vérification de signature Ed25519                        | Chrome + Firefox |
| Attaques par relecture                      | Fenêtre d'horodatage de 5 minutes                        | Chrome + Firefox |
| Compromission du CDN (CloudFlare)           | Discordance de la signature de réponse                   | Chrome + Firefox |
| Substitution de certificat                  | Comparaison de l'émetteur + auto-vérification du serveur | Firefox          |
| Confusion des clés dev/prod                 | Identifiants de clés liés à l'environnement              | Chrome + Firefox |

### 1.15.9 14.9 Limites

- **Chrome Manifest V3** : impossible d'inspecter les certificats TLS — seule la vérification de signature de réponse est disponible
- **Extension requise** : les utilisateurs sans l'extension ne bénéficient pas des protections Guardian



- **Ed25519 n'est pas post-quantique** : la verification de signature utilise la cryptographie classique. Un ordinateur quantique suffisamment puissant pourrait theoriquement forger des signatures Ed25519.

## 1.16 15. Archive e-mail d'entreprise (Blockchain)

### 1.16.1 15.1 Vue d'ensemble

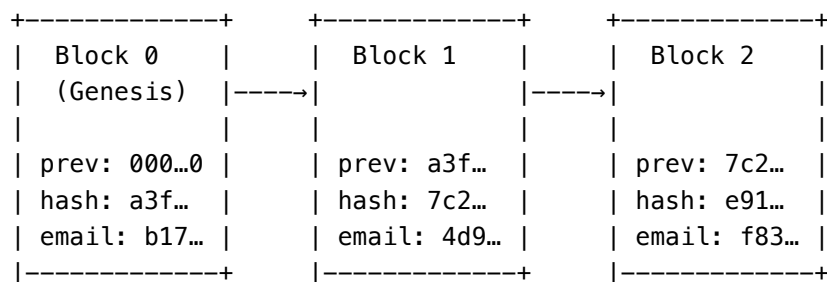
Le plan Entreprise d'Aionda Mail inclut une **archive e-mail conforme GoBD** securisee par une chaine de hachage cryptographique (blockchain). Chaque e-mail archive devient un bloc immuable dans une chaine propre a l'entreprise. Toute alteration — modification, suppression ou insertion de blocs — est detectee cryptographiquement.

L'archive combine deux couches de securite independantes :

1. **Chaine de hachage (SHA3-256)** : garantit l'integrite et l'immutabilite — prouve qu'aucun e-mail n'a ete modifie ou supprime apres l'archivage
2. **Chiffrement Hybrid KEM (CAK)** : garantit la confidentialite — le serveur ne peut pas lire le contenu des e-mails archives

### 1.16.2 15.2 Architecture de la chaine de hachage

Chaque e-mail archive devient un bloc dans une chaine sequentielle et inviolable :



#### Calcul du hachage de bloc :

```
block_hash = SHA3-256(
  prev_block_hash || "|" ||
  timestamp      || "|" ||
  email_hash     || "|" ||
  direction      || "|" ||
  sender_domain  || "|" ||
  recipient_domain
)
```

#### Proprietes :

- **Algorithme de hachage** : SHA3-256 (NIST FIPS 202)
- **Bloc de genese** : prev\_block\_hash = 64 zeros, block\_number = 0
- **Numerotation sequentielle** : imposee par la contrainte UNIQUE KEY (company\_uuid, block\_number) en base de donnees
- **Une chaine par entreprise** : isolation complete entre les entreprises
- **Hachage de l'e-mail** : SHA3-256(sender || recipient || timestamp || size) — preuve

d'integrite des donnees originales de l'e-mail

### 1.16.3 15.3 Detection d'alteration

L'algorithme de verification de la chaine detecte toute forme d'alteration :

For each block (ordered by block\_number ASC):

1. Verify link:      `block.prev_block_hash == expected_prev_hash`
2. Recalculate:    `expected = SHA3-256(prev_hash | timestamp | email_hash | ...)`
3. Verify content: `block.block_hash == expected`
4. Advance:        `expected_prev_hash = block.block_hash`

If ANY check fails → chain is broken at block N

| Tentative d'alteration                                     | Detection                                                                  |
|------------------------------------------------------------|----------------------------------------------------------------------------|
| Modifier le contenu de l'e-mail                            | email_hash change →<br>echec du recalcul de<br>block_hash                  |
| Modifier les metadonnees (expediteur, domaine, horodatage) | Inclues dans l'entree du<br>hachage → discordance de<br>block_hash         |
| Supprimer un bloc                                          | Le prev_block_hash du bloc<br>suivant devient orphelin                     |
| Inserer un bloc                                            | Rompt le block_number<br>sequentiel + la chaine<br>prev_block_hash         |
| Reordonner les blocs                                       | La contrainte UNIQUE KEY +<br>la verification sequentielle<br>l'empechent  |
| Remplacer toute la chaine                                  | Le hachage du bloc de<br>genese differerait de toute<br>sauvegarde externe |

**Le resultat de verification** indique le numero exact du bloc ou l'alteration a ete detectee, avec les valeurs de hachage attendues et reelles pour l'analyse forensique.

### 1.16.4 15.4 Cle d'archive d'entreprise (CAK) — Chiffrement Zero-Knowledge

Le contenu de l'archive est chiffre de bout en bout a l'aide d'une **cle d'archive d'entreprise** — une paire de cles Hybrid KEM (X25519 + ML-KEM-1024) generee cote client par le proprietaire de l'entreprise.

Company Owner's Browser

Server

1. Generate Hybrid KEM keypair (client-side):  
X25519 keypair (32 + 32 bytes)  
ML-KEM-1024 keypair (1568 + 3168 bytes)

2. Derive wrapping key from password:

```

wrappingKey = HKDF-SHA256(
  password,
  salt = "trashmail-archive-{account_id}",
  info = "trashmail-archive-key-wrap",
  length = 32
)

```

### 3. Wrap private keys:

```
AES-256-GCM(x25519_priv || mlkem_priv, wrappingKey)
```

### 4. Send to server:

- Public keys (plaintext)
- Wrapped private keys (encrypted)

### → Store:

```

archive_x25519_pub
archive_mlkem_pub
wrapped_archive_key

```

## Distribution de la cle aux autres employes (Administrateur, Responsable conformite) :

1. Le proprietaire dechiffre les cles privees CAK avec son mot de passe
2. Le proprietaire re-encapsule les cles privees avec la cle derivee du mot de passe de l'employe cible
3. Le serveur stocke la copie re-encapsulee sur l'enregistrement de l'employe
4. Chaque employe autorise dispose de sa propre copie encapsulee independamment

Le serveur ne voit jamais les cles privees CAK en clair.

### 1.16.5 15.5 Ce qui est chiffre

Lorsqu'un e-mail est archive, deux couches de chiffrement sont appliquees :

#### Metadonnees chiffrees (AES-256-GCM avec Hybrid KEM) :

```

{
  "d": "INBOUND",
  "s": "user@example.com",
  "r": "admin@company.de",
  "sd": "example.com",
  "rd": "company.de",
  "sz": 45000,
  "ts": "2026-02-27T10:30:00Z",
  "ha": true,
  "ac": 3,
  "en": "John Doe"
}

```

#### Contenu de l'e-mail chiffre (encapsulation Hybrid KEM separee) :

```

{
  "subject": "Meeting notes",
  "body": "<html>...</html>",
  "from": "sender@domain.com",
  "to": "recipient@company.de"
}

```

**Application Zero-Knowledge** : apres le chiffrement, les champs de metadonnees en clair dans la base de donnees (sender\_address, recipient\_address, domaines) sont remplaces par

leurs hachages SHA3-256. Le serveur ne stocke que des hachages — les valeurs originales n'existent que dans les blobs chiffrés.

#### 1.16.6 15.6 Piste d'audit

Chaque action sur l'archive est journalisée dans une **chaîne d'audit indépendante** (également enchaînée par hachages SHA3-256) :

| Action                                                         | Quand elle est journalisée              |
|----------------------------------------------------------------|-----------------------------------------|
| EMAIL_RECEIVED /<br>EMAIL_SENT /<br>DRAFT_ARCHIVED             | E-mail archive                          |
| VIEW_EMAIL /<br>VIEW_ATTACHMENT                                | Un employé lit un e-mail archive        |
| SEARCH_ARCHIVE                                                 | Recherche effectuée                     |
| EXPORT_EMAIL /<br>EXPORT_REPORT                                | Données exportées                       |
| VERIFY_CHAIN /<br>CHAIN_VERIFIED_OK /<br>CHAIN_VERIFIED_BROKEN | Vérification d'intégrité                |
| LEGAL_HOLD_SET /<br>LEGAL_HOLD_RELEASED                        | Mise en suspens juridique basculée      |
| ARCHIVE_DECRYPT                                                | CAK utilisée pour déchiffrer le contenu |
| ADMIN_ACCESS                                                   | Action administrative                   |

Chaque entrée d'audit enregistre : l'acteur (UUID + rôle), l'adresse IP, l'identifiant de session, le hachage de l'e-mail cible et si la chaîne était valide au moment de l'accès.

#### 1.16.7 15.7 Mise en suspens juridique & rétention

- **Durée de rétention** : configurable par entreprise (par défaut : 10 ans), calculée par e-mail comme `archived_at + retention_years`
- **Mise en suspens juridique** : des e-mails individuels peuvent être placés sous mise en suspens juridique, empêchant leur suppression jusqu'à la levée. Inclut le motif, l'acteur et l'horodatage.
- **Conformité GoBD** : la combinaison d'une chaîne de hachage immuable, d'une piste d'audit complète, d'une rétention configurable et d'une mise en suspens juridique satisfait les exigences du GoBD allemand (Grundsätze zur ordnungsmässigen Führung und Aufbewahrung von Büchern, Aufzeichnungen und Unterlagen in elektronischer Form sowie zum Datenzugriff).

#### 1.16.8 15.8 Export forensique

Les utilisateurs autorisés (Propriétaire, Administrateur) peuvent exporter l'état complet de la chaîne pour une vérification indépendante :

- Données complètes de la chaîne avec tous les hachages de blocs
- Résultat de vérification (valide/rompu, numéro de bloc rompu le cas échéant)
- Valeurs de hachage attendues et réelles pour l'analyse forensique
- 50 dernières entrées du journal d'audit

- Format JSON pour une re-verification externe avec toute implementation SHA3-256

## 1.17 16. Ce que le serveur voit — et ce qu'il ne voit pas

Cette section documente explicitement la frontiere Zero-Knowledge.

### 1.17.1 16.1 Le serveur PEUT voir

| Donnees                              | Raison de la visibilite                     | Mesure d'attenuation                                    |
|--------------------------------------|---------------------------------------------|---------------------------------------------------------|
| Adresse IP                           | Exigence TCP/IP                             | Utilisation d'un VPN/Tor possible si souhaite           |
| Horodatages                          | Heure de reception de l'e-mail              | Inherent au protocole e-mail                            |
| Blobs d'e-mails chiffrés             | Stockes pour la recuperation                | Chiffres avec AES-256-GCM, cle inconnue du serveur      |
| Tailles de texte chiffre rembourrees | Exigence de stockage                        | Le Bucket Padding masque les tailles reelles            |
| Adresse DEA du destinataire          | Exigence de routage                         | La DEA est jetable, ce n'est pas l'adresse reelle       |
| Existence du compte                  | Flux d'authentification                     | Protection contre l'enumeration d'utilisateurs deployee |
| Cles publiques                       | Requises pour le chiffrement par le serveur | Publiques par definition, non sensibles                 |
| Parts Shamir chiffrées               | Stockage pour l'utilisateur                 | XOR avec des cles que le serveur ne connait pas         |
| Enregistrements OPAQUE               | Protocole d'authentification                | Pas des hachages de mots de passe, chiffres au repos    |

### 1.17.2 16.2 Le serveur NE PEUT PAS voir

| Donnees                                      | Raison de l'invisibilite                                             |
|----------------------------------------------|----------------------------------------------------------------------|
| Contenu de l'e-mail (objet, corps, en-tetes) | Chiffre avec des cles ephemerées encapsulees via Hybrid KEM          |
| Mot de passe de l'utilisateur                | OPAQUE — le mot de passe n'est jamais transmis                       |
| Cle maitre                                   | Reconstruite uniquement dans le navigateur a partir des parts Shamir |
| Cles privees du coffre-fort                  | Chiffrees avec la cle maitre avant le stockage                       |
| Cles ephemerées d'e-mail                     | Encapsulees avec Hybrid KEM, le serveur n'a pas les cles privees     |
| Cle de recuperation / mnemonique             | Seul le hachage SHA3-256 de la cle derivee est stocke                |

| Donnees                   | Raison de l'invisibilite                                 |
|---------------------------|----------------------------------------------------------|
| Sorties PRF de la passkey | Liees au materiel, ne quittent jamais l'authentificateur |
| Noms de dossiers          | Chiffres avec des cles specifiques aux dossiers          |
| Signatures d'e-mail       | Chiffrees avec la cle maitre                             |
| Contenu des requetes API  | Chiffre via la couche de transport /e                    |
| Contenu des reponses API  | Chiffre avant la transmission                            |

### 1.17.3 16.3 Garantie cryptographique

Meme avec un acces complet a :

- La base de donnees complete
- Tout le trafic reseau
- Le code source et la configuration du serveur
- Tous les enregistrements OPAQUE et les cles du serveur

...un attaquant **ne peut pas** dechiffrer un seul e-mail sans le mot de passe de l'utilisateur (ou la passkey + la cle de recuperation). Ce n'est pas une politique — c'est une impossibilite mathematique imposee par la conception cryptographique.

## 1.18 17. Reference des algorithmes

### 1.18.1 17.1 Tableau complet des algorithmes

| Composant                            | Algorithme    | Parametres                                         | Standard        |
|--------------------------------------|---------------|----------------------------------------------------|-----------------|
| Authentification par mot de passe    | OPAQUE        | RFC 9807, aPAKE                                    | RFC 9807        |
| Derivation de cle de mot de passe    | PBKDF2-SHA256 | 600 000 iterations, sel de 32 o, sortie de 32 o    | NIST SP 800-132 |
| Chiffrement du coffre-fort           | AES-256-GCM   | Cle de 256 bits, nonce de 96 bits, tag de 128 bits | NIST SP 800-38D |
| Echange de cles classique            | X25519        | Curve25519, 256 bits                               | RFC 7748        |
| KEM post-quantique                   | ML-KEM-1024   | Kyber-1024, NIST Level 5                           | NIST FIPS 203   |
| Derivation de cle hybride            | HKDF-SHA256   | IKM de 64 o, info="trashmail-hybrid-kem-v1"        | RFC 5869        |
| Partage de secret                    | Shamir SSS    | k=2, n=3, GF(2 <sup>8</sup> )                      | Shamir (1979)   |
| Encodage de la cle de recuperation   | BIP39         | Entropie de 256 bits, 24 mots                      | BIP-0039        |
| Derivation de la cle de recuperation | HKDF-SHA3-256 | Sel lie au compte                                  | NIST FIPS 202   |

| Composant                                 | Algorithme                | Parametres                                  | Standard                   |
|-------------------------------------------|---------------------------|---------------------------------------------|----------------------------|
| Verification de la cle de recuperation    | SHA3-256                  | Sortie de 32 octets                         | NIST FIPS 202              |
| Chiffrement des enregistrements OPAQUE    | AES-256-GCM               | Chiffrement au repos cote serveur           | NIST SP 800-38D            |
| Deverrouillage du coffre-fort par passkey | WebAuthn PRF              | Base sur HMAC, lie au materiel              | WebAuthn Level 2           |
| Compression                               | gzip                      | Niveau 6                                    | RFC 1952                   |
| Bucket Padding                            | Personnalise              | 17 tailles (256 o–16 Mo), magic 0xDEAD      | —                          |
| Signature des reponses                    | Ed25519                   | Cle de 256 bits, signature de 512 bits      | RFC 8032                   |
| Chaine de hachage de l'archive            | SHA3-256                  | Hachage par bloc, enchainement sequentiel   | NIST FIPS 202              |
| Encapsulation de cle d'archive (CAK)      | HKDF-SHA256 + AES-256-GCM | Cle d'encapsulation derivee du mot de passe | RFC 5869 / NIST SP 800-38D |
| Verification de certificat                | SHA-256                   | Comparaison d'empreinte de certificat TLS   | —                          |
| Regroupement d'e-mails                    | SHA-256                   | Hachage du Message-ID                       | NIST FIPS 180-4            |

### 1.18.2 17.2 Niveaux de securite

| Algorithme           | Securite classique      | Securite post-quantique           |
|----------------------|-------------------------|-----------------------------------|
| X25519               | 128 bits                | Casse par l'algorithme de Shor    |
| ML-KEM-1024          | Equivalent 256 bits     | NIST Level 5 ( $\approx$ AES-256) |
| AES-256-GCM          | 256 bits                | 128 bits (algorithme de Grover)   |
| SHA-256              | 256 bits                | 128 bits (algorithme de Grover)   |
| SHA3-256             | 256 bits                | 128 bits (algorithme de Grover)   |
| Hybrid KEM (combine) | 128 bits (borne X25519) | Level 5 (borne ML-KEM)            |

### 1.19 18. Comparaison avec d'autres fournisseurs

| Fonctionnalite | Aionda Mail           | Tuta Mail           | Proton Mail |
|----------------|-----------------------|---------------------|-------------|
| <b>Pays</b>    | Allemagne (Stuttgart) | Allemagne (Hanovre) | Suisse      |

| Fonctionnalite                           | Aionda Mail                                                        | Tuta Mail                                       | Proton Mail           |
|------------------------------------------|--------------------------------------------------------------------|-------------------------------------------------|-----------------------|
| <b>Zero-Knowledge</b>                    | Oui (OPAQUE + cryptographie cote client)                           | Oui                                             | Oui                   |
| <b>Post-quantique</b>                    | Oui (ML-KEM-1024 + X25519 hybride)                                 | Oui (base sur Kyber)                            | En developpement      |
| <b>Protocole de mot de passe</b>         | OPAQUE (RFC 9807) — le mot de passe ne quitte jamais le navigateur | bcrypt (mot de passe envoye au serveur via TLS) | Base sur SRP          |
| <b>Objet chiffre</b>                     | Oui                                                                | Oui                                             | Non                   |
| <b>En-tetes chiffres</b>                 | Oui                                                                | Partiel                                         | Non                   |
| <b>Noms de contacts chiffres</b>         | Oui (dans le coffre-fort)                                          | Oui                                             | Non                   |
| <b>Adresses e-mail jetables</b>          | Oui (fonctionnalite principale, illimitees pour Plus)              | Non                                             | Oui (via SimpleLogin) |
| <b>Extension de navigateur</b>           | Oui (Chrome + Firefox)                                             | Non                                             | Via SimpleLogin       |
| <b>Partage de dossiers</b>               | Oui (Hybrid KEM par destinataire)                                  | Limite                                          | Oui                   |
| <b>Client open source</b>                | Non                                                                | Oui                                             | Oui                   |
| <b>Audit de securite</b>                 | Prevu                                                              | Oui                                             | Oui                   |
| <b>Recuperation de mot de passe</b>      | Non (par conception)                                               | Non (par conception)                            | Non (par conception)  |
| <b>Support des passkeys</b>              | Oui (FIDO2 + PRF)                                                  | Oui                                             | Oui                   |
| <b>Support PGP</b>                       | Oui (entrant + sortant)                                            | Non (protocole propre)                          | Oui (OpenPGP)         |
| <b>Archive e-mail conforme GoBD</b>      | Oui (chaine de hachage SHA3-256 + Hybrid KEM)                      | Non                                             | Non                   |
| <b>Detection MITM (ext. navigateur)</b>  | Oui (signatures Ed25519 + verification TLS)                        | Non                                             | Non                   |
| <b>Perfect Forward Secrecy (API)</b>     | Oui (cles ephemeres par requete)                                   | Inconnu                                         | Inconnu               |
| <b>Masquage de la taille des e-mails</b> | Oui (Bucket Padding)                                               | Inconnu                                         | Non                   |

## 1.20 19. Limites & transparence

### 1.20.1 19.1 Modele de confiance des applications web

Aionda Mail est une application web. A chaque chargement de page, le navigateur telecharge du JavaScript depuis les serveurs d'Aionda. Un attaquant sophistique qui compromettrait les serveurs pourrait theoriquement servir du JavaScript modifie qui exfiltrer les cles.



**Mesures d'attenuation actuelles :**

- Hachages d'integrite de sous-ressources (SRI) sur toutes les balises script
- En-tetes Content Security Policy (CSP) restreignant les sources de scripts
- Tout le code cryptographique critique est inclus dans le bundle principal de l'application
- **Extension de navigateur Guardian** (Section 14) : la verification des signatures Ed25519 sur toutes les reponses API detecte les alterations cote serveur ; la verification des certificats TLS (Firefox) detecte les proxys MITM

**Mesures d'attenuation prevues :**

- Mise en cache par Service Worker pour le fonctionnement hors ligne (reduit la frequence de confiance au chargement)

**1.20.2 19.2 Visibilite des metadonnees**

Bien que le contenu des e-mails soit entierement chiffre, certaines metadonnees sont visibles par le serveur :

- Quand les e-mails ont ete recus (horodatages)
- Quelle adresse DEA a reçu l'e-mail
- Taille approximative de l'e-mail (dans les limites des buckets)
- Schemas d'activite du compte

**1.20.3 19.3 Journal de traitement des e-mails**

A des fins de diagnostic, Aionda Mail inclut un **journal de traitement des e-mails** optionnel qui peut stocker temporairement le contenu brut des e-mails entrants. Cette fonctionnalite est configurable par adresse e-mail jetable (DEA) et peut etre activee ou desactivee dans les parametres de la DEA ("Journaliser le contenu des e-mails").

Lorsqu'il est active (par DEA) :

- Le message SMTP brut complet (en-tetes + corps) est stocke en clair sur le serveur
- Suppression automatique apres une courte periode de retention (moins de 7 jours)
- Accessible uniquement au proprietaire du compte via l'API authentifiee
- Objectif : depannage des problemes de livraison, verification du transfert, examen des decisions de filtrage anti-spam

Lorsqu'il est desactive :

- Aucun contenu d'e-mail n'est stocke dans le journal de traitement
- Seules les metadonnees sont journalisees (adresse de l'expediteur, horodatage, statut de livraison)
- Le chiffrement du coffre-fort reste le seul mecanisme de stockage

**Important :** Ce journal de traitement est independant du coffre-fort chiffre. Les e-mails stockes dans le coffre-fort sont toujours chiffres avec Hybrid KEM, quel que soit le parametre de journalisation. Le journal de traitement existe en tant que fonctionnalite heritee du systeme de transfert d'e-mails et offre une transparence operationnelle. Les utilisateurs necessitant un stockage strictement Zero-Knowledge pour tous les e-mails doivent desactiver cette option.

#### 1.20.4 19.4 Securite des e-mails externes

Les e-mails envoyes ou recus depuis des adresses non-Aionda transitent par l'infrastructure de messagerie standard (SMTP). Bien qu'ils soient stockes chiffrés dans le coffre-fort, le contenu de l'e-mail etait visible durant le transit, sauf si le chiffrement PGP etait utilise.

#### 1.20.5 19.5 Pas de sequestre de clés

Il n'existe pas de cle maitre, de porte derobee ou de mecanisme de recuperation disponible pour Aionda GmbH. Si un utilisateur perd son mot de passe et toutes les autres methodes de recuperation, ses donnees sont definitivement perdues. C'est une decision de conception intentionnelle qui prouve l'integrite du modele Zero-Knowledge.

### 1.21 20. Feuille de route

| Jalon                                      | Statut  | Objectif |
|--------------------------------------------|---------|----------|
| Architecture Zero-Knowledge                | Termine | —        |
| Hybrid KEM post-quantique (ML-KEM-1024)    | Termine | —        |
| Authentification OPAQUE (RFC 9807)         | Termine | —        |
| Shamir Secret Sharing (2-of-3)             | Termine | —        |
| Couche de transport API chiffrée           | Termine | —        |
| Support Passkey/WebAuthn PRF               | Termine | —        |
| Calendrier chiffre de bout en bout         | Termine | —        |
| Archive e-mail conforme GoBD (Blockchain)  | Termine | —        |
| Protection MITM Guardian (Ed25519)         | Termine | —        |
| Verification des certificats TLS (Firefox) | Termine | —        |

### 1.22 Historique du document

| Version | Date      | Modifications        |
|---------|-----------|----------------------|
| 1.0     | Mars 2026 | Publication initiale |

### 1.23 Contact

**Aionda GmbH** Stephan Ferraro Stuttgart, Allemagne

Email: [contact-46epp9ba@contact.aionda.com](mailto:contact-46epp9ba@contact.aionda.com) Web: <https://mail.aionda.com>

*Ce document decrit l'architecture de securite d'Aionda Mail en date de mars 2026. Les systemes cryptographiques evoluent — ce document sera mis a jour au fur et a mesure que l'architecture change.*