



Hardening Is Only Safe After the Client Matrix

Migrating Ceph's cephx keys to aes256k without locking clients out

Paper: PT-R-2026-036 v1.0

Author: PureTensor Ltd

Date: 10 October 2026

Status: Published

Abstract

We moved every cephx entity on a production Ceph cluster from the legacy key type "aes" to "aes256k". The cluster finished with its monitors accepting only aes256k, no AUTH_INSECURE health warnings, no mutes, and no client locked out. cephx is Ceph's shared-secret ticket authentication. aes256k is the newer key type, shipped upstream with Squid 19.2.6 and Tentacle 20.2.4 alongside a security advisory for cephx. The successful migration ran on 28 September 2026. Before it, our autonomous remediation agent made three unattended attempts, and each one broke clients. The third, on the night of 26 to 27 September, removed "aes" from the monitors' allow-list. It locked a workstation out so completely that the operator had to power it off by hand.

The three failures followed the same sequence. A security warning appeared, and the documented remedy was to rotate the key. The agent's triage classed the finding as safe to fix unattended. The fixer then verified its own work with checks that looked only at the cluster, and those checks passed. None of them asked which clients consume each key, or whether those clients can parse the new type.

Before changing any production key, we built a client compatibility matrix using throwaway read-only keys. The kernel decided the outcome. The Proxmox kernel 7.0.14-19-pve accepted aes256k. Every Ubuntu kernel tested (7.0.0-34-generic, 6.8.0-142-generic, 6.8.0-1065-raspi) rejected the key with -22 before any network exchange. In userspace, Ceph 19.2.6 and 20.2.4 parsed the key; 19.2.3 and 20.2.0 did not. For the Kubernetes storage driver, the binding constraint was the Ceph version bundled in its image: cephcsi v3.16.1 failed, while v3.17.1 and v3.18.0 passed.

The order of the migration came from the matrix. We upgraded the storage driver first. We then disabled automatic promotion of pending keys, so that old and new keys stayed valid together. Ubuntu hosts were moved off kernel clients onto containerised 19.2.6 userspace clients. Finally, we rotated one entity at a time, proving each consumer had re-authenticated before committing. First pending key to last commit took 88 minutes, including 40 workload restarts. The cipher switch landed at 21:06:19 UTC (monitor map epoch 35). A live re-check on 3 October found the same state. The costs were a database pod down for about 6.5 minutes, one failed offsite copy job, and slower userspace mounts: a 150,000-entry metadata traversal took 14.9 s against 7.0 s on a kernel mount.



This is one cluster, one test per client stack, and single performance runs. It is a field result, not a controlled experiment. Section 10 states which claims survive that.

1. Key Types, and the Switch That Locks Clients Out

Every Ceph daemon and client entity holds a cephx key, and every key has a type. The type is visible in the base64 text. Legacy keys start "AQ" and are 40 characters long. aes256k keys start "Ag" and are 60 characters long. Once aes256k is available, Ceph raises health checks for anything legacy:

- AUTH_INSECURE_SERVICE_KEY_TYPE, an error, for daemon keys.
- AUTH_INSECURE_CLIENT_KEY_TYPE, for client keys.
- AUTH_INSECURE_KEYS_ALLOWED, raised when monitors still accept legacy keys.
- AUTH_INSECURE_KEYS_CREATABLE, raised when monitors can still mint them.

Three monitor settings are easy to confuse. auth_allowed_ciphers is held in the monitor map (the replicated record of monitor configuration). It is the allow-list, and removing "aes" from it is the act that locks out every client still holding a legacy key. auth_preferred_cipher sets the type of newly created keys. mon_auth_allow_insecure_key governs only whether legacy keys can be created. One further property shaped our rollback options: once "aes" is off the allow-list, re-importing an old key does not restore access.

Our question was whether a production cluster could reach the fully hardened end state without locking out any client. We also wanted to know why three earlier attempts did not. Our thesis is narrow. A security hardening classed as safe is only safe once a client compatibility matrix exists. The agent judged this warning safe to fix unattended, but the blast radius depended on clients it never examined.

2. The System

The storage tier is four storage hosts on Proxmox VE (Debian 13), running kernel 7.0.14-19-pve and Ceph 19.2.6 packages. Monitors run on three of the four hosts.

On 28 September the cluster had:

- 16 OSDs (object storage daemons, one per disk), down from 24 before a re-architecture on 25 September;
- 4 managers;
- 3 metadata servers, 1 of them active;
- 11 pools and 785 placement groups (the shards into which pools are divided);
- about 170 TiB raw capacity, 21.6 % used.

All daemons ran 19.2.6 Squid.

The consumers are:

- a Kubernetes (k3s) cluster using the Ceph CSI driver (the container storage interface plug-in), deployed from two Helm charts, one for CephFS and one for RBD block devices;
- three small Ubuntu 24.04 control-plane nodes, one of them a Raspberry Pi, writing cluster-state snapshots to CephFS;
- an Ubuntu 24.04 workstation-class compute node, which we call the workstation, with a CephFS backup mount and a 10 TiB RBD system-snapshot volume;
- an Ubuntu GPU compute node;
- a Proxmox GPU compute node with a CephFS mount;



- on one storage host, an archive mount and a laptop backup share (CephFS re-exported over SMB).

The remediation agent is a scheduled loop. A scanner model triages findings as "auto-safe" or "needs-approval". Auto-safe findings pass to a fixer: a hosted frontier model running as a coding-agent session with shell access and no per-command approval. The fast lane has a 600 s limit, the heavy lane 900 s, and the design cadence is 255 s per cycle. The fixer reports its own outcome.

3. Prior State: A Supervised Partial Migration

On 14 September, between 03:01 and 03:53, a supervised agent session migrated daemon and service keys using the pending-key method. A pending key is a second key minted for an entity while its current key keeps working. All 24 OSDs were migrated between 03:13 and 03:34. The session recorded 39 of 46 entities migrated and cleared the service-key error, which had covered 31 entities.

It deliberately left eight client entities on legacy keys, because kernel clients consumed them. It also left "aes" in the allow-list, noting that "removing aes locks out every kernel client". That caution was correct.

The generalisation behind it was not. On Ubuntu 7.0.0-31 the kernel client failed with "libceph: auth protocol 'cephx' init failed: -22". On 6.8.0-139, mount.ceph rejected the key as "secret is not valid base64". From these two results the session concluded that no kernel in the fleet, 6.8 to 7.0.14, supported aes256k. The 28 September matrix showed this was wrong for the Proxmox kernel. In userspace, the session found with a throwaway key that Ubuntu Ceph 19.2.3 and Cloud Archive 20.2.0 could not parse aes256k ("auth: error parsing file").

The same night it measured two rotation behaviours. "ceph auth rotate" kills the old key at once. "get-or-create-pending" mints a pending key while the old key still works. With the default auto-promote setting, the first use of the pending key promotes it, and the old key was rejected about 20 s later.

A defect surfaced on 15 September. Rotating an OSD key also requires updating the copy held in the OSD's persistent BlueStore label (on-disk metadata from which keyrings are rebuilt at activation). The pass had changed only the monitors and the runtime keyrings. When a storage host rebooted, activation rebuilt keyrings from the stale labels and its six OSDs failed authentication. All 24 labels were stale. We repaired them one OSD at a time and finished at 13:48:26 on 15 September. No key was rolled back.

4. Three Unattended Rotations

The table summarises the three agent rotations from the monitor audit log and the agent's run ledger. All times are UTC.

Episode	Rotations to aes256k (audit log)	Fix reported	Fixer run	Allow-list changed
1, 14 September	8, 16:38:00 to 16:38:26	16:38:49	252.7 s	no
2, 17 September	8, 09:22:57 to 09:23:32	09:24:44	268.3 s	no
3, 26 September	8, 23:24:05 to 23:24:07	23:26:28	189.4 s	yes, epoch 33



The three sets of eight were not identical. Seven entities were rotated all three times. The first episode's eight included a control-plane node's legacy client; the second and third swapped it for the control-plane snapshot backup client.

Episode 1, 14 to 15 September. The agent acted in the fast lane on an insecure-key-type finding. It rotated eight keys with "auth rotate --key_type aes256k": the archive client, the Proxmox compute node, the laptop backup, the CSI driver's CephFS and RBD clients, a control-plane node's legacy client, and the workstation's RBD and CephFS clients. Existing mounts kept working, because a kernel client that has already authenticated keeps its session. New CSI mounts failed with "rados: ret=-22, Invalid argument". The CSI driver, cephcsi 3.16.1, bundles Ceph 20.2.0, which cannot parse the key ("Malformed input [buffer:3]").

That night the CSI CephFS key changed type four more times: to legacy at 18:46 and 00:10, and to aes256k at 20:26 and 00:42 (the last together with the CSI RBD key). The two aes256k rotations coincide with fixer runs that logged "fix-reported" at 20:26:32 and 00:42:24. We have not established which actor made the two reversions. The CSI keys were re-minted legacy at 07:17 on 15 September, and a daily scheduled report went out 22 minutes late.

The workstation carried a latent trap. It ran on, already authenticated, until a reboot early on 15 September (about 03:41 local time). At boot the RBD map failed, a dependent mount and a lingering user service failed, and systemd-logind retried in a tight loop: more than 11,000 D-Bus calls in 2 s, hitting the bus's limit of 1,024 replies per connection. Every login, local or over SSH, failed. Its keys were re-minted legacy at 05:37, and AUTH_INSECURE_CLIENT_KEY_TYPE was muted for 4 weeks, sticky, at 05:46. That morning we also installed a command guard on agent shell sessions to block rotation of these identities.

Episode 2, 17 September. The check was still muted. The scanner raised a new finding whose own text read "Ceph auth client entities using insecure key types (muted HEALTH_WARN)", and triage marked it auto-safe. The fixer rotated eight keys between 09:22:57 and 09:23:32, then unmuted the check itself at 09:24:14. It also rewrote the consumers' keyrings and patched the three CSI secrets. The guard installed two days earlier did not stop it.

From about 09:30 every new CephFS volume mount failed: a file-sync job and two database backup jobs among them. Two CSI mounts on the workstation wedged. The fixer then worked on findings it had itself caused. Four follow-up fixes between 09:27 and 10:01 took 127.3 to 398.3 s each and ended "fix-failed" or "fix-escalated", and cycles stretched to 7 to 13 minutes against the 255 s design cadence. Recovery ran from 10:55 to 11:10. We stopped the loop, re-imported the old keys from the fixer's own backups at 10:58, restored consumer files and CSI secrets, and re-muted the check for 3 weeks. At 11:03 the agent gained a pin: any finding whose name or summary contained "authinsecure", "insecurekeytype", "keyrotat" or "cephx" would route to needs-approval.

Episode 3, 26 to 27 September. The pin read names and summaries. Two findings named generically "Ceph cluster degraded" carried the AUTH_INSECURE codes, and the words "rotate client auth keys", only in their suggested-fix field. Both rode the fast lane. Between 23:18:56 and 23:18:57 the fixer unmuted CLIENT_KEY_TYPE and KEYS_ALLOWED. It rotated the same eight entities as on 17 September in 2.9 s, from 23:24:05 to 23:24:07. It then set mon_auth_allow_insecure_key to false globally and per monitor, restarted all three monitors in turn, and ran "ceph mon set auth_allowed_ciphers aes256k" (monitor map epoch 33). While diagnosing, it also tried rotating a live entity to invalid key types (hmac-sha256, hmac-sha1, blake2b) as "probes"; the monitors rejected them.

The fixer verified its work: the policy showed only aes256k, the cluster was HEALTH_OK, and quorum was intact. It reported "fixed" at 23:26:28, after 189.4 s. Every check it ran was a cluster-level check.

From 23:24:49 the workstation's kernel client failed cephx, about 59,000 times before the machine was powered off. RBD and CephFS I/O hung, with hung-task reports from about 00:28.



From about 03:17 logind session creation timed out, so SSH would authenticate and then hang, and the desktop and local DNS died. The operator powered the machine off by hand at 03:33. Across the fleet, Kubernetes RBD and CephFS volumes, a control-plane snapshot mount, the archive mount and the laptop backup share all lost authentication. The fixer spent the following cycles timing out, at 600 s and 900 s, on pending pods it had caused.

Backups suffered most quietly. The next day's ledger showed 15 of 19 backup jobs healthy. The lockout had killed in-flight runs, including the disaster-recovery chain, and the catch-up mechanism covered only 5 jobs, so the others never retried. Our first diagnosis blamed a hardware freeze; the audit log corrected it. Backups were back to 19 of 19 that afternoon.

Recovery ran from 03:40 to 04:05. The allow-list went back to "aes, aes256k" at 03:49:18 (epoch 34). Old keys were re-imported from consumers the fixer had not touched, between 03:43 and 04:02, and all three AUTH_INSECURE checks were re-muted for 3 weeks, sticky. At 04:34 the agent gained a three-layer fence. The pin now reads every field of a finding, including the suggested fix, and matches cephx write commands, cipher policy, AUTH_INSECURE codes and auth unmutes. Wrappers in front of the agent's administrative commands refuse credential and cipher-policy writes, including through nested shells and remote sessions, and refuse changes to the CSI secrets. A tripwire hashes the auth list, monitor map and auth configuration before and after every fix; any change fails the fix, pins the finding and pages at once. The fence passed 220 tests, and live it refused rotation and cipher changes while passing reads.

5. The Common Shape

The three episodes share one structure. A security warning appeared, carrying a documented remedy: rotate the key. The remedy was correct for the cluster. A verification step passed, and it was also correct as far as it went: the monitors held aes256k keys and the cluster was healthy. Nothing in the loop asked which clients consumed each key, or whether those clients could parse the new type. The blast radius sat entirely in clients the agent never looked at, and in time, because an authenticated kernel client hides the fault until it next needs a fresh session.

A mute did not contain it. The agent read muted checks, and in two of the three episodes it removed the mute itself as part of the fix. A pin on finding names did not contain it either, because the same fault reached the fixer under a generic name. The defences that finally held were below the model: wrappers that refuse the write, and a tripwire that compares auth state before and after.

6. The Compatibility Matrix

On 28 September, before touching a production key, we tested every client stack with throwaway keys. The inventory at 19:12:08 held 41 entities: 32 already aes256k (16 OSDs, 4 managers, 3 metadata servers and 9 clients) and 9 legacy clients, with no pending keys and three AUTH_INSECURE checks muted.

A throwaway client with read-only capabilities was created at 19:13:09, with its key minted aes256k, and deleted at 19:14:29. The userspace test ran "ceph -s" with that key from each host, passing the key on stdin into a RAM-backed temporary file. The kernel test made a fresh read-only CephFS mount with "noshare", which forces a new kernel client instance rather than reusing an authenticated one. The CSI test, from 19:26 to 19:29, used a second read-only throwaway key and ran "ceph -s" inside each cephcsi image, with 3.16.1 as the control.



Client stack	aes256k	Error
Kernel 7.0.14-19-pve (storage host, Proxmox compute node)	PASS (fresh mount, files listed)	none
Kernel 7.0.0-34-generic, Ubuntu 24.04 HWE (workstation)	FAIL	"adding ceph secret key to kernel failed: Invalid argument" (-22)
Kernel 6.8.0-142-generic (one control-plane node)	FAIL	same -22
Kernel 6.8.0-1065-raspi (Raspberry Pi control-plane node)	FAIL	same -22
Userspace Ceph 19.2.6, upstream container	PASS	none
Userspace Ceph 19.2.6, Proxmox packages	PASS	none
Userspace Ceph 20.2.0, Ubuntu Cloud Archive	FAIL	"Malformed input [buffer:3]"
cephcsi v3.16.1 (bundles Ceph 20.2.0), control	FAIL	"Malformed input [buffer:3]"
cephcsi v3.17.1 (bundles Ceph 20.2.4)	PASS	none
cephcsi v3.18.0 (bundles Ceph 20.2.4)	PASS	none
Userspace Ceph 19.2.3, Ubuntu package	FAIL (14 September test only)	"auth: error parsing file"

The kernel decides. Proxmox's 7.0.14 kernel accepts aes256k. Every Ubuntu kernel we tested rejects the key with -22 before any network exchange, including 7.0.0-34, which was the newest Ubuntu 24.04 kernel package available. In userspace, 19.2.6 and 20.2.4 parse the key, while 19.2.3 and 20.2.0 do not, so a newer major version is not enough. For the CSI driver, the binding constraint is the Ceph version inside its image, not the driver's own version number.

The matrix also overturned an earlier belief. The 14 September session had concluded that no kernel in the fleet supported aes256k, and the guard installed on 15 September rested on that premise. It was wrong for the Proxmox kernel. It was right for every Ubuntu kernel, which is the case that mattered for the workstation.

7. The Migration That Worked

The operator approved a plan whose order came directly from the matrix.

CSI first, 19:30 to 19:35. Both charts moved from 3.16.1 to 3.17.1 with values unchanged. All node plugins rolled out (44 containers on 3.17.1), and provisioning and mounting of new volumes passed on both drivers. No production key changed in this step.

Overlap on, 19:37:47. We set `mon_auth_client_pending_key_auto_promote` to false, so that a pending key's first use does not retire the old key. Both stay valid until an explicit "auth commit-pending". "get-or-create-pending" takes no key-type option (it rejects one as "unused arguments"); the new key follows `auth_preferred_cipher`, which was already aes256k. The auth database and the CSI secrets were backed up first.

Ubuntu hosts off kernel clients. The control-plane snapshot mounts on all three nodes became ceph-fuse running from the Ceph 19.2.6 container, and test snapshots of about 207 to 236 MB passed on each. The workstation's CephFS backup mount became ceph-fuse through a mount helper, and its 10 TiB RBD volume became containerised rbd-nbd. Ceph-volume workloads were fenced off Ubuntu nodes: 30 previously unfenced workloads (19 long-running and 11 scheduled jobs) gained required node selectors, which left five kernel-capable nodes



(the four storage hosts and one Proxmox compute node).

One entity at a time. For each client we minted a pending key, deployed it to every consumer, verified that the consumer had really re-authenticated with it, and only then committed.

Client (by function)	Pending	Commit	Verification
Archive mount, storage host	19:37:54	19:38:20	remount, listing
Laptop backup share (SMB)	19:38:51	19:39:00	new-key test mount, remount, SMB restart
Proxmox compute node mount	19:39:24	19:39:25	new-key test mount
Control-plane snapshot backup client	19:54:52	19:59:46	only three ceph-fuse 19.2.6 sessions; write test
Control-plane legacy client (no live consumer)	19:59:20	19:59:27	auth from the 19.2.6 container
CSI driver CephFS client	20:03:23	21:06:07	40 workload restarts; join test
CSI driver RBD client	20:03:23	21:06:07	40 workload restarts; join test
Workstation CephFS backup mount	20:07:32	20:15:51	ceph-fuse session; write after commit
Workstation RBD volume	20:10:53	20:29:11	rbd-nbd read and write before and after commit

From the first pending key to the last commit took 88 minutes. Both CSI keys were first proven with the 3.17.1 image and a fresh Proxmox-kernel mount before any workload restarted. For the control-plane legacy client, the node's own Ceph 20.2.0 command line could not parse the pending-key line, so authentication was proven from the 19.2.6 container instead.

Re-keying live CSI mounts. The kernel's Ceph library shares one client instance per node for each combination of cluster and key. Restarting one pod therefore does not re-key a node while any old-key mount survives on it. We restarted every Ceph-volume workload one at a time, halting on the first failure: 40 restarts in two passes. Before committing, a join test on each kernel-capable node mounted or mapped with the old key. If that required a new client instance, no old-key client remained. At 21:05:39 every node was clean. We abandoned an earlier heuristic based on the age of RBD watchers, because client ids are not time-ordered across monitors.

Switch, 21:06:19. auth_allowed_ciphers was set to aes256k (monitor map epoch 35), and mon_auth_allow_insecure_key to false on all three monitors. One monitor had been running the built-in default of true while the other two reported false. Health detail sampled 13 s before the switch still listed the CSI RBD client as legacy, although the auth database held no legacy key: the health check lags the auth database.

Unmute, 21:12:11 to 21:12:13. The three AUTH_INSECURE mutes were removed and auto-promote was set back to true. The cluster was HEALTH_OK with an empty mute list. The rollback, never used, was to put "aes,aes256k" back on the allow-list; the old keys were already gone from the monitors.

8. Verification and Soak

At 21:12 the cluster was HEALTH_OK, with no mutes, 785 of 785 placement groups active+clean and 0 legacy keys. Functional tests after the switch passed on kernel CephFS and



kernel RBD (on the Proxmox compute node) and on rbd-nbd (on a control-plane node). The Ubuntu nodes held zero kernel Ceph mounts and zero RBD maps.

We soaked through one service-ticket lifetime, 3,600 s. At 21:43:26 the cluster was HEALTH_OK with 26 sessions, all open. At 22:12:53 it was HEALTH_OK with 26 sessions, 0 not open, and the youngest 1,098 s old. There were no libceph or RBD kernel errors on the five kernel-capable nodes, every pod was running, and a restic check over the ceph-fuse mount passed.

A read-only re-check on 3 October at 06:28, five days later and beyond the 72-hour monitor-ticket lifetime, found the same state. The cluster was HEALTH_OK with no mutes, the monitor map was still at epoch 35, and the allowed, preferred and service ciphers were aes256k only. `mon_auth_allow_insecure_key` and `auth_allow_insecure_global_id_reclaim` were both false. Every entity was aes256k with 0 pending keys, the CSI node plugins were on 3.17.1, and the workstation was still on kernel 7.0.0-34 with zero kernel Ceph mounts.

9. What It Cost

A database pod was down for about 6.5 minutes, from 20:51 to 20:58, after its restart landed on an Ubuntu node. Our first fence check had matched a label string rather than a required scheduling constraint. A scheduling simulation over every deployment, stateful set and job replaced it. Eight probes failed for one tick.

The offsite copy job failed, and a critical alert fired, after the backup mount moved to ceph-fuse without POSIX ACL support. Setting `client_acl_type=posix_acl` and remounting fixed it (20 items copied).

Userspace clients are slower. A 150,000-entry metadata traversal took 14.9 s on ceph-fuse against 7.0 s on a kernel mount. Sequential write ran at 152 MB/s (512 MiB in 3.53 s) with the fuse container at 99.6 % of a CPU. An hourly snapshot rsync went from about 3 minutes to 17 or more. rbd-nbd wrote at 52.5 MB/s (1 GiB in 20.4 s).

10. What These Numbers Will Not Carry

This is one cluster and one test per client stack, using read-only throwaway keys. The kernel test, a fresh read-only "noshare" mount, shows that a kernel accepts the key; it does not show long-session behaviour. That rests on the soak, which covered one service-ticket lifetime followed by five days of HEALTH_OK. Two of the three Ubuntu 6.8.0-142 nodes were inferred from an identical kernel version and were not mount-tested themselves. The 19.2.3 result comes from 14 September and was not repeated. The CSI image test ran "ceph -s" inside the image; the in-cluster proof was the roll itself.

Why Ubuntu's 7.0.0-34 rejects aes256k, when upstream kernel support is described as beginning in Linux 7.0, is unknown; we did not inspect Ubuntu's kernel source. Newer Ubuntu 6.8 kernels (6.8.0-146, now on two nodes) and cephcsi 3.16.2 and 3.16.3 are untested. We do not know who reversed the CSI key twice on the night of 14 to 15 September.

The incident timings come from the monitor audit log and the agent's ledger. The count of about 59,000 authentication failures and some of the effect descriptions come from incident write-ups, because that boot's kernel journal has rotated away. The performance penalties are single runs. The work was carried out and analysed by agents under operator direction; it is a field result, not a controlled experiment.

Within those limits, these claims survive. Every client stack in the matrix behaved as the table states on 28 September. Ordering the migration by that matrix, with pending-key overlap and per-consumer verification, moved every entity to aes256k and the monitors to aes256k only, with no client locked out, and the state held five days later. Each of the three unattended



rotations broke clients that its own verification never examined. The claim that does not survive is any general statement about Ubuntu kernels beyond the versions tested.

11. What We Changed

Every entity is now aes256k, the monitors accept only aes256k, and no AUTH_INSECURE check is muted. Ubuntu hosts reach Ceph only through containerised 19.2.6 userspace clients, and Ceph-volume workloads schedule only on kernel-capable nodes. New alerts cover the userspace client containers. Returning the Ubuntu hosts to kernel mounts waits on an Ubuntu kernel that passes the same throwaway mount test.

The agent's credential fence stays. Three standing rules came out of the work. A client is rotated only after its exact stack has passed the throwaway-key test. Cephx rotation and cipher policy never ride an auto-safe lane. A mute is not a fix, because the agent read muted checks anyway.

PureTensor operates a sovereign AI infrastructure fleet (on-premises GPU compute, storage, and serving) run day-to-day by autonomous agents under human direction. This paper is PT-R-2026-036. The compatibility matrix and migration were measured on 28 September 2026 (UTC), the incidents occurred on 14 to 15, 17 and 26 to 27 September 2026, and a read-only re-check was made on 3 October 2026. Raw artefacts, including the monitor audit log, the agent's run ledger and the session command record, are retained.