RECEVOIR VOTRE PISTE D'AUDIT DANS

Elastic

Recevez la piste d'audit REVIEWIT dans un flux de données Elasticsearch par l'API _bulk — Elastic Cloud, Serverless ou auto-hébergé : un modèle d'index, une clé d'API à deux privilèges, pas de Logstash, pas d'agent, pas d'intégration Fleet.

VERSION
1.0 · Octobre 2026

DURÉE
Vingt minutes, en trois étapes

DIFFUSION
Confidentiel · Clients REVIEWIT

CE QU'IL VOUS FAUT

01 **Un administrateur Elastic** avec accès aux Outils de développement de Kibana sur votre déploiement, capable de poser un modèle d'index et de créer une clé d'API à privilèges restreints.

02 **Un administrateur de la souscription** REVIEWIT. Créer, activer et remplacer un identifiant exigent une authentification forte de moins de dix minutes : la console vous envoie chez votre fournisseur d'identité et vous ramène.

03 **L'ordre de ce guide** : d'abord la cible chez vous, puis la destination dans REVIEWIT, puis le test. Rien n'est livré avant qu'une écriture réelle sous vos identifiants ait réussi.

LE PARCOURS, EN TROIS ÉTAPES

ÉTAPE 1

Préparer la cible

Créer le point de réception et l'identifiant que REVIEWIT utilisera — droits minimaux, rien de plus.

ÉTAPE 2

Déclarer la destination

Quatre écrans dans le Subscription Manager : cible, résidence, contenu, récapitulatif.

ÉTAPE 3

Tester, puis activer

Une écriture de sonde sous vos identifiants ; un refus dit pourquoi. La livraison commence dans la minute.

DANS CE GUIDE

1 Comment BEACON livre votre piste d'audit

2 Ce que vous recevez

3 Préparer la cible chez Elastic

4 Déclarer la destination dans REVIEWIT

5 Tester et activer

6 Vérifier hors ligne avec beacon-verify

7 Exploiter

8 Dépannage et liste de contrôle

01 Comment BEACON livre votre piste d'audit

BEACON est le canal de diffusion de la piste d'audit REVIEWIT. Il ne produit pas la preuve et ne possède pas les événements : **la piste d'audit prouve, REVIEWIT SEAL scelle, BEACON livre.** Chaque événement d'audit devient un maillon — un rang strictement croissant, le SHA-256 de l'enregistrement, l'empreinte du maillon précédent et l'enregistrement original verbatim — et la tête de chaque chaîne est signée par l'autorité de scellement. Ce qui arrive chez vous est une chaîne que vous recalculez et un sceau que vous contrôlez, sans jamais rappeler REVIEWIT.

VOTRE RÉGION

Piste d'audit

Événements chaînés par domaine et classe de rétention, scellés toutes les heures par REVIEWIT SEAL.

→

VOTRE RÉGION

BEACON

Une passe par minute : des lots de 500 événements ou 5 Mo au plus, un manifeste par lot, un battement toutes les 15 minutes.

→

CHEZ VOUS

Elastic

L'API _bulk de votre déploiement Elasticsearch, dans le flux de données `logs-reviewit.audit-default`, sous une clé d'API à deux privilèges.

PRINCIPE DIRECTEUR

Rien n'est perdu et rien n'est prétendu. La livraison avance chaîne par chaîne, sur un acquittement strict seulement ; un échec reprend exactement où il s'est arrêté ; après douze tentatives, la destination est mise en pause et une alerte est levée. Une purge de rétention est livrée comme une coupe signée, un filtre laisse des jalons qui gardent la chaîne vérifiable, et le silence est couvert par le battement : jamais de trou non signé.

PRIS EN CHARGE PAR

La carte **Elastic** de l'assistant de destination (profil Elasticsearch). Elastic Cloud Hosted, Serverless et auto-hébergé 8.x. L'authentification est une **clé d'API** que vous créez et confiez, scellée sous garde REVIEWIT SEAL, ne portant que `create_doc` et `auto_configure` sur votre flux : REVIEWIT ne lit rien, ne liste rien, ne supprime rien. L'acquittement est strict : une réponse 2xx **et** `errors: false` ; un bulk partiellement refusé est un lot refusé. Prouvé de bout en bout le 19 septembre 2026 sur Elastic Cloud Serverless.

02 Ce que vous recevez

Trois sortes d'enregistrements voyagent vers votre cible, toutes en JSON UTF-8 dont l'ordre des propriétés est le contrat. Un **maillon** est un événement d'audit : `metadata.uid` (l'identifiant stable de l'événement), `reviewit.chain` (clé, genèse), `reviewit.integrity` (`sequence`, `previous_hash`, `hash`, `raw_data`). Un **manifeste** clôt chaque lot : chaîne, plage, empreintes d'entrée et de sortie, décomptes, `batch_id`, et la référence du dernier sceau de la chaîne. Un **battement** porte, toutes les 15 minutes, la tête et la position acquittée de chaque chaîne.

ENREGISTREMENT	OÙ IL ARRIVE	À QUOI IL SERT
`kind: link`	Un document par maillon dans `logs-reviewit.audit-default` ; `@timestamp` est l'heure de l'occurrence ; `event` est le maillon (type `flattened`)	L'événement d'audit, avec ses champs de preuve et `raw_data` verbatim
`kind: manifest`	Un document par lot, manifeste nu ; `@timestamp` est l'heure de l'envoi	Plage, empreintes, décomptes, `batch_id`, référence de sceau
`kind: heartbeat`	Un document toutes les 15 minutes	L'état de chaque chaîne — alertez sur son absence
`kind: probe`	Un document à la création, à chaque test et à l'activation	L'écriture de sonde qui prouve l'hôte, le flux, la clé et ses privilèges

AU MOINS UNE FOIS, DÉDOUBLONNÉ PAR VOUS

Après un doute réseau, le même lot peut être présenté de nouveau, avec le même `batch_id` et les mêmes identifiants d'événements. Dédoublonnez sur `metadata.uid`, et ordonnez entre chaînes par l'heure de l'occurrence. Un enregistrement qui n'arrive jamais est un trou que votre SIEM peut voir : chaque maillon est numéroté, et le battement dit où se tient la tête.

LE PULL, MÊME CONTRAT

Votre SIEM peut aussi lire la piste lui-même par l'API publique REVIEWIT, sous la portée `audit.read` : `GET /api/v1/audit/chains` liste les chaînes de votre souscription avec leur curseur de départ, `GET /api/v1/audit/stream?cursor=…&limit=500&manifest=true` rend les mêmes maillons et les mêmes manifestes que le push. Push et pull se vérifient avec le même outil.

03 Préparer la cible chez Elastic

Un administrateur Elastic pose, une fois, le modèle d'index qui type le flux, puis crée la clé d'API qui est l'identité de REVIEWIT dans votre déploiement. Les deux se font dans Kibana › Outils de développement › Console.

01 Le modèle d'index

Collez le modèle ci-dessous et exécutez-le : `{ "acknowledged": true }`. **`index.mode: standard` n'est pas décoratif** : sur Serverless (et sur 8.17+ selon la configuration) un flux `logs-*` naît en mode `logsdb`, dont le `_source` est synthétique — les entiers deviennent des chaînes, les tableaux sont réordonnés — et `beacon-verify` refuse alors l'export. Le mode `standard` garde le `_source` tel que reçu. Le champ `event` est `flattened` : indexé tel quel, cherchable en KQL, jamais pris par les correspondances dynamiques ECS.

02 La clé d'API

Exécutez la seconde requête. La réponse porte `id`, `api_key` et `encoded` : c'est `encoded` que vous confiez à REVIEWIT (section 4). Elle ne s'affiche qu'une fois ; si elle est perdue, invalidez la clé et créez-en une autre. Une échéance (`"expiration": "365d"`) est votre choix : à l'échéance, la destination passe en pause pour échecs et le renouvellement est votre geste. Le même geste existe dans Kibana › Gestion › Clés d'API › Restreindre les privilèges.

03 Ne créez pas le flux de données

Le premier document de REVIEWIT — la sonde de la section 5 — le crée d'après le modèle. Un autre nom de flux (`logs-reviewit.audit-<espace>`) ou un index simple convient, s'il correspond au motif du modèle, à l'URL de la section 4 et aux privilèges de la clé.

04 Vérifiez le mode après le premier document

`GET logs-reviewit.audit-default/_settings?filter_path=*.settings.index.mode` doit répondre `standard`. Si le flux a déjà été créé en `logsdb`, corrigez le modèle puis `POST logs-reviewit.audit-default/_rollover` : le nouvel index d'écriture prend le bon mode.

MODÈLE D'INDEX, PUIS CLÉ D'API (KIBANA DEV TOOLS)

```
PUT _index_template/reviewit-audit
{
  "index_patterns": ["logs-reviewit.audit-*"],
  "data_stream": {},
  "priority": 500,
  "template": {
    "settings": { "index": { "mode": "standard" } },
    "mappings": { "properties": {
      "@timestamp": { "type": "date" },
      "kind": { "type": "keyword" },
      "event": { "type": "flattened" } } }
  }
}

POST _security/api_key
{
  "name": "reviewit-beacon",
  "role_descriptors": { "reviewit_beacon": { "indices": [
    { "names": ["logs-reviewit.audit-*"], "privileges": ["create_doc", "auto_configure"]
} ] } },
  "metadata": { "application": "REVIEWIT BEACON", "purpose": "audit trail delivery (_bul
k)" }
}
```

CE QUE VOUS SAISIREZ DANS L'ASSISTANT

CHAMP DE L'ASSISTANT	VALEUR
Transport	**Elastic** (API _bulk d'Elasticsearch)
Nom	Comment la destination se lira dans votre liste — « Elastic — projet sécurité ».
URL de la cible	L'**URL `_bulk` complète** du flux : `https://<projet>.es.<région>.<fournisseur>.elastic.cloud/logs-reviewit.audit-default/_bulk` (Serverless : l'adresse Elasticsearch du projet, pas celle de Kibana `…kb…` ; Hosted : l'adresse du déploiement en 443 — 9243 n'est pas admis par la garde d'egress ; auto-hébergé : votre hôte HTTPS). Nom en minuscules, pas de requête.
Clé d'API	La valeur `encoded` de l'étape 02 (ou `id:api_key`).
Géographie de la cible	La région de votre projet ou déploiement (`europe-west1` = EUR, `us-central1` = USA…). Déclarative.

04 Déclarer la destination dans REVIEWIT

Subscription Manager › **Paramètres** › **Intégrations & IA** › **BEACON** › **Nouvelle destination**. L'assistant a quatre écrans ; rien n'est créé avant le dernier, et la cible est contrôlée par la garde d'egress avant que la destination existe.

01 Cible

Nommez la destination, choisissez la carte **Elastic**, saisissez l'URL de la cible et l'identifiant de la section 3. L'identifiant est ouvert en mémoire pour ce seul geste, puis scellé sous garde REVIEWIT SEAL : l'écran ne le montre plus jamais.

02 Résidence

Déclarez la **géographie de votre projet ou déploiement**. Pour un point HTTP elle est déclarative. Si elle diffère de la géographie de votre souscription REVIEWIT, une justification écrite est exigée et consignée dans la piste d'audit : la piste quitte sa géographie, et c'est votre décision.

03 Contenu

Départ : depuis le début de chaque chaîne (historique complet) ou depuis l'activation (seulement ce qui suit). **Événements envoyés en clair** : laissez vide pour le flux complet. Un filtre remplace chaque événement écarté par un jalon — rang et empreintes, aucune donnée métier — et la preuve scellée s'arrête au premier jalon ; l'écran le dit avant que vous confirmiez.

04 Récapitulatif

Relisez, puis cliquez **Créer**. La garde d'egress juge d'abord la cible : HTTPS seulement, aucun identifiant dans l'URL, port 443 ou 8088, un hôte routable publiquement. Une cible refusée nomme le contrôle fautif ; rien n'est créé.

LA GARDE D'EGRESS, AVANT CHAQUE ENVOI

Chaque cible passe les mêmes sept contrôles à la création, à chaque test et avant chaque livraison : URL valide · HTTPS · aucun identifiant dans l'URL · port 443 ou 8088 · aucune zone interne REVIEWIT · l'hôte se résout · l'hôte n'est pas une adresse privée ou réservée. Les redirections ne sont jamais suivies, et la vérification TLS n'est jamais désactivée.

05 Tester et activer

Une destination n'est activée qu'après une écriture réelle sous vos identifiants. Ouvrez le panneau de la destination et cliquez **Tester la cible** : REVIEWIT écrit un enregistrement de sonde (`kind: probe` sur le flux). « La cible passe la garde d'egress » signifie que l'hôte, l'identifiant et ses droits sont prouvés. Un refus dit sa cause — jamais le secret.

01 Tester la cible

REVIEWIT écrit un document `probe` sur le flux sous la clé. « La cible passe la garde d'egress » signifie que l'hôte, le flux, la clé et ses privilèges sont prouvés, et le flux de données existe désormais (`GET _data_stream/logs-reviewit.audit-default`). Un refus nomme sa cause : **clé d'API refusée** (401 — clé fausse ou invalidée) ; **clé sans privilège sur le flux** (403 — descripteur de rôle incomplet, ou nom de flux hors du motif) ; **un élément du lot est refusé** (200 avec `errors` — le plus souvent un mapping : le modèle manque ou sa priorité est trop basse).

02 Activer

Depuis le panneau, **Activer** (ou cochez « Activer dès la création » dans l'assistant). La livraison commence à la passe suivante, dans la minute ; le premier battement suit.

03 Lire le panneau

Par chaîne : le point de reprise acquitté, l'âge du plus ancien événement non livré, la marge avant la prochaine coupe de rétention, le dernier sceau et ce qui est au-delà. L'historique des lots liste chaque manifeste ; le journal du canal liste chaque geste d'administration avec son code d'audit.

06 Vérifier hors ligne avec beacon-verify

`beacon-verify` est un outil en ligne de commande, distribué avec sa spécification publique. Il lit ce que votre cible a reçu et le trousseau que REVIEWIT publie à `/.well-known/seal-keyring.json`, ancré sur l'empreinte de la clé racine imprimée dans l'annexe sécurité de votre contrat, et rend un verdict par chaîne et par plage — sans réseau et sans appel à REVIEWIT.

THE COMMAND

```
beacon-verify <export> --keys seal-keyring.json --root <empreinte racine> --namespace <v
otre espace de scellement>
```

Votre **espace de scellement** est affiché dans le panneau de la destination (`t-` suivi de 26 caractères). Passez-le avec `--namespace` : un sceau d'un autre espace est, pour vous, une rupture. Sans `--root` ni `--namespace`, les plages sont rendues scellées *sous réserve*.

01 **Chercher, trié, en lots entiers**

`GET logs-reviewit.audit-default/_search?size=10000&sort=@timestamp:asc` avec le corps `{ "query": { "terms": { "kind": ["link", "manifest"] } } }`. Au-delà de 10 000 documents, paginez par `search_after` ou exportez par fenêtres de `@timestamp` — sans jamais couper un lot.

02 **Copier la réponse telle quelle**

Panneau de sortie de la console › ⋮ › Copier, dans `search.json`. Le vérificateur lit `hits.hits[]._source` et l'objet du contrat dans `event`. Un NDJSON des seuls objets `_source` (`jq -c '.hits.hits[]._source'`) se vérifie de la même façon.

03 **Lancer le vérificateur**

`beacon-verify search.json --keys seal-keyring.json --root <racine> --namespace <t-…>`. L'export de l'essai du 19 septembre 2026 — 1 747 maillons, 7 manifestes — a rendu le code 3 (chaînes scellées avant la liaison NON VÉRIFIABLE, comme attendu) ; un octet altéré a rendu ROMPU au rang 100, code 1.

VERDICT	SIGNIFICATION
`SCELLÉ`	Chaîné jusqu'à une tête dont la signature est valide, trousseau ancré, espace épinglé.
`SCELLÉ SOUS RÉSERVE`	Idem, mais la racine ou l'espace n'ont pas été épinglés sur la ligne de commande.
`CHAÎNAGE SEUL`	Intact, au-delà du dernier sceau ou derrière un jalon — le prochain sceau le couvrira.
`NON VÉRIFIABLE`	Un sceau sans espace, un algorithme inconnu ou une clé absente du trousseau — jamais dit faux.
`ROMPU`	Une empreinte fausse, un trou, un doublon divergent, une signature invalide ou un espace étranger — avec la cause et le rang.

Codes de sortie : **0** rien de rompu et tout ce qui précède le dernier sceau est scellé · **1** une rupture ou un espace étranger · **2** entrée inexploitable (une propriété JSON en double, une valeur à la fois brute et enveloppée, un enregistrement tronqué à l'indexation) · **3** rien de rompu, preuve incomplète.

07 Exploiter

Chaque geste sur une destination est posé par un administrateur de la souscription, tracé dans la piste d'audit sous son nom, et les gestes sensibles exigent une authentification forte de moins de dix minutes. Le canal qui exporte la piste est lui-même sous la piste.

Renouveler la clé d'API	Créez la nouvelle clé (section 3), puis panneau › **Remplacer la clé d'API** ; invalidez ensuite l'ancienne (`DELETE _security/api_key` avec son `id`, ou Kibana). Le remplacement est sec : l'ancienne clé cesse d'être utilisée aussitôt. Prouvé en direct le 19 septembre 2026 : clé invalidée, clé remplacée, sonde et deux lots livrés.
Pause pour échecs	Un refus persistant — identifiant révoqué, droit retiré, cible disparue — met la destination en pause après douze tentatives et lève une alerte dans le Centre d'alertes. Rien n'est perdu : la reprise repart du point de reprise acquitté de chaque chaîne, automatiquement quand la sonde horaire revoit la cible, ou par **Reprendre**.
Changer de cible	**Modifier** › nouvelle URL, même forme. La clé doit porter les privilèges sur le nouveau nom ; **Tester la cible** le prouve.
Retirer	**Désactiver**, puis supprimer ou archiver. Une destination qui a livré est archivée, jamais supprimée : ses manifestes sont des preuves de livraison. Révoquez ensuite l'identifiant de votre côté : il ne sert plus à rien.

08 Dépannage

CE QUE VOUS VOYEZ	CAUSE	CE QU'IL FAUT FAIRE
Clé d'API refusée (401) au test	Valeur `encoded` fausse, ou clé invalidée	Créez une nouvelle clé ; remplacez-la dans le panneau ; retestez.
Clé sans privilège sur le flux (403)	Descripteur de rôle sans `create_doc` ou `auto_configure`, ou nom de flux hors de `logs-reviewit.audit-*`	Recréez la clé avec le descripteur de la section 3 ; gardez le nom de flux dans le motif.
« Un élément du lot est refusé » (200 avec errors)	Le modèle d'index ne s'applique pas : absent, priorité inférieure à un autre, ou un mapping dynamique a pris un champ	`GET _index_template/reviewit-audit` ; montez `priority` ; assurez-vous que `event` est `flattened`.
beacon-verify : « entrée inexploitable » (code 2) sur un export propre	Le flux est en mode `logsdb` : `_source` synthétique, entiers réécrits en chaînes	Corrigez le modèle (`index.mode: standard`), `_rollover` ; les documents déjà écrits dans l'ancien index restent synthétiques.
La garde d'egress refuse le port 9243	Elastic Cloud Hosted écoute aussi en 9243	Utilisez l'adresse 443 du déploiement.
Destination en pause, « après échecs »	Clé invalidée ou expirée, privilège retiré, refus de mapping répété	Corrigez côté Elastic, **Tester la cible**, puis **Reprendre**. Rien n'a été perdu.

Liste de contrôle avant de commencer

6 POINTS

- [ ] Le modèle d'index `reviewit-audit` posé, avec `index.mode: standard` et `event` en `flattened`
- [ ] Une clé d'API `reviewit-beacon` avec `create_doc` + `auto_configure` sur `logs-reviewit.audit-*`, sa valeur `encoded` sous la main
- [ ] L'URL `_bulk` Elasticsearch du flux, en 443, en minuscules, sans requête
- [ ] La région du déploiement connue, et sa justification si elle diffère de celle de votre souscription
- [ ] Un administrateur de la souscription avec son second facteur prêt
- [ ] L'empreinte racine de l'annexe de votre contrat, pour beacon-verify

Ce guide est fourni aux clients REVIEWIT qui raccordent leur SIEM ou leur stockage à BEACON. Les chemins de menus sont ceux des consoles des éditeurs à la date de publication ; ils peuvent évoluer. En cas d'écart, le support REVIEWIT vous accompagne.