Verity Edge: continuity test records and evidence mapping

Version 3, 7 Oct 2026. Source: public continuity-event record (api.verityedge.eu) and the backend audit log. Site hetzner-demo, agent demo-hetzner-01.

Controlled test on our own Hetzner (EU) test environment, not a customer site. No customer data is involved.

1. What was tested

An agent runs at a site and sends status data to the Verity Edge backend. In this test the agent collected status metrics of its own test host (CPU, memory, disk, network and uptime); no application or database content. For each test we blocked the agent host's outbound connection to the backend for a set time. The host and the agent kept running; only the link was cut. While cut off, the agent stored payloads on local disk, and after the link came back it sent them to the backend. The question tested: does a loss of connection leave a silent gap, or is it detected, timed and accounted for?

2. Recorded events in this pack

Test	Event id	Offline start (UTC)	Offline end (UTC)	Min	Stored payloads	Field "lossless"	Record SHA-256 (first 16)
30 min cut	`de8c2310`	2026-09-29 13:48:49	2026-09-29 14:19:20	31	12	true	`b0370d4cbf5eee70`
60 min cut	`5a272fea`	2026-09-29 14:19:21	2026-09-29 15:19:53	61	24	true	`8fd7e1e96a19323d`
120 min cut	`ae278b52`	2026-09-29 15:19:53	2026-09-29 17:20:28	121	48	true	`d169a85be193c200`
26 min cut	`3547cfa1`	2026-09-30 09:41:16	2026-09-30 10:07:23	26	10	true	`6011736930052d76`

How to read the fields. *Offline start* is when the backend last received data before the gap; *offline end* is when it received the first data after the gap. Both are backend clock times in UTC, shown to the second (fractions cut off). A gap is recorded only when it is longer than 90 seconds. The three September 29 cuts ran back to back, so each began right after the previous one ended. *Stored payloads* is the number of payloads the agent held locally and reported when it reconnected. *"lossless"* is a field name in our system: the backend sets it to true when the reconnecting agent reports at least one stored payload. It is not a count of payloads received; delivery is checked separately in section 3.

Sampling rate during an outage. While disconnected, the agent stores one payload per failed delivery attempt. Each attempt waits for a network timeout, so it stored about one payload every 2.5 minutes, against the normal interval when connected (default 30 seconds). The outage is covered, but at a lower sampling rate.

The 26-minute cut was planned as a test in which the backend is crashed during the outage. We have not yet confirmed that the crash hit the backend process that serves the agent, so we do not claim it as a crash test here. It is included as a recorded 26-minute cut.

3. What the backend keeps, and its integrity

- **What is kept:** for each gap, an outage event (start, end, duration, stored payloads). Each payload the agent sends, including the stored ones after reconnecting, is written as an entry in the backend audit log. The agent deletes a stored payload only after the backend has answered that it received it.
- **Delivery check, 7 Oct 2026:** during a drain, each heartbeat in the audit log records how many stored payloads are still waiting. A read-only check on the live backend confirmed that for every outage below, each countdown value appeared exactly once, meaning every stored payload was delivered exactly once,

with none missing and none duplicated: 30 min 12 of 12, 60 min 24 of 24, 120 min 48 of 48, 26 min 10 of 10, and the 24-hour cut 576 of 576. Each drain took between under one second and eight seconds.

- **What is not kept yet:** the backend does not store the content of the stored payloads with their original collection time. The outage period can therefore be shown as detected and accounted for, but its measurements cannot be reconstructed from this record.
- **Audit-log integrity check, 6 Oct 2026:** the audit log is hash-chained: each entry contains the SHA-256 hash of the entry before it. A read-only check on the live backend recomputed every hash and every link from the first entry to the last. Result: 311,306 entries, 0 broken links, 0 hash mismatches. Tail hash (first 16 characters): `07098c6577a2608e`.
- **Limit of that check:** it shows the chain is internally consistent. Someone with write access to the storage could in theory rewrite and re-hash the whole chain. Stating the tail hash in this dated report creates a record outside our system; a later check must still find that hash in the chain.

4. What this evidence supports

Reference framework: NIS2 Directive (EU) 2022/2555, Article 21(2). In the Netherlands this is implemented through the Cyberbeveiligingswet and the Cyberbeveiligingsbesluit, both in force since 15 August 2026. ISO/IEC 27001:2022 Annex A references are added because many auditors work from them.

Requirement	What this record contributes	What it does not cover
NIS2 Art. 21(2)(c): business continuity, such as backup management and disaster recovery, and crisis management	Shows that a loss of connection does not create a silent gap: it is detected and timed, the agent kept collecting locally (at a reduced rate) and sent the stored payloads after reconnecting.	The entity's continuity and recovery plans, backup restores, crisis management. Reconstruction of measurements during the outage.
NIS2 Art. 21(2)(b): incident handling	Each loss of connection is recorded as an event with a start time, end time and duration. This gives a fixed timeline to compare with the incident register.	Detection, analysis, response, root-cause analysis, incident reporting.
NIS2 Art. 21(2)(f): policies and procedures to assess the effectiveness of cybersecurity risk-management measures	A repeatable test (cut, store, restore, record) with recorded results that can be re-run on demand.	An assessment of the entity's overall measures.
ISO/IEC 27001:2022 A.8.15 Logging, A.8.16 Monitoring activities	Outages of the monitoring link are logged as events; the audit log is hash-chained and was verified intact.	Which assets are logged, alert thresholds, log review.
ISO/IEC 27001:2022 A.5.28 Collection of evidence, A. 5.30 ICT readiness for business continuity	This report gives a SHA-256 hash for each record and for the pack (method in section 5), so later changes are detectable. The monitoring link was tested against disruption.	The entity's evidence procedure and wider ICT continuity.

Commission Implementing Regulation (EU) 2024/2690, Annex section 3.2 (monitoring and logging), is relevant as a technical reference. It applies directly only to specific digital service providers (for example cloud, data centre and managed service providers). **Whether this evidence is sufficient for a given entity is the auditor's judgement. It supports specific measures; it does not demonstrate compliance on its own.**

5. How to verify this record (optional)

The pack hash fixes the four records in section 2 at one moment. If anyone changed them later, the hash would no longer match. It takes a few minutes and needs Python 3 (free).

```
Pack hash (SHA-256): 9992a52cabd21084f932af8f7228157c64f3221a5c618da815c9429d81696578
```

1. Save the two files sent with this report in one folder: `verify_pack.py` and `continuity_events_snapshot_2026-10-06.json`.

2. In a terminal in that folder, run: `python verify_pack.py continuity_events_snapshot_2026-10-06.json`. You should see **MATCH**.
3. To check the live system as well, run `python verify_pack.py` with no file name. MATCH means the same records are still served today.

Method (used for the pack hash and for each record hash in section 2): the record(s), sorted by event id, serialised as JSON with sorted keys and no spaces, hashed with SHA-256. This shows the records have not changed since this report. It does not prove independently that the outages happened, because both copies come from Verity Edge. For that, compare the times with the site's own network or firewall logs.

6. Every other event in the same public record

Listed so nothing is left out. All are on the same test agent. Start times are cut to the minute. The four 23 June events carry the organisation id "demo"; the later events carry "ve-ops". Same agent and site.

Event id	Offline start (UTC)	Min	Stored	"lossless"	Context
`7f906433`	2026-06-23 10:14	31	12	true	First confirmatory pass (30/60/120)
`1935e591`	2026-06-23 10:45	61	24	true	
`3f249acb`	2026-06-23 11:45	121	48	true	
`716bf939`	2026-06-23 17:14	4	1	true	Short interruption, same day
`bb53b1df`	2026-06-28 08:17	1442	576	true	24-hour cut (28 to 29 Jun)
`0ae9f667`	2026-07-16 10:14	38	0	false	Interruption during maintenance. The agent reported no stored payloads, so this 38-minute gap is **not covered**. It is shown, not hidden.
`e98264bb`	2026-09-29 08:03	31	12	true	Earlier or aborted runs on 29 Sep, before the back-to-back cuts in section 2
`0e28795c`	2026-09-29 08:33	136	54	true	
`c5e65f6e`	2026-09-29 10:50	31	12	true	
`463bf63b`	2026-09-29 11:20	61	24	true	
`71774bcb`	2026-09-29 12:21	17	6	true	

7. Limits

- Test environment only; no customer site. One agent, one site, monitoring its own host.
- Longest tested outage: 1,442 minutes. The agent's local storage has no fixed limit; the practical limit is disk space.
- During an outage the agent samples less often than when connected (about every 2.5 minutes).
- The backend does not yet keep the content of stored payloads with their collection time, so measurements during an outage cannot be reconstructed from this record.
- Times come from the backend server clock (UTC). Clock synchronization of the test hosts (ISO 27001 A.8.17) is not covered here.
- The record shows how the monitoring link behaved. It does not show what happened inside applications or databases at the site during the outage.