Abilian
CS LAB

OPEN INTEROP · S4

S4 : la réversibilité du stockage objet, vérifiable

Stefane Fermigier (Abilian SAS) · 11 novembre 2026 · DRAFT

Version 0.4
Licence CC BY 4.0

S4 (*Sovereign S3*) est le premier chantier du groupe de travail cloud d'Open Interop. Il traite une question simple : un client peut-il quitter son fournisseur de stockage objet, récupérer toutes ses données dans un état utilisable et le prouver ? S4 formalise le standard de fait qu'est devenue l'API S3, pour les fonctions utiles à la réversibilité : une version datée, indépendante des évolutions décidées par AWS, avec des comportements vérifiables. Le chantier a été lancé lors de l'atelier Open Interop du 9 octobre 2026 (Track 6, « Connecting the Clouds »). Il produit un profil, des contrats comportementaux, une scorecard, des scénarios de test et leur implémentation, le CTK S4.

Le problème

Le stockage objet est le service managé le plus standardisé : presque tous les fournisseurs se disent « compatibles S3 ». Cette compatibilité ne garantit pourtant pas la réversibilité.

- **Les fonctions prises en charge varient** : étiquettes, versions, verrouillage, classes de stockage. Une offre « compatible S3 » ne dit pas lesquelles.
- **Les comportements diffèrent à interface identique** : pagination, cohérence, limites de débit, codes d'erreur, cas limites.
- **Certaines métadonnées ne transitent pas par l'API** : date de dernière modification, identifiants de version, ETag d'un objet multipartie.
- **L'interface est contrôlée par un seul acteur.** Au début de 2025, une modification par défaut des outils clients d'AWS a rendu de nombreux services « compatibles » inutilisables du jour au lendemain.

Le Data Act impose depuis le 12 septembre 2025 le changement de fournisseur de services cloud ; pour la circulaire du Premier ministre du 5 février 2026, la réversibilité est un critère clé du choix des solutions. Aucun référentiel technique ne permet encore de vérifier qu'une sortie a réussi.

La réponse

- **Un profil S4 ouvert et versionné.** Il définit, dans une version datée, les fonctions de l'API S3 qu'un fournisseur peut revendiquer, par niveaux : Core d'abord, puis versions, gouvernance et fonctions étendues. Une déclaration de conformité précise le niveau et les fonctions revendiquées. Voir la note constat et propositions.

- **Des contrats comportementaux.** Trente-cinq exigences, en neuf familles (export, métadonnées, intégrité, performance, interface, reconstruction, sécurité et droits, reprise, exigences organisationnelles et juridiques), décrivent le comportement attendu et la manière de le vérifier. Voir les contrats.
- **Un manifeste d'export.** Un fichier JSON Lines décrit chaque objet (clé, taille, empreinte SHA-256, métadonnées) et transporte les valeurs que la cible ne peut pas reproduire.
- **Une scorecard sans compensation.** Dans une première étape, cinq critères obligatoires (complétude, intégrité, métadonnées, droits d'accès, portabilité fonctionnelle) doivent tous être satisfaits ; durée, débit, reprise, coût et effort sont publiés à part, avec leurs conditions. Aucune moyenne pondérée ne fonde un verdict. Voir la scorecard.
- **Des scénarios de test.** Six scénarios couvrent l'export complet, l'intégrité, les métadonnées et les droits, l'incident et la reprise, l'exploitation sur la cible après révocation des accès à la source, enfin la performance. Chaque critère reçoit l'un des quatre verdicts du profil S4 : conforme, non conforme, non applicable, non concluant. Voir les scénarios.

Le CTK S4

Le CTK S4 met en œuvre cinq des six scénarios. Il migre un jeu de données déterministe d'une source vers une cible, compare les manifestes, injecte des corruptions et des coupures de connexion, fait travailler une application témoin sur la cible et produit un rapport qu'un tiers peut rejouer. Il est lui-même testé contre sept implémentations libres de l'API S3 : Garage, Ceph RGW, MinIO, SeaweedFS, Versity Gateway, RustFS et leBLOB. Son code est publié sous licence Apache 2.0 : https://git.sr.ht/~sfermigier/s4-ctk.

Ses premiers essais, au volume d'essai rapide, montrent l'écart entre déclaration et comportement :

- **Garage** est conforme au niveau Core sur les cinq scénarios, à une exception près : il ignore les lectures conditionnelles (`If-Match`), qui permettent de détecter qu'un objet a changé pendant une reprise. Il ne prend pas non plus en charge les étiquettes d'objet : revendiquées, elles donnent un verdict « non conforme ».
- **Ceph RGW**, sur son moteur de stockage autonome, ne termine jamais la lecture d'un objet multipartie dont la dernière partie fait un octet.
- **MinIO, SeaweedFS, Versity Gateway et RustFS** ne conservent pas toutes les clés difficiles que Garage conserve : tous refusent les segments `.` et `..` ; SeaweedFS et Versity Gateway réécrivent `//` en `/` sans le signaler. Une migration depuis Garage perd donc entre trois et quatre objets du jeu d'essai ; elle en stocke parfois sous une autre clé, sans jamais altérer le contenu d'un objet.
- **leBLOB** 0.4.0.dev1 (anciennement theBLOB), logiciel expérimental écrit par l'auteur du CTK, est conforme sur les cinq scénarios avec étiquettes, y compris à 10 Go entre deux instances. Il conserve les treize clés difficiles et respecte `If-Match`. Depuis MinIO, les objets que MinIO a acceptés arrivent avec leurs étiquettes. Ce résultat reste à rejouer par un tiers et ne démontre pas la réversibilité entre fournisseurs.
- **Le client AWS de référence** (boto3, depuis janvier 2025) rejette par défaut des réponses de Garage ; le CTK fixe donc le comportement de son propre client et vérifie l'intégrité par sa propre empreinte.

Aucun de ces écarts n'apparaît dans une déclaration « compatible S3 ». Voir la note le CTK S4.

La méthode

S4 suit la proposition de doctrine pour Open Interop : l'interopérabilité se prouve par un résultat reproductible, sur le périmètre revendiqué. Le rapport du CTK sert aux trois niveaux de preuve (autoévaluation par le fournisseur, vérification par un acheteur ou un tiers, essai croisé entre fournisseurs). C'est une

preuve technique : son usage dans un contrat demande une rédaction propre, qui fixe le périmètre, la mesure, les conditions de test et les conséquences d'un écart. La méthode a vocation à s'étendre à d'autres services cloud (chantier CL4 de la fiche cloud).

Calendrier

Échéance proposée	Étape	État au 11 novembre 2026
Début novembre 2026	TechSprint, CTK minimal	En cours ; CTK-01 à CTK-05 implémentés
Fin novembre 2026	Essais croisés entre fournisseurs, rapport d'écarts	À faire, au volume fonctionnel
Décembre 2026	Profil V0.1, proposition d'annexe contractuelle	À faire

Les échéances ont été proposées par l'atelier du 9 octobre 2026 et restent à confirmer par le groupe de travail. Les fournisseurs, utilisateurs et contributeurs qui souhaitent participer aux essais croisés sont les bienvenus ; la note Participer aux essais croisés détaille ce qui est demandé.

Les notes S4

Note	Objet
Par où commencer	Où trouver quoi, comment les documents s'articulent, parcours selon le rôle, lecture d'un rapport, contribution
Constat et propositions	Limites de « compatible S3 », cadre réglementaire, propositions, état du consensus, feuille de route, points ouverts
Contrats comportementaux	Les exigences du profil famille par famille, avec leur vérification
Scorecard	Critères obligatoires, mesures publiées, contexte, fiche de profil fournisseur
Scénarios de test	Les scénarios CTK-01 à CTK-06, les jeux de données, le rapport, le manifeste
Le CTK S4	Implémentation, couverture, premiers résultats, limites, suite
Participer aux essais croisés	Ce qui est demandé aux fournisseurs, déroulement d'un essai, modèle de déclaration