hash_tool

Analyse des créneaux non occupés

Positionnement open source · Février 2026

Objet de ce document

Ce document identifie les niches écosystémiques que hash_tool occupe et qui ne sont pas actuellement couvertes par les outils existants. Il analyse la population d'utilisateurs potentiels, les cas d'usage spécifiques, et les raisons structurelles pour lesquelles l'espace est vacant.

Table des matières

1. Panorama des outils existants

1.1 Outils de vérification d'intégrité courants

Outil	Type	Algorithme	Lacunes vis-à-vis du créneau
md5sum / sha256sum	CLI Unix	MD5 / SHA-256	Fichier par fichier, pas de dossier, pas de rapport, pas de compare
rclone check	CLI Go	Variable	Couplé au stockage cloud, pas adapté aux usages locaux/hors-ligne
Duplicati	GUI/daemon	SHA-256	Logiciel de sauvegarde complet, lourd, overkill pour la vérification seule
TeraCopy	GUI Windows	Plusieurs	Windows uniquement, propriétaire, pas scriptable
hashdeep	CLI C	Plusieurs	Pas de BLAKE3, pas de rapport HTML, pas de pipeline, inactif depuis 2015
fclones	CLI Rust	BLAKE3	Orienté déduplication, pas intégrité temporelle, pas de compare snapshots
par2	CLI C++	Reed-Solomon	Réparation de données, pas détection de corruption sur dossier existant

1.2 Conclusion de l'analyse

Aucun outil de la liste ci-dessus ne combine simultanément : **(1)** BLAKE3, **(2)** gestion de dossiers complets avec chemins relatifs, **(3)** comparaison de deux snapshots à des instants différents, **(4)** rapport HTML autonome, **(5)** pipeline JSON déclaratif, **(6)** image Docker Alpine légère. La conjonction de ces six caractéristiques définit le créneau.

2. Créneau principal : intégrité de collections de fichiers à long terme

2.1 Définition du créneau

Le créneau est celui de la **vérification d'intégrité périodique de collections de fichiers** stockées sur des supports locaux (disques durs, NAS, archives), par un opérateur technique qui n'est pas développeur, n'a pas de serveur dédié à la surveillance, et ne souhaite pas déployer un outil lourd.

2.2 Pourquoi ce créneau est vacant

Raison 1 — Le seuil de complexité des alternatives

Les outils sérieux (Duplicati, rclone, Bacula) exigent une configuration initiale significative, un daemon en arrière-plan, ou un compte cloud. Pour quelqu'un qui veut simplement savoir si ses fichiers sont intacts après un transfert ou six mois de stockage, ces outils sont disproportionnés.

Raison 2 — L'absence de BLAKE3 dans les outils shell existants

hashdeep est la référence historique pour ce type d'usage. Il n'a pas été mis à jour depuis 2015 et ne supporte pas BLAKE3. Les outils shell qui supportent BLAKE3 (`b3sum`) n'ont pas de couche d'abstraction pour la gestion de dossiers, la comparaison de snapshots, ou les rapports.

Raison 3 — L'absence de rapport lisible sans outil supplémentaire

Les outils CLI produisent du texte brut. Les opérateurs non-développeurs (archivistes, sysadmins de PME, photographes, chercheurs) ont besoin d'un résultat interprétable par un non-technicien. Un rapport HTML autonome, sans serveur, sans base de données, correspond exactement à ce besoin.

Raison 4 — L'absence de solution portable sans installation

Sur un NAS Synology, on ne peut pas facilement installer hashdeep ou un outil Go. Sur Windows, md5sum n'est pas natif. L'image Docker Alpine à 14 Mo est la première solution qui tourne de façon identique sur ces trois environnements sans aucune configuration de dépendances.

2.3 Population cible

Profil	Cas d'usage principal	Besoin spécifique non couvert
Sysadmin de PME / indépendant	Vérifier l'intégrité de sauvegardes NAS après restauration	Outil léger, sans agent, rapport lisible par le client
Photographe / vidéaste	Garantir l'intégrité d'archives de médias sur disques durs	Interface simple, pas de dépendance cloud, hors-ligne

Profil	Cas d'usage principal	Besoin spécifique non couvert
professionnel		
Archiviste numérique / bibliothèque	Détecter le bitrot sur des collections à long terme	Rapport horodaté, comparaison de snapshots, exportable
Chercheur / laboratoire	Valider l'intégrité de datasets après transfert entre systèmes	Portabilité, chemins relatifs, pas de compte tiers requis
Développeur DevOps	Intégrer une vérification d'intégrité dans un pipeline CI/CD	Mode --quiet, exit code propagé, image Docker légère

3. Créneaux secondaires

3.1 Post-transfert sur supports chiffrés

Les utilisateurs de partitions chiffrées (VeraCrypt, LUKS, BitLocker) font face à un problème spécifique : le transfert de fichiers vers ou depuis une partition chiffrée est une opération à risque (coupure d'alimentation, démontage forcé, erreur de transfert). Aucun outil dédié ne propose un workflow compute → verify → compare adapté à ce contexte. Le pipeline JSON de hash_tool s'y prête directement.

3.2 Validation de migration de données

Migrations de serveurs, changements de NAS, restructuration d'arborescences — ces opérations nécessitent de comparer l'état avant et après. Les outils existants (diff, rsync --checksum) travaillent sur des copies simultanées, pas sur des snapshots temporels. hash_tool compare deux .b3 produits à n'importe quel intervalle de temps.

3.3 Intégration CI/CD légère

Les pipelines CI qui vérifient l'intégrité d'artefacts de build ou de datasets de test utilisent généralement des checksums ad hoc (SHA-256 d'un seul fichier). hash_tool propose une approche structurée avec `--quiet`, exit code propre, et image Docker légère — sans introduire une dépendance lourde comme rclone ou un service cloud.

3.4 Archivage numérique à long terme

La communauté de l'archivage numérique (bibliothèques, musées, institutions de recherche) utilise des outils comme BagIt ou PREMIS pour l'intégrité à long terme. Ces outils sont complexes, orientés XML, et inadaptés aux petites structures. hash_tool offre un sous-ensemble fonctionnel utilisable sans formation.

4. Limites du positionnement

4.1 Ce que hash_tool n'est pas

> ! **Périmètre intentionnellement limité**
>
> hash_tool n'est pas un logiciel de sauvegarde. Il ne copie pas, ne restaure pas, ne compresse pas.
>
> hash_tool n'est pas un outil de sécurité au sens cryptographique. BLAKE3 pour l'intégrité accidentelle, pas pour l'authentification.
>
> hash_tool n'est pas un outil de surveillance temps réel. Il opère par snapshots sur demande.

4.2 Pourquoi ces limites sont un avantage

La clarté du périmètre est une qualité en open source. Les outils qui font une seule chose bien sont plus faciles à auditer, à maintenir, à tester, et à intégrer dans un pipeline plus large. hash_tool est conçu pour être une brique, pas une solution complète.

4.3 Concurrence future

Le seul risque de désintermédiation sérieux serait qu'un outil comme `rclone` ou `restic` implémente nativement BLAKE3 + comparaison de snapshots + rapport HTML + Docker léger. Leur complexité intrinsèque rend ce scénario peu probable à court terme.

5. Synthèse

Créneau	Intensité du besoin	Vacance actuelle	Priorité
Intégrité de collections de fichiers locaux à long terme	Élevée	Totale	Primaire
Post-transfert sur supports chiffrés	Moyenne	Totale	Secondaire
Validation de migration de données	Élevée	Partielle	Secondaire
Intégration CI/CD légère (BLAKE3)	Moyenne	Partielle	Tertiaire
Archivage numérique petites structures	Faible	Totale	Tertiaire

¡ **Recommandation de positionnement**

Présenter hash_tool comme "un outil de snapshot d'intégrité BLAKE3 pour collections de fichiers locales".

Ne pas le présenter comme un outil de sauvegarde ni comme un outil de sécurité.

Mettre en avant : BLAKE3, portable, sans installation (Docker), rapport HTML autonome.

Le README doit s'ouvrir sur le cas d'usage "archivage long terme" avant tout autre exemple.