BreachSpider

Technical white paper: exact-version vulnerability, fix and lifecycle intelligence for industrial and IT devices. Second edition, 8 November 2026.

1. Summary
2. Data at a glance
3. The problem
4. Architecture
5. How the engine decides
6. Evidence and sources
7. The web app
8. Documents vault and post-quantum seals
9. Integrations
10. Security model
11. How data quality is kept
12. Measured performance
13. Limits and open work

Every number in this paper was measured or read from the system; database counts were read on 8 November 2026. Product names belong to their owners. CITED Relevance LLC.

1. Summary

BreachSpider tells industrial (OT) and IT organizations which known vulnerabilities apply to the exact devices and versions they run, which single update clears the most of them, and whether each device is still supported by its vendor and on current firmware. Every fact links to the source that states it, with the date it was read.

The decision path is deterministic: the same device and version always give the same answer and the same result hash, whether the question comes through the web app, the API, the Python SDK, the partner widget or an assistant connected through the Model Context Protocol. Language models are used only to explain and draft, never to decide, and every model output is checked against the facts it cites before anyone sees it.

This edition adds the current size of the data, the applicability and fix classes the engine reports, the asset operations modules (labels, maintenance, documents and backups), the documents vault with hybrid post-quantum seals, and an updated account of the security model and of what is still open.

2. Data at a glance

Counts read from the production database on 8 November 2026 (read-only queries):

Measure	Count
CVE records held (rejected identifiers excluded)	401,879 (420,989 including rejected)
CVEs linked to 42 industrial automation vendors	21,401; 90% of them carry version-bounded data
Known-exploited CVEs (catalog listings)	1,769
CVEs with an exploit probability score	401,888
Catalog products	169,612
Vendors in the catalog	36,712
CVE to product links	1,595,047
Affected-version rows	3,457,347
Of which from numbering authority, government or vendor advisories, with provenance	about 229,600 (numbering authorities 135,286; their enrichment 47,884; government ICS advisories 39,010; vendor CSAF about 7,400)
	235,387

Microsoft fix records (CVE, product, fixed build, update)	
Products with a vendor lifecycle fact	1,957, plus 43,007 part numbers and models, from 96 vendors (20 from the vendor's own lifecycle publications)
Lifecycle facts without a source link	0
Firmware currency	6 vendors, 1,329 firmware lines, 8,481 releases

Newest CVE record published 7 November 2026 22:16 UTC; known-exploited catalog read 8 November 03:30 UTC; exploit database read 8 November 02:06 UTC. The public exploit repository index was last pulled 23 May 2026 (see limits).

3. The problem

An industrial site runs controllers, protection relays, gateways, switches, HMIs and Windows hosts from many vendors, often on firmware that is years old. Vulnerability records name products loosely, use different version schemes, and frequently say "all versions" when the vendor's own advisory gives a precise fixed release. A tool that matches by product name alone floods a site with findings that do not apply and hides the few that do.

Three questions matter to the people running the site: which CVEs truly affect this device at this version; what is the one update that clears the most risk, known-exploited first; and is the device still supported, so the answer is a patch rather than a replacement. BreachSpider answers all three for every device, and says so plainly when it cannot.

Zero CVEs is never presented as no risk: an empty result can mean the product was not identified, no source publishes version data for it, or the vendor stopped supporting it and stopped publishing advisories. The engine says which.

4. Architecture

- **One engine.** A single correlation path serves the API, the partner widget, the nightly change feed, the web app's background checks and both MCP servers, on a read-only database connection.
- **Stored mode.** Each device's last check (result hash, status hash, status facts, CVE identifiers and summary) is stored; differences between checks become change events; every check is logged. Every device is checked again nightly after the data pipeline, and on demand.
- **Background jobs.** Durable jobs claimed with row locks that skip locked rows, one job per organization at a time, batches of 50, heartbeat, cancel, retry with back-off and recovery of stale jobs. Workers end their database transactions every pass.
- **Daily posture snapshots** per organization and site, computed with the Overview's own predicates, feed Progress and its reports.

- **Data pipeline.** Nightly CVE pull with a completeness check against the published record list, exploit probability, exploit database and national CERT feeds; daily known-exploited catalog; weekly numbering-authority ranges and government ICS advisories; Microsoft monthly and daily; weekly vendor lifecycle connectors.
- **Releases.** Every change is staged, baselined and pre-checked (unit tests, public figures, benchmarks, regression drift against signed workbooks, tenant isolation gate, integrity verify) before the API restarts, then post-checked against the running service. A failed pre-check restores the files and leaves the service untouched. Data changes pass an integrity gate: applied in one transaction with critical invariants, committed only on a pass.

5. How the engine decides

Resolution: from inventory text to an exact product

Customers describe devices the way their inventories do: "AB ControlLogix 5580 fw 37.011", "Siemens S7-1516-3 PN/DP", "Modicon M340 BMXP342020 2.20". The resolution layer turns that text into ranked catalog candidates with a numeric confidence, a band (high 80 or more, medium 60 to 79, low under 60) and a decision: resolved, ambiguous (candidates within 12 points; no findings until a person chooses) or unresolved. It uses vendor aliases and canonical vendor roll-up, exact and normalized names, curated order-code patterns, separately supplied part numbers and fuzzy matching, and adopts a retry only when it resolves cleanly (precision over coverage). A family without a model is answered at family level, held below high confidence and flagged for review.

Exact-version matching against stated ranges

A device's CVEs are decided by its version against each record's affected range: start and end, inclusive or not, in the vendor's own numbering. Match tiers are exact (a listed version), range, or product-wide (no version bound, or no version sent). Vendor schemes are handled explicitly: release prefixes such as V, R and SV; Schneider Electric's two-digit minor versions (3.4 is read as 3.40); suffixes compared approximately and flagged; date-style bounds never compared with ordinary versions. Where a vendor's advisory states a precise fixed release, that statement is stored with its provenance and outranks a looser "all versions" listing.

Applicability and fix classes

Each CVE on a device carries an applicability class: confirmed affected (the version was compared and the model is known), possibly affected (not compared, with the reason) or likely not affected (a version-less record whose vendor-stated fix is at or below the installed version, an inference labelled as such). Each fix carries a class: vendor confirmed, range derived (always labelled, to be confirmed in the advisory), workaround only, no fix published, not needed, or unknown. Fixes are chosen in a fixed order: the vendor's stated version for this product, a vendor no-fix statement (quoted), a stored version, a version derived from the range end, a version-less vendor fix, then no fix known.

Windows: the vendor's own fixed build decides

For Windows hosts, Microsoft's update data decides: the host's build and installed updates are compared with the fixed build Microsoft names for each CVE on that build line; older Windows is decided by installed updates, rollups that replace them and extended security update enrollment. Disagreements are needs review, never guessed. The update that closes the most CVEs is computed over every update, not only the latest.

Determinism and hashes

Every device result carries a result hash (identity, version, each CVE with its tier and a digest of its range source and vendor guidance) and a status hash (known-exploited, lifecycle and firmware facts), both SHA-256 over canonical JSON. Equal input gives equal hashes across every channel. Measured at 10,000 devices: 50 of 50 randomly chosen stored results were identical to a fresh engine run. A change check recomputes only the hashes, at a tenth of the cost.

6. Evidence and sources

Sources, each with its last successful read stored and shown: CVE records and the numbering authorities' own records with their enrichment; vendor security advisories in the CSAF format and vendor advisory pages; government ICS advisories (CSAF); Microsoft's update data; the known-exploited vulnerabilities catalog (listing date, due date, ransomware use); the exploit database; a public exploit repository index; exploit probability scores; national CERT feeds; vendor lifecycle and firmware pages.

Every fact shown links to the document that states it and says what its date means (read, as of, retrieved, published or modified). Sources are tiered: the vendor's own statements (tier A), government restatements of vendor statements (tier B), and third-party listings (tier C), which are always labelled as such, never presented as the vendor's word, and always yield to a vendor fact of the same kind.

Lifecycle and firmware currency: vendor statements only, linked

Support status comes from vendor statements: lifecycle pages, end-of-sale and end-of-life notices, discontinued product lists, and end-of-life statements inside vendors' own advisories. Absence of a notice is never read as support. Where a vendor's website terms ban automated collection, statements are transcribed by hand with exact quotes and the page link. Where nothing is held, the answer links the vendor's own lifecycle page or the exact product page for the part number. Firmware currency compares the installed version with the latest release the vendor lists and quotes the security lines of later release notes; lifecycle and firmware are kept separate because they lead to different decisions (a replacement budget versus a patching task).

Collection rules

Every connector reads robots.txt and stores it with the terms page and a checksum before collecting; a refusal (HTTP 403 or 429) stops the run at once and is never worked around;

nothing behind a login is read; only facts with a link are kept, never republished vendor material.

7. The web app

The web app is organized around decisions, not lists, in five groups:

- **Overview, Progress, Fix first.** Exposure with every number opening its devices (same predicate as the list, tested equal); a start-here action; trend from recorded daily snapshots with gaps never estimated. Progress credits a fix only when a finding cleared in the same check as a version change on that device; data and match changes are shown apart and never credited. Fix first groups devices by the single action that clears the most, ordered by overdue known-exploited CVEs, then known-exploited count, severity and findings cleared; a target version is never invented.
- **Inventory: Assets, Sites, Changes.** Search, views, bulk actions with preview fingerprint, stale-preview refusal and undo; the device panel with fix plan, findings with sources, lifecycle, firmware, history; acknowledgements that only label and never change counts; recorded actions where "patched" is never a checkbox; change digests that group without deleting, with newly known-exploited CVEs always first and alone.
- **Intelligence: Lifecycle, Firmware, Windows, Research.** Vendor-stated support with sources; firmware against the latest listed release; Windows hosts by build and updates; CVE search and a watchlist.
- **Operations: QR and identifiers, Maintenance, Documents and backups, Reports.** QR labels hold only a random code that opens a device for members of the organization and nothing for anyone else, with print tracking, revoke and replace, camera scan and identifier import; maintenance work records with work orders, states and history, recurrence, links to Fix first and to configuration backups, where a completed firmware update triggers a re-check and findings change only when that check confirms the version; the documents vault and encrypted workspace backups (next section); reports of seven types saved as compressed canonical snapshots with a SHA-256 re-verified on every open, kept three years, with optional framework labels that are explicitly not a certification or an audit opinion.
- **Administration: Integrations, Clients, Settings.** API keys shown once and stored hashed, connected apps with revoke, signed webhooks with delivery history, the partner widget; MSSP client portfolio and switcher; two-factor sign-in, sessions, team roles, API usage, audit log and retention read from the purge job's own rules.

Everything a person changes is reversible: device edits and removals, sites, deleted reports, bulk edits, actions, imports and acknowledgements each have an undo, refused when the item changed again since.

The assistant: explains and drafts, never decides

The assistant (Claude, by Anthropic) drafts report narratives, explains a finding for an operator or a manager, summarizes the change digest, answers questions about the organization's own environment, drafts change tickets and suggests products for import rows the catalog could not

identify. It receives only the facts of the record in question. A deterministic guardrail refuses any line that cites no fact, contains a number, CVE identifier or version not present in the facts it cites, or makes a safety, compliance or speculative claim; refused drafts are repaired at most twice and otherwise not shown. Import suggestions are re-run through the resolver and accepted by a person, and the catalog's own name is written, never the model's text. Every call is logged, and the assistant is off unless the organization turns it on.

8. Documents vault and post-quantum seals

Encryption in the browser

Configuration files and documents are encrypted in the browser before upload, with AES-256-GCM in 4 MiB chunks whose associated data binds organization, device, version, kind, chunk index and last-chunk flag. Each version has a fresh 256-bit data key wrapped by a 256-bit organization vault key, which is stored only wrapped by a key derived with Argon2id (64 MiB, 3 passes) from the vault passphrase and, separately, from a printed 150-bit recovery key. File name, type, notes, size and the original file's SHA-256 are encrypted with the data key. The ciphertext goes from the browser straight to a private object storage bucket through a presigned upload for one never-reused object name. BreachSpider holds no key and cannot read the files; if its database, application server and storage were all exposed together, an attacker would learn sizes, counts, device links and times only.

Sealing by a separate service

A signing service running as its own operating system user, with its own keys, its own read-only storage credential and its own append-only ledger (updates and deletes refused), reads the stored object itself, measures its SHA-256 and length, and signs a canonical manifest (organization, device, version, kind, object name, ciphertext digest and length, algorithms, uploader, time, sequence number and the previous seal's hash). It never signs a digest it is told. Seals form a hash chain per organization, and once a day the service signs every organization's chain head and writes this anchor to a separate bucket under 400-day governance retention.

Hybrid signatures

Since 7 November 2026 every seal is signed with Ed25519 and with ML-DSA-65 (FIPS 204, module-lattice based) over the same bytes, and verification requires both. Ed25519 rests on the elliptic curve discrete logarithm problem, which Shor's algorithm would solve on a large fault-tolerant quantum computer; ML-DSA rests on module lattice problems for which no efficient quantum or classical algorithm is known. A forger would have to break both. Earlier seals carry Ed25519 only, keep verifying and are labelled classical; the daily anchor carries both signatures. Signing key identifiers on 8 November 2026: Ed25519 f5d5b03ec4359653, ML-DSA-65 39b2f241c9eef5e0.

Verification and change detection

Verification is answered from the signing service's ledger and the stored bytes, never from the application's records: both signatures, the chain, then a fresh read and digest of the stored object. Results are Verified, Mismatch or Unable to verify, with reasons; every sealed version is re-verified at least every 30 days, and before every download, which a mismatch blocks except for a logged administrator override. After download the browser decrypts and compares the recovered file's digest with the one sealed at upload. To answer "has the device's configuration changed since this backup", the browser hashes the current file locally and compares it with the sealed original digest; the server records only the outcome together with the backup's own integrity state at that moment, never the file, its name or a digest.

Workspace backups and service backups

Workspace backups (devices, sites, tags, identifiers, maintenance records) are encrypted in the browser with PBKDF2-SHA-256 (600,000 iterations) and AES-256-GCM and downloaded to the customer; BreachSpider keeps no copy. Restore brings back only what is missing, never overwrites, stays within the originating organization and has one undo. BreachSpider's own nightly database dump is encrypted as it is written and stored offsite in object storage, and a nightly encrypted, deduplicated whole-server backup is kept with daily, weekly and monthly retention; these are encrypted with keys BreachSpider holds, and contain vault material only as ciphertext.

9. Integrations

- **API**: device checks (vendor, product, version as written in the inventory, no CPE needed), stateless Windows host checks, change checks by result and status hash, a frozen v1 contract with a versioning policy, and machine readable retry guidance on every error (retryable, wait, action).
- **Asynchronous batches**: up to 5,000 devices or 1,000 Windows hosts per job, idempotent submission, encrypted inputs and results, 30-day expiry, metering reserved at submission and refunded for unprocessed work, signed callbacks delivered only to public addresses. Measured: 25 devices in 6.6 s, results identical to the synchronous call; a worker stopped mid-job recovered with every device metered exactly once.
- **Python SDK** (package breachspider 0.4.0): typed results, paging, retries that follow the retry guidance, batch helpers; the key is never printed or logged.
- **Model Context Protocol**: a local server package (breachspider-mcp 0.2.0, seven tools) that sends only vendor, product, version and a neutral identifier, and a hosted server at mcp.breachspider.com with OAuth sign-in, per-grant connection credentials limited to the tool routes, and refusal of host names, addresses, serial numbers, emails, user names and site names. Tested with Claude and ChatGPT; open per organization.
- **Partner widget**: an embeddable device check for partners' portals, with the same engine and hashes.

10. Security model

Every organization table is protected by Postgres row-level security; the API's database role cannot bypass it, sets the caller's organization on each pooled connection, fails closed when none is set and clears it on release. Managed service providers see their clients through a server-checked client context. The release process includes an isolation gate that calls every read route as one organization while another's rows exist: 167 routes, 328 calls, 0 leaks, 0 server errors, 39 foreign identifiers refused.

Sign-in uses single-use email links valid 60 minutes, optional enforced two-factor codes with single-use backup codes, and session cookies of up to 30 days that scripts cannot read, travel only over HTTPS and are same-site (Lax). API keys are shown once, stored as SHA-256 hashes, may expire, and are refused at once when expired or revoked; key, webhook, billing, account and vault management accept only a signed-in session. Per-key limits are 60 requests and 5,000 devices per minute by default. Devices never keep an IP address or serial number (a database trigger clears both on every write), and the API refuses device fields containing addresses, serial numbers, emails or user names without echoing them. Outbound requests (webhooks, logos, callbacks) go only to public addresses. The origin accepts web traffic only from the edge's published ranges.

The public demo gives each visitor a personal copy of a loaded workspace that expires after 24 hours; copies cannot send invitations, register webhooks, create keys or connect apps.

11. How data quality is kept

- **Integrity gate**: every data change is applied inside one transaction with critical invariants (golden device sets, Windows results reproduced exactly, provenance on every new range) and committed only on a pass.
- **Catalog audit**: properties tested per product with generated versions and spellings, including vendor ground truth (the vendor's own ranges decide affected or not affected at boundary versions); findings classified by severity, fixed by reviewed script rules where the pattern is general, and re-checked nightly. Product and vendor twins are caught nightly.
- **Completeness**: a nightly check compares the CVE records held with the published record list.
- **End-to-end tests**: every screen exercised in a browser on the production site at desktop and phone widths.

12. Measured performance

Measured on a synthetic organization of 10,000 devices (20 sites, 8,000 identifiable devices and 2,000 data-quality cases), server timings, network excluded:

Measure	Result	Target
Asset list search, sort, filter, page (p95)	worst 357 ms, most 10 to 220 ms	under 1 s
Device panel	5 to 18 ms	under 500 ms
Overview first view	615 ms server, 946 ms cold page load	under 2 s
Background check of all devices	21 min 21 s, 0 failed (2 min 8 s per 1,000)	1,000 under 5 min
Import of 10,000 rows	preview 0.47 s, resolve 9.0 s, commit 6.2 s	preview under 30 s
Hash parity, stored versus fresh	50 of 50 identical	equal

Signed-in app shell on a simulated mid-range phone: Lighthouse performance score 95 to 96, largest contentful paint 2.35 to 2.46 s. Public homepage after optimization: score 94 to 95, largest contentful paint about 2.0 s.

13. Limits and open work

- An empty result is limited to published CVEs in the sources held; vulnerabilities described only in advisory text, configuration weaknesses and network exposure are outside its scope.
- Lifecycle coverage depends on what vendors publish openly. Several large OT vendors keep product status behind customer portals or refuse automated requests; for those, BreachSpider shows their advisory statements, statements transcribed by hand and a link to the vendor's page. Firmware currency covers six vendors so far.
- Some identity cases still need work, for example switches from one vendor that run another vendor's operating system and must be checked against both vendors' ranges.
- The public exploit repository index has not been refreshed since 23 May 2026.
- The documents vault is enabled per organization and is in early use; a downloadable, offline-checkable verification receipt is not available yet.
- Single sign-on with customers' identity providers is not built.