# INSACERMO Resonance Key V2.2

Dossier de synthèse technique et conceptuelle

| | |
|---|---|
| Statut | Démonstrateur local durci |
| Couche crypto | ChaCha20-Poly1305 / AEAD |
| Verdict | PASS_LOCAL_AEAD_DEMO |
| Usage | Échange confidentiel et audit externe |

**Le coffre n'est pas l'objectif final : il sert à montrer qu'une information peut dépendre d'un état complet, et pas seulement d'une clé ou d'une valeur isolée.**

# 2. Résumé exécutif

Resonance Key V2.2 est un démonstrateur de coffre contextuel multimodal. Il ne cherche pas à remplacer les primitives cryptographiques standards. Son originalité est dans la dérivation contextuelle de l'état : l'ouverture dépend de la reconstruction exacte d'un état informationnel complet.

Le coffre n'est pas l'objectif final : il sert à montrer qu'une information peut dépendre d'un état complet, et pas seulement d'une clé ou d'une valeur isolée.

Dans la V2.2, la couche de chiffrement pédagogique HMAC-stream + XOR + HMAC a été remplacée par une couche AEAD standard : ChaCha20-Poly1305. INSACERMO intervient comme dérivation contextuelle de l'état ; l'AEAD fournit la confidentialité et l'authentification du coffre.

La V2.2 a obtenu le statut PASS_LOCAL_AEAD_DEMO sur les tests locaux réalisés. Cela signifie que le démonstrateur résiste aux tests locaux effectués, mais ne remplace pas un audit cryptographique externe.

# 3. Principe général : de la valeur isolée à l’état informationnel

Le principe INSACERMO sous-jacent est simple :

> **Un système ne dépend pas seulement d'une valeur isolée ; il dépend d'un état informationnel cohérent.**

Dans Resonance Key :

- la passphrase seule ne suffit pas ;
- l'image seule ne suffit pas ;
- l'artefact seul ne suffit pas ;
- le coffre seul ne suffit pas ;
- l'ouverture apparaît seulement si un état complet est reconstruit.

## Phrase canonique :

> **Resonance Key ne traite pas l'image comme un secret autonome ; elle sert d'ancrage dans un état informationnel complet dont la reconstruction exacte conditionne l'ouverture.**

L'image A n'est donc pas un secret isolé. Elle est un ancrage d'état. L'artefact B est un facteur contextuel exact. La passphrase reste le secret humain. Le coffre strict, le salt, le nonce et la crête participent à la trajectoire d'ouverture.

# 4. Resonance Key comme démonstrateur cyber

Le coffre est le banc d'essai volontairement dur du principe. Il met INSACERMO dans un régime extrême : le mode exact.

Dans un modèle IA, trop de rigidité mémoire peut produire de l'overfitting et détruire la généralisation. Dans Resonance Key, cette rigidité change de valeur : elle devient une propriété de scellement.

> **Dans l'IA, la rigidification de la mémoire détruit la généralisation ; dans Resonance Key, cette même rigidification devient une propriété de scellement.**

Le démonstrateur ne cherche pas à généraliser vers des états proches. Il doit au contraire refuser toute variation non autorisée. Une image recompressée, un artefact modifié, une passphrase proche ou un coffre altéré doivent échouer.

Ce comportement n'est pas une théorie générale à lui seul. Il montre un cas limite : une information devient accessible seulement quand l'état qui la rend possible est reconstruit.

# 5. Architecture V2.2 AEAD

La V2.2 repose sur une séparation claire : dérivation contextuelle INSACERMO, couche AEAD standard et coffre strict.

## Image A

L'image A est un ancrage d'état. Elle peut être connue sans casser automatiquement le modèle, car elle ne constitue pas le secret complet.

## Artefact B

L'artefact B est un facteur contextuel exact. Il peut être multimodal : image, audio, MP3, WAV, EEG, ECG, PDF, ZIP, signal capteur, série temporelle ou document scientifique. Dans la V2.2 actuelle, il est traité en mode exact.

## Passphrase

La passphrase est le secret humain. Si image A, artefact B et coffre sont connus, la sécurité pratique repose fortement sur la force de la passphrase.

## Salt et nonce

Le salt et le nonce sont publics. Ils ne sont pas secrets. Ils servent à la dérivation et au chiffrement.

## Crête / paramètres publics

La crête et les paramètres publics définissent la trajectoire de dérivation. Ils appartiennent au protocole public attendu.

## Coffre strict

Le coffre V2.2 utilise un schéma strict. Les champs inconnus, manquants, de type incorrect ou structurellement invalides doivent être rejetés.

## ChaCha20-Poly1305 / AEAD

ChaCha20-Poly1305 fournit la confidentialité et l'authentification du coffre. Les métadonnées minimales sont authentifiées comme associated data. La V2.2 ne repose plus sur un chiffrement pédagogique maison.

# 6. Modèle de menace

Le modèle de menace retenu est défensif et local.

L'attaquant peut connaître :

- l'algorithme ;
- le coffre ;
- les paramètres publics ;
- l'image A.

L'image connue ne suffit pas. Elle n'est pas le secret complet.

## Image A connue, artefact B inconnu, passphrase inconnue

L'ouverture doit échouer. Les leurres d'artefact et les passphrases faibles ne doivent pas ouvrir le coffre.

### Image A et artefact B connus, passphrase inconnue

L'ouverture doit échouer. Dans ce cas, la résistance pratique dépend fortement de la force de la passphrase.

### Image A et passphrase connues, artefact B inconnu

L'ouverture doit échouer. Les artefacts de même taille, proches, renommés ou modifiés doivent être rejetés.

### Coffre + passphrase seuls

Cette combinaison ne doit pas suffire. C'était précisément le type de bypass critique observé en V1 à cause de l'exposition d'une empreinte d'état.

## 7. Résultats expérimentaux

Résultats consolidés :

- V2.2 : 156/156 tests PASS ;
- 1000 leurres artefact : 0 ouverture ;
- 1000 leurres image : 0 ouverture ;
- Phase 3 : 1599 tentatives négatives, 0 ouverture ;
- bypass V1 non reproduit ;
- state_digest_sha256 absent du coffre public ;
- image_sha256 et artifact_sha256 non exposés comme valeurs exploitables ;
- secret_name et secret_len non exposés en clair ;
- champ JSON racine inconnu rejeté ;
- coffre modifié ou tronqué rejeté ;
- erreurs opaques : RECONSTRUCTION_FAILED.

Ces résultats sont favorables au niveau local. Ils ne constituent pas une validation cryptographique générale.

## 8. Historique des failles et corrections

### V1 : échec critique

La V1 exposait metadata.state_digest_sha256. Cette valeur permettait de remplacer l'état A+B par une empreinte déjà présente dans le coffre. Le coffre pouvait alors être ouvert avec coffre + passphrase, sans reconstruire réellement image A + artefact B.

Verdict V1 : CRITICAL_FAIL.

### V2 : correction de la fuite d’état

La V2 a supprimé l'empreinte d'état exploitable. Le bypass V1 n'a plus été reproduit. Une faiblesse de format restait cependant présente : le parseur acceptait un champ racine inconnu.

### V2.1 strict : correction du format

La V2.1 a imposé un schéma JSON strict :

- champs racine autorisés ;
- rejet des champs inconnus ;
- rejet des champs manquants ;
- rejet des types incorrects ;
- erreurs opaques.

Verdict V2.1 : PASS_LOCAL_STRONG.

### Phase 3 : image connue

La Phase 3 a testé le modèle de menace où l'image A est connue. Le résultat local confirme que l'image connue ne casse pas le modèle tant que l'artefact B exact et/ou la passphrase exacte restent inconnus.

Verdict Phase 3 : PASS_LOCAL_THREAT_MODEL.

### V2.2 : passage AEAD

La V2.2 remplace le chiffrement pédagogique par ChaCha20-Poly1305. Cette évolution sépare plus clairement l'idée INSACERMO de la couche cryptographique standard.

Verdict V2.2 : PASS_LOCAL_AEAD_DEMO.

## 9. Lecture INSACERMO : rigidité, mémoire, scellement

Resonance Key révèle une inversion de valeur.

Dans l'apprentissage IA, trop de mémoire peut réduire la plasticité. Un modèle trop fidèle à son passé perd de la liberté pour son futur de généralisation.

Dans Resonance Key, l'absence de plasticité devient protectrice. Le coffre ne doit pas généraliser. Il doit refuser les états voisins.

INSACERMO ne dit donc pas que la rigidité est toujours mauvaise ou toujours bonne. Il cherche à lire le régime dans lequel la mémoire devient utile, dangereuse ou protectrice.

Pour apprendre, il faut conserver une marge de variation. Pour sceller, il faut réduire cette marge presque à zéro.

## 10. Angles morts identifiés

### Side-channels

Les tests ont traité les fuites évidentes : messages d'erreur, logs, comportements grossiers et timing simple.

Les side-channels fins restent hors périmètre :

- cache CPU ;
- prédiction de branchement ;
- consommation électrique ;
- variance temporelle fine ;
- skewness des distributions de temps ;
- DPA ;
- dépendances bas niveau à l'implémentation ou au matériel.

### DoS topologique / amnésie contextuelle

Le mode exact protège contre l'ouverture illégitime, mais crée un risque d'indisponibilité. Si l'image A, l'artefact B, leur support ou leur encodage exact est altéré, l'utilisateur légitime peut perdre l'état d'ouverture.

> **Le risque principal du mode exact n'est pas seulement l'extraction du secret, mais la perte irréversible de l'état d'ouverture.**

Recommandations :

- sauvegardes contrôlées des artefacts ;
- checksums privés ;
- redondance ;
- procédures de restauration ;
- documentation utilisateur stricte ;
- séparation claire entre mode exact et mode tolérant futur.

### Passphrase faible

Une passphrase faible expose le système à un risque de dictionnaire hors ligne si les autres éléments sont connus.

### Artefact B public

Si l'artefact B devient public, un facteur contextuel important est perdu.

## 11. Roadmap théorique

### V2.2 actuelle : mode exact AEAD

Le mode exact exige la reconstruction exacte de l'état. Il convient au scellement, au cold storage, à la preuve locale et à l'archive sensible.

> **Le mode exact peut être vu comme la limite où le volume de tolérance de l'espace d'état tend vers zéro.**

### V2.3 possible : DoS topologique et restauration

Une prochaine étape documentaire pourrait formaliser :

- les risques d'indisponibilité ;
- les procédures de sauvegarde ;
- les checksums privés ;
- la restauration contrôlée ;
- la documentation utilisateur.

### V3 possible : invariants informationnels

Pour des usages plus robustes, il faudrait explorer des invariants :

- spectre d'entropie ;
- corrélations internes ;
- structures fréquentielles ;
- matrices de transition informationnelle ;
- signatures moins sensibles aux métadonnées non pertinentes.

Cette approche doit rester compatible avec la sécurité et ne pas faciliter les collisions.

### V4 possible : attracteur de résonance / mode tolérant

Un futur mode tolérant pourrait être conçu comme un bassin d'attraction. Un état bruité n'ouvrirait pas directement ; il devrait converger vers une orbite stable si sa distance topologique reste dans un bassin admissible.

Des grandeurs INSACERMO comme R, E, alpha ou d50 pourraient aider à construire une métrique continue d'état.

Cette branche n'est pas implémentée dans la V2.2.

### Étape indépendante : audit externe

Avant toute extension opérationnelle :

- audit externe cryptographique ;
- revue de code ;
- revue du modèle de menace ;
- tests side-channel ;
- tests multi-OS ;
- verrouillage des dépendances.

## 12. Limites et recommandations

Limites actuelles :

- pas de preuve cryptographique formelle ;
- pas d'audit externe indépendant ;
- pas de revue side-channel complète ;
- pas de benchmark multi-OS / multi-environnements ;
- tests locaux limités en volume ;
- KDF actuel à revoir pour un produit réel ;
- sécurité pratique dépendante de la passphrase dans certains scénarios ;
- artefact B à protéger opérationnellement ;
- mode exact exposé au risque de perte d'accès légitime ;
- pas encore de mode tolérant implémenté.

Recommandations :

- conserver ChaCha20-Poly1305 ou AES-GCM comme couche AEAD ;
- auditer la dérivation contextuelle ;
- envisager Argon2id ou un KDF mémoire-dur pour une version produit ;
- verrouiller les dépendances ;
- renforcer les tests timing et side-channel ;
- documenter la gestion des artefacts ;
- séparer strictement mode exact et mode tolérant.

## 13. Conclusion

Resonance Key V2.2 démontre localement qu'un coffre peut dépendre de la reconstruction exacte d'un état informationnel complet : image A, artefact B, passphrase, salt, crête / paramètres et coffre strict.

La progression V1 -> V2 -> V2.1 -> V2.2 renforce la crédibilité du prototype parce qu'elle documente les échecs, les corrections et les limites. La V1 a montré ce qui casse le principe : exposer une empreinte d'état permet de remplacer les composants réels. La V2.1 a durci le format. La V2.2 a remplacé la couche pédagogique par AEAD standard.

Le point défendable n'est pas une nouvelle crypto. Le point défendable est la séparation entre :

- dérivation contextuelle INSACERMO ;
- couche AEAD standard ;
- modèle d'état non interchangeable.

Verdict final :

> **Resonance Key V2.2 est OK comme démonstrateur local durci, OK pour échange technique confidentiel et OK pour audit externe ; il ne doit pas être présenté**

**comme produit de sécurité fini sans audit indépendant.**

# 14. Annexes techniques courtes

## Statuts

- V1 : CRITICAL_FAIL.
- V2 : bypass V1 corrigé, faiblesse JSON détectée.
- V2.1 strict : PASS_LOCAL_STRONG.
- Phase 3 : PASS_LOCAL_THREAT_MODEL.
- V2.2 AEAD : PASS_LOCAL_AEAD_DEMO.

## Composants requis à l’ouverture

- image A exacte ;
- artefact B exact ;
- passphrase exacte ;
- salt public ;
- nonce public ;
- crête / paramètres publics ;
- coffre strict intact.

## Phrase canonique

**Resonance Key ne traite pas l'image comme un secret autonome ; elle sert d'ancrage dans un état informationnel complet dont la reconstruction exacte conditionne l'ouverture.**

## Phrase de portée

**Le coffre n'est pas l'objectif final : il sert à montrer qu'une information peut dépendre d'un état complet, et pas seulement d'une clé ou d'une valeur isolée.**