RECEIVE YOUR AUDIT TRAIL IN

Elastic

Receive the REVIEWIT audit trail in an Elasticsearch data stream through the _bulk API — Elastic Cloud, Serverless or self-hosted: an index template, an API key with two privileges, no Logstash, no agent, no Fleet integration.

VERSION
1.0 · October 2026

DURATION
Twenty minutes, in three steps

DISTRIBUTION
Confidential · REVIEWIT customers

WHAT YOU NEED

01 **An Elastic administrator** with access to Kibana Dev Tools on your deployment, able to put an index template and create an API key with restricted privileges.

02 **A subscription administrator** of your REVIEWIT subscription. Creating, activating and replacing a credential ask for a strong authentication of less than ten minutes: the console sends you to your identity provider and brings you back.

03 **The order of this guide**: first the target on your side, then the destination in REVIEWIT, then the test. Nothing is delivered before a real write under your credentials has succeeded.

THE PATH, IN THREE STEPS

STEP 1

Prepare the target

Create the receiving endpoint and the credential REVIEWIT will use — minimal rights, nothing else.

STEP 2

Declare the destination

Four screens in the Subscription Manager: target, residence, content, review.

STEP 3

Test, then activate

A probe write under your credentials; a refusal says why. Delivery starts within the minute.

IN THIS GUIDE

1 How BEACON delivers your audit trail

2 What you receive

3 Prepare the target at Elastic

4 Declare the destination in REVIEWIT

5 Test and activate

6 Verify offline with beacon-verify

7 Operate

8 Troubleshooting and checklist

01 How BEACON delivers your audit trail

BEACON is the delivery channel of the REVIEWIT audit trail. It does not produce the evidence and does not own the events: **the audit trail proves, REVIEWIT SEAL seals, BEACON delivers.** Every audit event becomes a link — a strictly increasing sequence, the SHA-256 of the record, the hash of the previous link, and the original record verbatim — and the head of every chain is signed by the sealing authority. What arrives at your target is a chain you can recompute and a seal you can check, with no call back to REVIEWIT.

YOUR REGION

Audit trail

Events chained per domain and retention class, sealed every hour by REVIEWIT SEAL.

→

YOUR REGION

BEACON

A pass every minute: batches of up to 500 events or 5 MB, a manifest per batch, a heartbeat every 15 minutes.

→

YOUR SIDE

Elastic

The _bulk API of your Elasticsearch deployment, into the data stream `logs-reviewit.audit-default`, under an API key with two privileges.

GUIDING PRINCIPLE

Nothing is lost and nothing is pretended. Delivery advances chain by chain, only on a strict acknowledgement; a failure resumes exactly where it stopped; after twelve failed attempts the destination is paused and an alert is raised. A retention purge is delivered as a signed cut, a filter leaves stubs that keep the chain verifiable, and silence is covered by the heartbeat: never an unsigned gap.

SUPPORTED THROUGH

The **Elastic** card of the destination wizard (Elasticsearch profile). Elastic Cloud Hosted, Serverless, and self-hosted 8.x. Authentication is an **API key** you create and entrust, sealed under REVIEWIT SEAL custody, carrying only `create_doc` and `auto_configure` on your stream: REVIEWIT reads nothing, lists nothing, deletes nothing. Acknowledgement is strict: a 2xx answer **and** `errors: false`; a partially refused bulk is a refused batch. Proven end to end on 19 September 2026 on Elastic Cloud Serverless.

02 What you receive

Three kinds of record travel to your target, all as UTF-8 JSON whose property order is the contract. A **link** is one audit event: `metadata.uid` (the stable event identifier), `reviewit.chain` (key, genesis), `reviewit.integrity` (`sequence`, `previous_hash`, `hash`, `raw_data`). A **manifest** closes each batch: chain, range, entry and exit hashes, counts, `batch_id`, and the reference of the latest seal of the chain. A **heartbeat** carries, every 15 minutes, the head and acknowledged positions of every chain.

RECORD	WHERE IT LANDS	WHAT IT IS FOR
`kind: link`	One document per link in `logs-reviewit.audit-default`; `@timestamp` is the occurrence time; `event` is the link (type `flattened`)	The audit event, with its proof fields and `raw_data` verbatim
`kind: manifest`	One document per batch, bare manifest; `@timestamp` is the send time	Range, hashes, counts, `batch_id`, seal reference
`kind: heartbeat`	One document every 15 minutes	State of every chain — alert on its absence
`kind: probe`	One document at creation, at every test and at activation	The probe write that proves the host, the stream, the key and its privileges

AT-LEAST-ONCE, DEDUPLICATED BY YOU

After a network doubt, the same batch may be presented again with the same `batch_id` and the same event identifiers. Dedupe on `metadata.uid`, and order across chains by occurrence time. A record that never arrives is a gap your SIEM can see: every link is numbered, and the heartbeat says where the head stands.

THE PULL, SAME CONTRACT

Your SIEM can also read the trail itself through the REVIEWIT public API, with the scope `audit.read`: `GET /api/v1/audit/chains` lists the chains of your subscription with their start cursor, `GET /api/v1/audit/stream?cursor=…&limit=500&manifest=true` returns the same links and the same manifests as the push. Push and pull verify with the same tool.

03 Prepare the target at Elastic

An Elastic administrator puts, once, the index template that types the stream, then creates the API key that is REVIEWIT's identity in your deployment. Both are done in Kibana › Dev Tools › Console.

01 The index template

Paste the template below and run it: `{ "acknowledged": true }`. `index.mode: standard` is **not decorative**: on Serverless (and on 8.17+ depending on configuration) a `logs-*` stream is born in `logsdb` mode, whose `_source` is synthetic — integers become strings, arrays are reordered — and `beacon-verify` then refuses the export. The `standard` mode keeps the `_source` as received. The `event` field is `flattened`: indexed as is, searchable by KQL, never caught by ECS dynamic mappings.

02 The API key

Run the second request. The response carries `id`, `api_key` and `encoded`: it is `encoded` that you entrust to REVIEWIT (section 4). It is shown once; if lost, invalidate the key and create another. An expiry (`"expiration": "365d"`) is your choice: at expiry the destination pauses for failures and the renewal is your gesture. The same gesture exists in Kibana › Management › API keys › Restrict privileges.

03 Do not create the data stream

REVIEWIT's first document — the probe of section 5 — creates it from the template. Another stream name (`logs-reviewit.audit-<space>`) or a plain index is fine as long as it matches the template pattern, the URL in section 4 and the key's privileges.

04 Check the mode after the first document

`GET logs-reviewit.audit-default/_settings?filter_path=*.settings.index.mode` must answer `standard`. If the stream was already created in `logsdb`, fix the template then `POST logs-reviewit.audit-default/_rollover`: the new write index takes the right mode.

INDEX TEMPLATE, THEN API KEY (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)" }
}
```

WHAT YOU WILL ENTER IN THE WIZARD

WIZARD FIELD	VALUE
Transport	**Elastic** (Elasticsearch _bulk API)
Name	How the destination will read in your list — "Elastic — security project".
Target URL	The **full `_bulk` URL** of the stream: `https://<project>.es.<region>.<provider>.elastic.cloud/logs-reviewit.audit-default/_bulk` (Serverless: the Elasticsearch address of the project, not Kibana's `…kb…`; Hosted: the deployment address on 443 or 9243 — 9243 is not allowed by the egress guard, use 443; self-hosted: your HTTPS host). Lowercase name, no query string.
API key	The `encoded` value from step 02 (or `id:api_key`).
Target geography	The region of your project or deployment (`europe-west1` = EUR, `us-central1` = USA…). Declarative.

04 Declare the destination in REVIEWIT

Subscription Manager › **Settings** › **Integrations & AI** › **BEACON** › **New destination**. The wizard has four screens; nothing is created before the last one, and the target is tested against the egress guard before the destination exists.

01 Target

Name the destination, choose the **Elastic** card, enter the target URL and the credential from section 3. The credential is opened in memory for this action only, then sealed under REVIEWIT SEAL custody: the screen never shows it again.

02 Residence

Declare the **geography of your project or deployment**. For an HTTP endpoint it is declarative. If it differs from the geography of your REVIEWIT subscription, a written justification is required and recorded in the audit trail: the trail leaves its geography, and that is your decision.

03 Content

Start: from the beginning of each chain (full history) or from activation (only what follows). **Events sent in full**: leave empty for the full flow. A filter replaces every excluded event with a stub — sequence and hashes, no business data — and the sealed proof stops at the first stub; the screen says so before you confirm.

04 Review

Check the summary and click **Create**. The egress guard judges the target first: HTTPS only, no credentials in the URL, port 443 or 8088, a publicly routable host. A refused target names the failing check; nothing is created.

THE EGRESS GUARD, BEFORE EVERY SEND

Every target passes the same seven checks at creation, at every test and before every delivery: valid URL · HTTPS · no credentials in the URL · port 443 or 8088 · no internal REVIEWIT zone · host resolves · host is not a private or reserved address. Redirects are never followed, and TLS verification is never disabled.

05 Test and activate

A destination is activated only after a real write under your credentials has succeeded. Open the destination panel and click **Test the target**: REVIEWIT writes one probe record (`kind: probe` on the stream). "The target passes the egress guard" means the host, the credential and its rights are proven. A refusal says its cause — never the secret.

01 Test the target

REVIEWIT writes one `probe` document on the stream under the key. "The target passes the egress guard" means the host, the stream, the key and its privileges are proven, and the data stream now exists (`GET _data_stream/logs-reviewit.audit-default`). A refusal names its cause: **API key refused** (401 — wrong or invalidated key); **key without privilege on the stream** (403 — incomplete role descriptor, or a stream name outside the pattern); **an item of the bulk is refused** (200 with `errors` — most often a mapping: the template is missing or its priority too low).

02 Activate

From the panel, **Activate** (or tick "Activate as soon as it is created" in the wizard). Delivery begins at the next pass, within the minute; the first heartbeat follows.

03 Read the panel

Per chain: the acknowledged checkpoint, the age of the oldest undelivered event, the margin before the next retention cut, the latest seal and what lies beyond it. The batch history lists every manifest; the channel journal lists every administrative gesture with its audit code.

06 Verify offline with beacon-verify

`beacon-verify` is a command-line tool distributed with its public specification. It reads what your target received and the keyring REVIEWIT publishes at `/.well-known/seal-keyring.json`, anchored on the root fingerprint printed in your contract's security annex, and renders a verdict per chain and per range — with no network and no call to REVIEWIT.

THE COMMAND

```
beacon-verify <export> --keys seal-keyring.json --root <root fingerprint> --namespace <y
our sealing namespace>
```

Your **sealing namespace** is shown in the destination panel (`t-` followed by 26 characters). Pass it with `--namespace`: a seal from another namespace is, for you, a break. Without `--root` and `--namespace`, ranges are reported as sealed *with reserve*.

01 **Search, sorted, in whole batches**

`GET logs-reviewit.audit-default/_search?size=10000&sort=@timestamp:asc` with body `{ "query": { "terms": { "kind": ["link", "manifest"] } } }`. Beyond 10 000 documents, paginate with `search_after` or export by `@timestamp` windows — never cutting a batch.

02 **Copy the response as is**

Console output panel › ⋮ › Copy, into `search.json`. The verifier reads `hits.hits[]._source` and the contract object in `event`. An NDJSON of the `_source` objects alone (`jq -c '.hits.hits[]._source'`) verifies the same way.

03 **Run the verifier**

`beacon-verify search.json --keys seal-keyring.json --root <root> --namespace <t-…>`. The trial export of 19 September 2026 — 1 747 links, 7 manifests — rendered exit 3 (chains sealed before the binding NOT VERIFIABLE, as expected); one byte altered rendered BROKEN at rank 100, exit 1.

VERDICT	MEANING
`SEALED`	Chained up to a head whose signature is valid, keyring anchored, namespace pinned.
`SEALED, WITH RESERVE`	Same, but the root or the namespace were not pinned on the command line.
`CHAIN ONLY`	Intact, beyond the last seal or behind a stub — the next seal will cover it.
`NOT VERIFIABLE`	A seal without namespace, an unknown algorithm or a key absent from the keyring — never called false.
`BROKEN`	A wrong hash, a gap, a divergent duplicate, an invalid signature or a foreign namespace — with the cause and the rank.

Exit codes: **0** nothing broken and everything before the last seal is sealed · **1** a break or a foreign namespace · **2** unusable input (a duplicated JSON property, a value both raw and wrapped, a record truncated at indexing) · **3** nothing broken, proof incomplete.

07 Operate

Every gesture on a destination is made by a subscription administrator, traced in the audit trail under their name, and the sensitive ones ask for a strong authentication of less than ten minutes. The channel that exports the trail is itself under the trail.

Renew the API key	Create the new key (section 3), then panel › **Replace the API key**; invalidate the old one afterwards (`DELETE _security/api_key` with its `id`, or Kibana). The replacement is dry: the previous key stops being used at once. Proven live on 19 September 2026: key invalidated, key replaced, probe and two batches delivered.
Pause for failures	A persistent refusal — a revoked credential, a removed right, a target gone — pauses the destination after twelve attempts and raises an alert in the Alerts Center. Nothing is lost: resumption restarts at the acknowledged checkpoint of every chain, automatically when the hourly probe sees the target again, or with **Resume**.
Change the target	**Edit** › new URL, same form. The key must carry the privileges on the new name; **Test the target** proves it.
Retire	**Disable**, then remove or archive. A destination that delivered is archived, never deleted: its manifests are delivery proofs. Revoke the credential on your side afterwards: it serves no purpose any more.

08 Troubleshooting

WHAT YOU SEE	CAUSE	WHAT TO DO
API key refused (401) at test	Wrong `encoded` value, or key invalidated	Create a new key; replace it in the panel; test again.
Key without privilege on the stream (403)	Role descriptor missing `create_doc` or `auto_configure`, or stream name outside `logs-reviewit.audit-*`	Recreate the key with the descriptor of section 3; keep the stream name within the pattern.
"An item of the bulk is refused" (200 with errors)	The index template is not applied: missing, lower priority than another, or a dynamic mapping caught a field	`GET _index_template/reviewit-audit`; raise `priority`; make sure `event` is `flattened`.
beacon-verify: "unusable input" (exit 2) on a clean export	The stream is in `logsdb` mode: synthetic `_source`, integers rewritten as strings	Fix the template (`index.mode: standard`), `_rollover`; documents already written in the old index stay synthetic.
The egress guard refuses port 9243	Elastic Cloud Hosted also listens on 9243	Use the 443 address of the deployment.
Destination paused, "after failures"	Key invalidated or expired, privilege removed, repeated mapping refusal	Fix on the Elastic side, **Test the target**, then **Resume**. Nothing was lost.

Checklist before you start

6 ITEMS

- [ ] The index template `reviewit-audit` put, with `index.mode: standard` and `event` as `flattened`
- [ ] An API key `reviewit-beacon` with `create_doc` + `auto_configure` on `logs-reviewit.audit-*`, its `encoded` value at hand
- [ ] The Elasticsearch `_bulk` URL of the stream, on 443, lowercase, no query string
- [ ] The region of the deployment known, and its justification if it differs from your subscription's
- [ ] A subscription administrator with their second factor ready
- [ ] The root fingerprint from your contract annex, for beacon-verify

This guide is provided to REVIEWIT customers who connect their SIEM or storage to BEACON. Menu paths are those of the vendors' consoles at publication date; they may change. In case of discrepancy, REVIEWIT support will assist you.