RECEIVE YOUR AUDIT TRAIL IN

CrowdStrike Falcon

Receive the REVIEWIT audit trail in CrowdStrike Falcon Next-Gen SIEM (and Falcon LogScale) through its HEC data connector: the connector's API key, entrusted once, no agent, no intermediate component.

VERSION
1.0 · October 2026

DURATION
Thirty minutes, in three steps

DISTRIBUTION
Confidential · REVIEWIT customers

WHAT YOU NEED

01 **A Falcon administrator** with the right to create a data connector in Next-Gen SIEM (Data onboarding › Data connectors), and the repository that will hold the trail.

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 CrowdStrike Falcon

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

CrowdStrike Falcon

The HEC data connector of your Falcon Next-Gen SIEM, under the connector's API key.

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 **CrowdStrike** card of the destination wizard — a preset of the HTTP Event Collector profile: same envelope, same transport, same strict acknowledgement (`{"code":0}`). Authentication: BEACON sends `Authorization: Splunk <key>`; if the connector answers 401, it retries once with `Bearer` and keeps the scheme for the pass. Nothing to configure. The HEC profile is proven end to end on Splunk; the proof on a Falcon tenant is scheduled with REVIEWIT's trial access — see Troubleshooting if your connector acknowledges differently.

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
`reviewit:audit`	One event per link in the connector's repository; the event payload is the link	The audit event, with its proof fields and `raw_data` verbatim
`reviewit:audit:manifest`	One event per batch, bare manifest	Range, hashes, counts, `batch_id`, seal reference
`reviewit:audit:heartbeat`	One event every 15 minutes	State of every chain — alert on its absence

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 CrowdStrike Falcon

A Falcon administrator creates **a HEC data connector dedicated to REVIEWIT**. Falcon shows the connector URL and the API key once; both go into the REVIEWIT wizard and nowhere else.

01 **Next-Gen SIEM › Data onboarding › Data connectors › Add connection**

Search **HEC / HTTP Event Collector** and open it.

02 **Connector settings**

Name: **REVIEWIT BEACON**. Parser: **none** (or a JSON parser): every event is a JSON object; the link's fields are queried as `reviewit.chain.key`, `reviewit.integrity.sequence`, and `raw_data` holds the original record verbatim. Repository: the one that will hold the trail, with the retention you want.

03 **Save, then copy the API URL and the API key**

Falcon displays the connector's **API URL** — `https://<instance>.ingest.<region>.crowdstrike.com/api/ingest/hec/<id>/v1/services/collector` — and the **API key**, once. The key is entered in the REVIEWIT wizard (section 4) and never shown again on either side.

04 **Falcon LogScale (self-managed or cloud)**

Same path from the repository: Settings › Ingest tokens › New token (parser: none). The collector URL is `https://<your logscale host>/api/v1/ingest/hec` — enter it as is, or its origin: BEACON completes a URL that ends in `/services/collector` and uses a full URL untouched.

WHAT YOU WILL ENTER IN THE WIZARD

WIZARD FIELD	VALUE
Transport	**CrowdStrike** (Falcon Next-Gen SIEM — HEC data connector)
Name	How the destination will read in your list — "Falcon — SOC".
Target URL	The connector URL shown by Falcon (`…/api/ingest/hec/<id>/v1/services/collector`), used as is. No query string, no credentials in the URL.
API key	The key Falcon showed once at creation.
Target geography	The region of your Falcon instance (US-1, US-2, EU-1, US-GOV-1…). 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 **CrowdStrike** 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 **region of your Falcon instance** — the host of the connector URL names it (`ingest.us-1`, `ingest.eu-1`…). 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.

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 (a test event under the connector's key). "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 posts one event to the connector URL under the key — `Splunk` scheme first, `Bearer` on a 401 — and expects a 2xx answer with `{"code":0}`. A refusal names its cause: **HEC HTTP 401/403** — key invalid or connector disabled; **HEC HTTP 404** — wrong connector URL (copy it again from Falcon); **HTTP 2xx without acknowledgement** — the connector answered without a `code` field (see Troubleshooting).

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 the repository over an ingestion window**

`#repo=<repository> #sourcetype=reviewit:audit*` with the time picker on the **ingest** timestamp (`@ingesttimestamp`), wide enough to hold whole batches. Links carry the occurrence time in `@timestamp`; manifests and stubs carry the ingestion time: never bound the export by `@timestamp`.

02 **Export the raw events**

Export › **JSON** or **NDJSON** of the `@rawstring` field — one JSON object per line, exactly as received. Do not export a table view: a value both raw and wrapped is refused by the verifier (exit code 2).

03 **Run the verifier**

`beacon-verify export.ndjson --keys seal-keyring.json --root <root> --namespace <t-…>`. Exit 0 when everything before the last seal is sealed; 3 when a chain has no seal yet under your namespace (expected right after activation — the next hourly seal covers it).

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.

Rotate the API key	Create a new connector key in Falcon (or a new connector), then panel › **Replace the API key**. The replacement is dry: the previous key stops being used at once. Revoke the old key in Falcon afterwards.
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 is kept.
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
Every send refused with HEC HTTP 401	The key was revoked, or the connector disabled	Create a new key; replace it in the panel; **Resume**.
HEC HTTP 404 on every batch	The target URL is not the connector URL Falcon shows	**Edit** › paste the connector URL again, unchanged; **Test the target**.
"HEC HTTP 200 without acknowledgement (code absent)"	The connector acknowledged without the standard `{"code":0}` body	Send REVIEWIT support the exact response body: the HEC profile acknowledges only on `code: 0`, by design, and the mapping for your connector's answer is a product change, not a setting.
Destination paused, "after failures"	Twelve consecutive refusals (key, URL, network)	Read the cause in the panel, fix it, **Test the target**, then **Resume**. Nothing was lost.
beacon-verify reports BROKEN on an export that looks complete	The export was bounded by `@timestamp`, or truncated at ingestion	Export by ingestion window, raw events only; run again. If the break holds, contact REVIEWIT support with the chain key and the rank.
The egress guard refuses the target	A query string or credentials in the URL, or a non-public host	Paste the connector URL exactly as Falcon shows it; Falcon ingestion hosts are public on 443.

Checklist before you start

6 ITEMS

- [ ] A HEC data connector dedicated to REVIEWIT, parser none or JSON
- [ ] Its connector URL and API key, copied once from Falcon
- [ ] A repository with the retention you want
- [ ] The region of your Falcon instance 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.