SAWBILL

LIVRE BLANC / 04

EVIDENCE

Prouver sans surpromettre

Evidence, attestations, provenance et auditabilité dans Sawbill

Comment Sawbill transforme des observations hostiles en preuves liées, validées, attribuées et vérifiables.

LECTEURS

Architectes / Responsables sécurité et audit / Responsables qualité et conformité / Responsables de plateforme IA

Sommaire

Prouver sans surpromettre

L'enjeu en une minute

Les systèmes agentiques produisent une abondance de traces : messages, sorties d'outils, statuts de jobs, tests, signatures, commits et logs fournisseurs. Cette abondance peut donner une illusion de preuve. Pourtant, un log peut être partiel, une signature peut couvrir les mauvais octets et un statut `Completed` peut ne rien dire sur l'objectif métier.

Sawbill transforme ces signaux en une chaîne de preuve contrôlée :

> **observer, lier à une affirmation exacte, vérifier, valider, admettre comme evidence, évaluer les gates, décider, puis conserver une histoire auditable.**

Chaque étape augmente l'utilisabilité de l'information sans élargir ce qu'elle permet réellement d'affirmer.

Des données abondantes ne constituent pas une preuve

Un artefact n'est utilisable que si l'on peut répondre à plusieurs questions :

- quels octets exacts ont été reçus ?
- de quelle source et à quel moment ?
- quelle affirmation ces octets sont-ils censés soutenir ou réfuter ?
- sur quel sujet, quelle version et quelle tentative ?
- le format, la signature et la chaîne de confiance sont-ils valides ?
- la preuve est-elle fraîche, révoquée ou rejouée ?
- le validateur possède-t-il la qualification et l'indépendance requises ?
- quelles limites de couverture ou d'interprétation subsistent ?

Sans ces bindings, une preuve devient un document voisin du sujet, pas une raison vérifiable de prendre une décision.

Le parcours unique de l'evidence

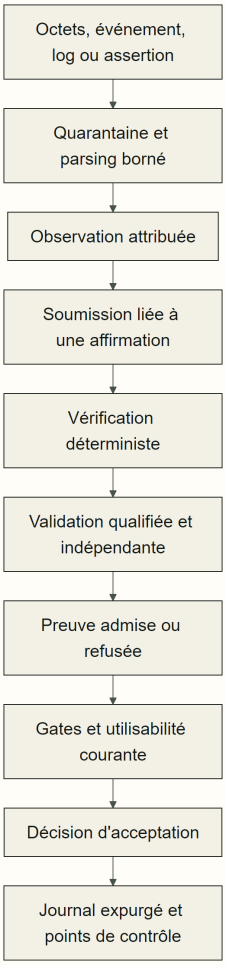

Figure 1 - Sawbill

Il n'existe pas de chemin privilégié pour un administrateur, un webhook, un fournisseur ou un agent interne. Chacun commence avec une observation hostile.

1. Observer sans encore croire

Une observation attribuée enregistre qu'un événement ou des octets ont été reçus depuis une source donnée, sous un périmètre et un cutoff donnés. Elle ne dit pas encore que le contenu est vrai ni qu'il constitue une preuve.

Cette distinction est décisive pour les intégrations :

- `Completed` devient « le fournisseur a émis ce statut » ;
- un rapport de scanner devient « ces octets ont été reçus de cet outil » ;
- une signature devient « une assertion cryptographique à vérifier » ;
- un témoignage humain devient « cette personne a formulé cette affirmation ».

Le signal est conservé sans que son producteur puisse choisir lui-même sa force probante.

L'intégration transforme les sorties d'outils en entrées contrôlées

Dans une chaîne existante, Sawbill collecte sans accorder une confiance implicite. Les webhooks, API, fichiers, logs et attestations arrivent par des adaptateurs qui préservent leur source et leur format. Ils deviennent d'abord des observations ; leur usage comme preuve est une décision séparée.

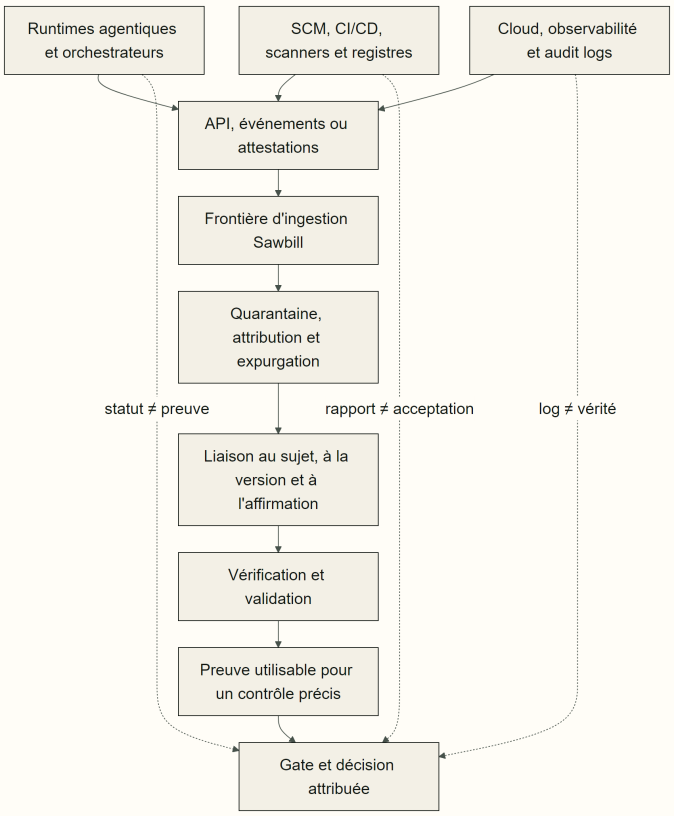

Figure 2 - Sawbill

La frontière est ainsi déplacée de la conservation de traces vers la qualité de leur liaison. Une plateforme externe peut affirmer qu'un job est terminé ; elle ne choisit ni ce que ce signal prouve, ni pour quelle version, ni avec quel poids dans une décision Sawbill.

2. Lier l'observation à une affirmation précise

Une soumission de preuve propose l'utilisation d'une observation pour un prédicat, un sujet et une finalité exacts. Elle peut soutenir ou réfuter une exigence, mais elle ne devient pas encore une evidence admise.

La liaison comprend selon le cas :

- l'identité de l'observation brute ;
- le sujet et sa version ;
- l'exigence ou le prédicat ;
- la polarité attendue ;
- la cible, la réalisation, le contrat ou le pack ;
- la tentative et le contexte d'exécution ;
- le périmètre organisationnel et technique pertinent ;
- la méthode, l'outil et la configuration ;

- la fraîcheur, la rétention et les limites déclarées.

Les mêmes octets utilisés pour deux affirmations donnent deux soumissions distinctes. L'identité de contenu peut être la même ; la relation probante ne l'est pas.

3. Donner une identité cryptographique au contenu

Un digest seul est ambigu si le format et le domaine ne sont pas connus. Sawbill associe donc l'empreinte au type de contenu, au format et à l'algorithme utilisés.

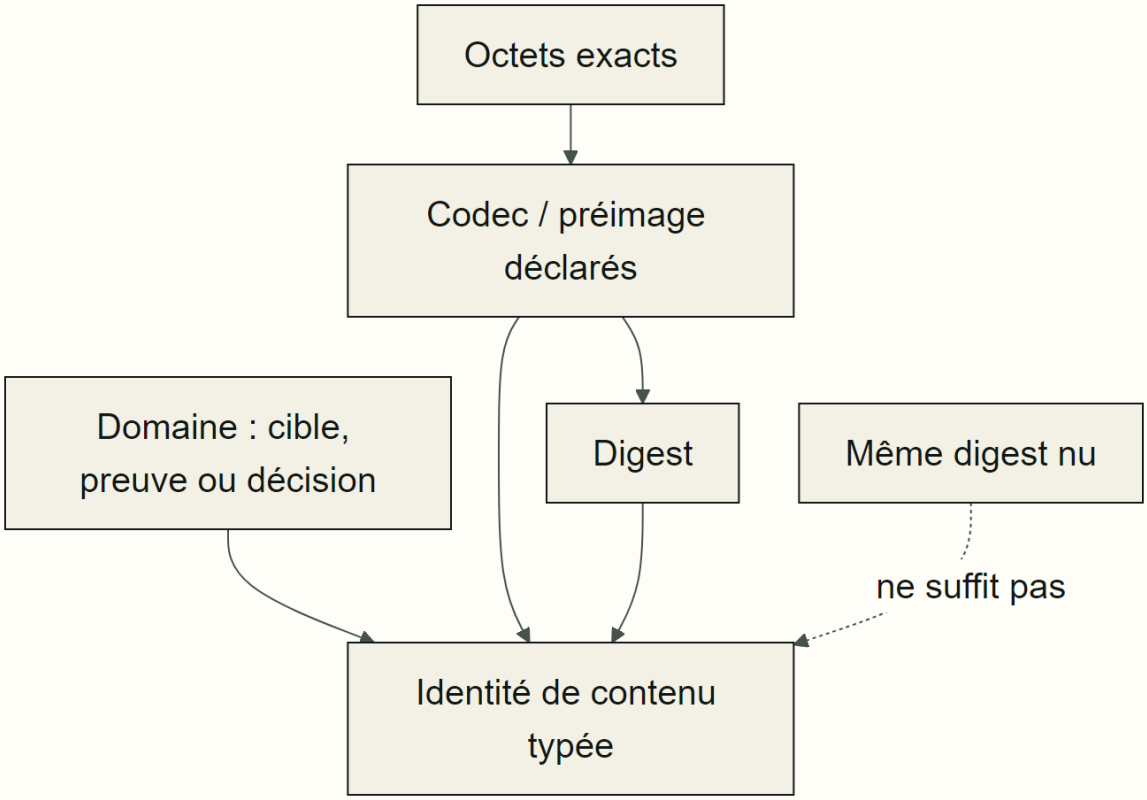

Figure 3 - Sawbill

Pour les records JSON Sawbill, un profil canonique peut s'appuyer sur le JSON Canonicalization Scheme, RFC 8785. Pour un artefact externe, Sawbill conserve l'encodage natif prévu par son standard.

Cette règle prévient les substitutions entre domaines : l'empreinte d'une cible ne devient pas silencieusement celle d'une preuve, et une re-sérialisation change l'identité cryptographique des octets concernés.

4. Authentifier une déclaration avec DSSE et in-toto

Sawbill utilise des standards de la chaîne logicielle pour transporter des attestations authentifiées :

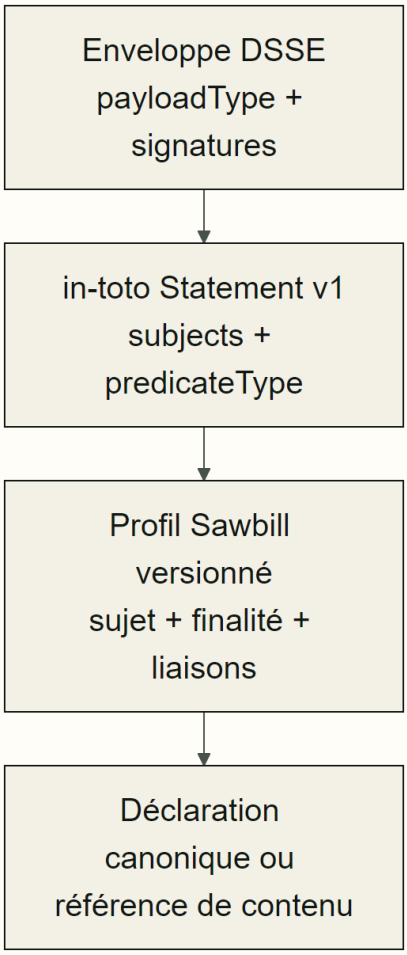

Figure 4 - Sawbill

DSSE authentifie le type et les octets du payload à l'aide d'une construction de signature définie. Le framework d'attestation in-toto lie un prédicat à un ou plusieurs sujets identifiés par digest. Sawbill applique un profil de gouvernance versionné pour transporter le sujet, la finalité et les liaisons nécessaires à la décision.

La séparation des responsabilités reste nette :

- DSSE fournit l'enveloppe de signature ;
- in-toto fournit la structure d'attestation ;
- le profil Sawbill définit les champs nécessaires à un type de record ;
- la politique Sawbill décide si le signataire, le sujet et l'usage sont admis.

Une enveloppe fraîche ne répare pas un artefact natif invalide. Un rapport externe mal formé ou une signature expirée reste invalide même s'il est ensuite enveloppé et signé par Sawbill.

5. Une signature ne dit pas que l'affirmation est vraie

La cryptographie permet d'établir des faits précis : des octets ont été signés avec une clé correspondant au profil vérifié, dans un contexte de confiance donné. Elle ne prouve pas que le contenu signé est complet, compris, applicable ou vrai.

Fait cryptographique	Établit	N'établit pas
Digest valide	Identité exacte des octets sous un codec	Origine, vérité ou autorité

Fait cryptographique	Établit	N'établit pas
Signature valide	Authenticité des octets couverts sous une clé	Droit du signataire ou vérité du payload
Credential de confiance	Acceptation du credential pour un usage	Permission métier
Horodatage	Existence d'une donnée avant un instant sous le modèle de la TSA	Exactitude sémantique du contenu
Preuve d'inclusion	Présence d'une feuille dans un checkpoint	Vérité, exhaustivité ou acceptation

Sawbill sépare donc vérification cryptographique et validation sémantique.

6. Vérifier mécaniquement, valider sémantiquement

Une vérification déterministe possède un prédicat fermé : recalculer un digest, valider une signature, comparer des bytes ou évaluer un schéma.

Une validation sémantique examine une affirmation qui ne possède pas d'oracle total : adéquation d'une architecture, réalisation d'une intention, qualité d'une explication ou applicabilité d'une exigence. Elle doit rester :

- attribuée à son validateur ;
- liée aux sources et contre-preuves examinées ;
- limitée par une méthode et un scope ;
- contestable et réévaluable ;
- indépendante selon les axes exigés par le profil.

Un deuxième appel au même modèle n'est pas nécessairement une validation indépendante. Sawbill peut examiner principal, organisation, modèle, outil, credential, runtime et domaine de contrôle pour établir le niveau d'indépendance réellement démontré.

7. Admettre une evidence pour un seul usage

Après vérification et validation, Sawbill peut admettre une preuve utilisable pour un prédicat précis. Son état n'est pas seulement positif ou négatif :

- admise ;
- rejetée ;
- devenue obsolète ;
- révoquée ;
- remplacée ;
- inconnue ;
- bloquée.

Une evidence historique n'est jamais réécrite. Un changement de politique, une révocation, une nouvelle contre-preuve ou une nouvelle version ajoute un record d'invalidation ou de réévaluation.

Cette propriété empêche le passé de changer silencieusement tout en obligeant les consommateurs à vérifier l'utilisabilité actuelle.

8. Les gates et l'acceptation restent deux décisions

Une gate évalue un vecteur complet de contrôles. Si un contrôle obligatoire manque, si une evidence est périmée ou si sa polarité est inconnue, la gate ne peut pas devenir favorable.

L'acceptation intervient ensuite comme une décision de gouvernance : l'autorité compétente accepte ou refuse un sujet exact sur la base de gates et d'evidence exactes.

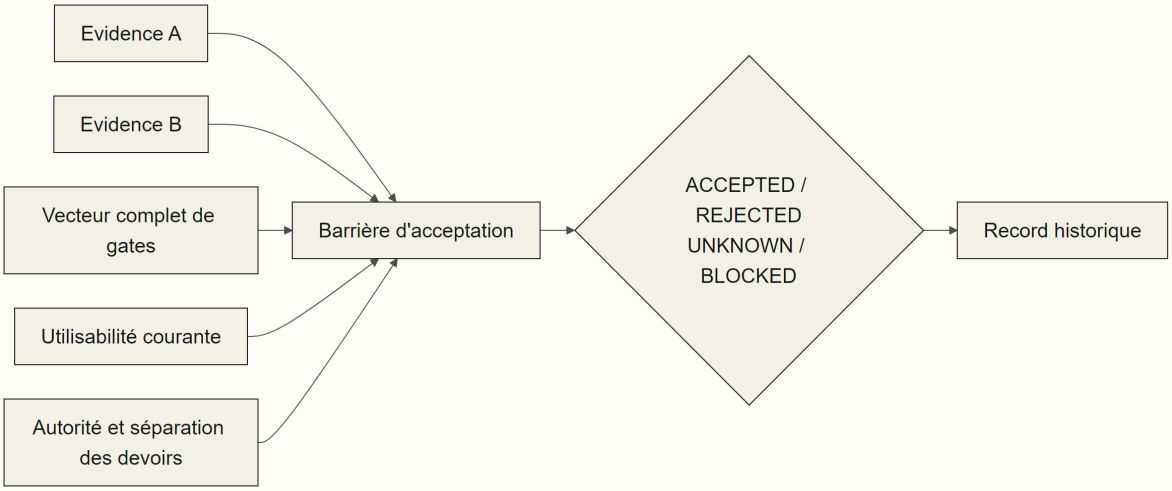

Figure 5 - Sawbill

Une décision `ACCEPTED` ne vaut pas permission de fusionner ou déployer. L'effet externe traverse encore son propre chemin d'autorité, d'admission et de contrôle.

9. Un ledger append-only, avec des garanties bornées

Le ledger Sawbill conserve des observations expurgées et leurs relations causales. Il ne devient pas un second moteur de décision. Une entrée de ledger ne fabrique ni evidence, ni gate, ni acceptation.

Pour détecter les modifications et fournir des preuves d'inclusion, le ledger peut utiliser une structure de Merkle et des checkpoints signés :

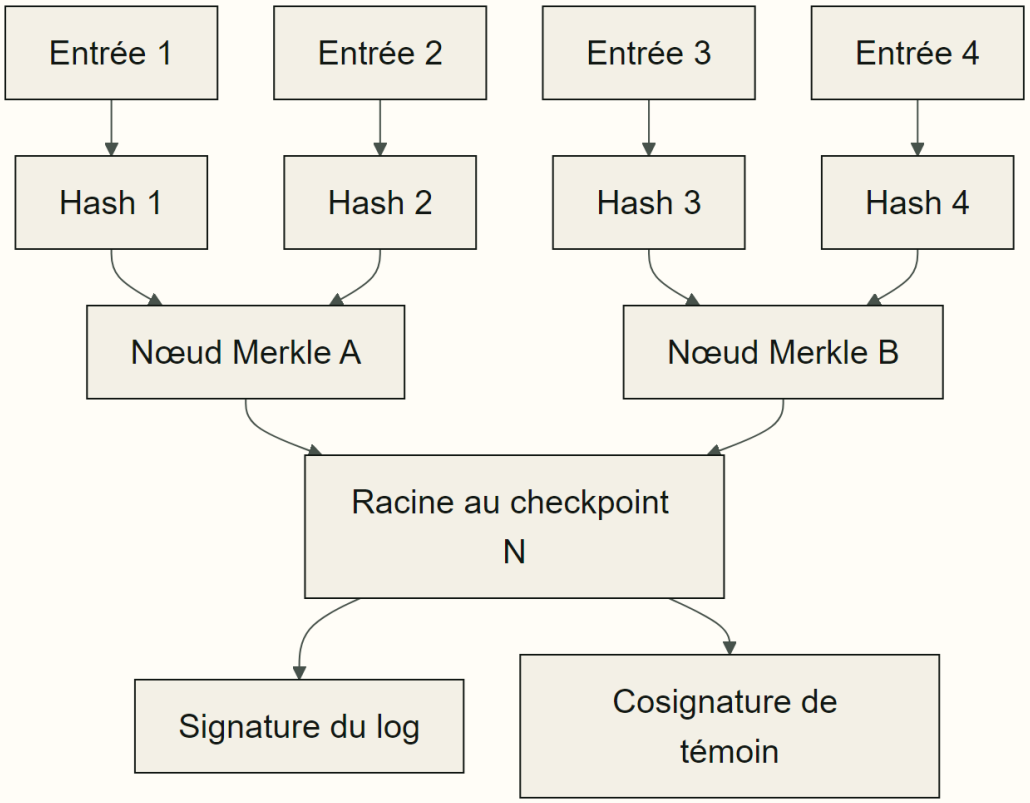

Figure 6 - Sawbill

La RFC 9162 décrit Certificate Transparency v2 et ses journaux append-only auditables. Les spécifications C2SP tlog-tiles et tlog-witness fournissent des formats et protocoles ouverts pour les points de contrôle, preuves et témoins. Sawbill borne leur usage par une origine identifiée, une couverture déclarée et une politique de clés explicite.

10. Ce que prouve réellement la transparence

Fait de ledger	Peut soutenir	Ne prouve pas
Entrée retenue à une séquence donnée	Rétention et ordre bornés	Vérité du contenu
Inclusion dans un checkpoint	Présence d'une feuille dans cet arbre	Capture de tous les événements du monde
Consistance entre deux checkpoints	Extension cohérente d'un préfixe pour une origine	Non-équivocation globale entre toutes les vues
Checkpoint cosigné	Un témoin nommé a vu ce checkpoint	Indépendance si le témoin et le log ont le même contrôle
Checkpoint exporté	Possibilité de comparaison ultérieure	Continuité avant l'ancrage exporté

Un journal local hash-chainé peut détecter des modifications internes ; sans checkpoint externe, il ne prouve pas à lui seul qu'une fin d'historique n'a pas été supprimée. Sawbill expose cette différence plutôt que d'employer le mot « immuable » comme garantie absolue.

11. Fraîcheur, révocation et anti-rejeu

Une assertion historiquement valide peut ne plus être utilisable. Sawbill lie les vérifications à :

- un instant ou cutoff ;
- un snapshot de confiance ;
- un état de révocation ;
- un contexte anti-rejeu ;
- un sujet, une tentative et une finalité ;
- une durée de fraîcheur définie par le profil.

La réutilisation protégée lie l'assertion, le signataire et la finalité à un contexte déterminé. Changer de tentative ou de ressource exige un nouveau contexte ; recopier la même signature ne suffit pas.

Pour lier une donnée au temps, un profil peut utiliser le Time-Stamp Protocol de la RFC 3161, mis à jour par la RFC 5816. L'horodatage reste une preuve d'existence bornée, pas un verdict sur le contenu.

12. Confidentialité et garde des clés

Une preuve utile ne doit pas devenir une fuite durable. Sawbill classe et expurge les données avant leur persistance, leur indexation ou leur entrée dans le ledger. Les secrets, clés privées, bearer tokens et données personnelles brutes ne sont pas ajoutés à une attestation « pour faciliter l'audit ».

Les clés restent derrière des ports de signature ou de custody. Selon le profil, Sawbill peut s'intégrer à des interfaces standard comme PKCS #11 v3.2 ou KMIP v2.1. Le chiffrement par enveloppe sépare la clé de données de la clé de wrapping et lie le ciphertext à son scope via des données authentifiées.

Chiffrer ne prouve ni l'autorité, ni la résidence, ni la destruction de toutes les copies. Ces affirmations nécessitent leurs propres observations et preuves.

Scénario : reconstruire l'acceptation de l'export de données

Pour la fonction d'export, un auditeur veut savoir pourquoi la version a été acceptée.

Sawbill peut reconstruire :

1. les identités exactes de la cible et de la réalisation ;
2. les observations reçues des tests, scanners et environnements ;
3. l'affirmation précise soutenue par chaque soumission ;
4. les résultats de vérification cryptographique ;
5. les validations et leur indépendance démontrée ;
6. les preuves admises, refusées ou devenues obsolètes ;
7. le vecteur complet de gates utilisé ;
8. l'autorité et le cutoff de la décision d'acceptation ;
9. les entrées de ledger et leurs relations causales ;
10. les reçus et observations de la mutation externe ultérieure.

Si la configuration de résidence était inaccessible, l'audit voit `UNKNOWN`. Si une evidence a été révoquée après l'acceptation, l'histoire conserve le verdict initial et l'invalidation ultérieure. Ni l'un ni l'autre n'est effacé.

Valeur pour l'entreprise

Une auditabilité reconstruite, pas racontée

L'audit part d'identités de contenu, de relations et de décisions attribuées plutôt que d'une chronologie de captures d'écran.

Des preuves portables

DSSE, in-toto, SLSA et les formats de transparence facilitent la vérification par des outils et parties indépendantes, sans enfermer toute la chaîne dans un format propriétaire.

Des limites visibles

La distinction entre signature, validation, evidence, gate et acceptation réduit les conclusions excessives et les faux sentiments de conformité.

Un historique compatible avec le changement

Rotation, révocation, correction et réévaluation ajoutent des événements. Ils ne réécrivent pas le passé et ne maintiennent pas silencieusement une confiance devenue obsolète.

Références officielles

Source	Statut externe	Apport utilisé	Limite conservée par Sawbill
RFC 8785 - JSON Canonicalization Scheme	RFC informative	Représentation JSON canonique pour hashing et signature	Le codec et le domaine restent explicitement profilés
DSSE v1	Spécification ouverte maintenue	Enveloppe et préimage de signature avec type authentifié	DSSE ne fournit ni gestion des clés ni trust policy
in-toto Attestation Framework v1.2	Version courante du framework d'attestation	Liaison entre sujets et prédicats authentifiés	Le prédicat Sawbill porte la sémantique de gouvernance propre au produit
SLSA v1.2	Version publiée de la spécification	Provenance et niveaux de chaîne de build et de release	La provenance de build ne remplace pas l'acceptation du livrable
RFC 3161 - Time-Stamp Protocol	IETF Proposed Standard, mis à jour par RFC 5816	Preuve d'existence d'une donnée avant un instant	L'horodatage ne prouve pas la vérité du contenu
RFC 9162 - Certificate Transparency v2	RFC expérimentale, remplace RFC 6962	Constructions de journaux Merkle auditables	Le profil Sawbill borne l'origine, la couverture et la portée des garanties
C2SP tlog-tiles v0.1.0	Spécification C2SP versionnée	Distribution statique de checkpoints, entrées et tuiles Merkle	Le transport du log ne décide pas de l'evidence
C2SP tlog-witness v1.0.0	Spécification C2SP versionnée	Cosignature de checkpoints après vérification de consistance	L'indépendance dépend du contrôle réel du témoin
PKCS #11 v3.2	OASIS Standard, 3 juin 2026	Interface avec tokens et modules cryptographiques	L'interface de custody ne définit pas les politiques Sawbill
KMIP v2.1	OASIS Standard, 14 décembre 2020	Interopérabilité de gestion de clés	La gestion de clés ne devient pas une décision d'autorité

Ces mécanismes sont adoptés dans leur périmètre propre. Sawbill ne réinvente pas les primitives cryptographiques et ne transforme jamais la popularité d'un outil en garantie automatique.