ALIN LI

OPEN COMPUTING • IOT • ARCHITECTURE • TRAINING

WHITE PAPER

Security Data Without the Bill

An edge-classification approach to SIEM cost, evidence and control

For
CISOs, SOC managers, security architects and CIOs responsible for SIEM cost, detection coverage and audit evidence.

Published
October 2026

Version
1.0

ABSTRACT

In brief

> Most SIEM platforms are priced on what they ingest, so the bill follows volume rather than value. This paper describes an alternative: classify every event at the edge, before it reaches the indexer, using plain-text rules that a security team can read, test and reverse. Each event ends in one of three places: forwarded and indexed, archived in full at storage prices, or dropped with a recorded reason. A raw archive keeps every event regardless of verdict, so nothing is lost and every decision can be audited. We set out the architecture, the rule process, how to measure the effect honestly, and a practical route from proof of concept to production.

EXECUTIVE SUMMARY

What leaders need to know

- SIEM cost is driven largely by ingest volume, and a significant share of what most estates ingest is never searched, alerted on or read; the remedy is to decide where each event should live before it is billed, not after.
- Classifying events at the edge, on an open-source collector such as Vector, lets an organisation send the SIEM only what is worth indexing while keeping everything else, in full and searchable, at storage prices.
- Every event has one of three endings: forwarded (indexed and billed), archived (kept at full fidelity off the ingest line) or dropped with a reason, and an event can only be dropped after it has been named twice, first as low value and then as noise.
- A raw archive receives every event whatever its verdict, which makes each classification decision auditable and allows an archived window to be replayed into the SIEM when an investigation needs it.
- Rules are plain text, each with a case and a near miss, validated by the engine itself, replayed against last week's traffic before release, pushed as a versioned configuration pack and reversible with a full audit trail.
- On one production estate this approach sent 86.4% less volume to the SIEM, measured across 7,959,818 events (Verified; not a guarantee for every estate); the only figure that matters for your organisation is the one measured on your own traffic.
- The approach is delivered on open-source components, handed over to the organisation that runs it, and owned by that organisation: rules, configuration, documentation and data.

01

The problem: paying to index what nobody reads

Most security information and event management platforms are priced on ingestion: gigabytes per day, events per second, or a licence tier set by one or both. The commercial logic is simple, and it has a consequence that is easy to overlook. Every event is billed at the same rate whatever it is worth. A repeated firewall accept for the same internal flow, a health check that fires every few seconds, a debug line left switched on after a release and the authentication failure an analyst actually needs to see all cost the same to index.

No security team designs its estate to work this way. It happens gradually. When a SIEM is first connected, forwarding everything is the safe default, because nobody yet knows which sources will matter. Once a source is flowing, nobody is asked to justify it again. New applications arrive with verbose logging, infrastructure teams add forwarders for their own troubleshooting, and the volume grows faster than the value. A few years later the security budget is paying indexer prices for data whose principal function is to exist.

The usual responses are unsatisfactory. Raising the licence tier treats the symptom and repeats the problem at a higher price. Switching sources off removes visibility in ways that are rarely documented and often discovered only during an incident. Shortening retention in the SIEM reduces storage but not ingest, and can leave the organisation unable to answer questions an auditor or investigator will eventually ask. Each of these trades cost against evidence, and none of them gives the security team control over the trade.

The underlying error is to treat the question as keep or delete. It is not. The real question is where each event should live, and at what price. Some events belong in the hot path of the SIEM. Many belong somewhere cheaper, still complete and still searchable. A small, explicitly identified share is genuine noise. A design that makes this distinction deliberately, and records why, changes the economics without changing what the organisation can prove.

> **The bill follows the volume, not the value; the fix is to decide where each event lives before it is billed.**

02

Why filtering at the SIEM is the wrong place

Most SIEM platforms offer some form of filtering, and it is natural to start there. The difficulty is one of position. By the time an event reaches the SIEM's own filtering layer it has usually crossed the

point at which it is metered. Filtering after ingest can tidy an index, but it rarely reduces what the organisation pays for, and it does nothing for the network and forwarding capacity that carried the event there.

Filtering inside a SIEM also tends to be opaque. Exclusion logic lives in a vendor console, is often configured through screens rather than files, and is difficult to review, test or version in the way a security architect would expect of anything that governs evidence. When an auditor asks why a particular event is not in the SIEM, the answer may depend on the memory of whoever configured the exclusion.

Filtering at the SIEM also deepens dependence on one platform. Classification logic expressed in a proprietary query language or console has to be rebuilt if the organisation changes its SIEM, adds a second analytics platform or wants the same decisions applied to a data lake. The decisions about what security data is worth are the organisation's own; they should not be encoded in a form only one vendor can read.

Finally, a filter is a binary decision: pass or discard. What a security team usually needs is a third option, in which an event is kept in full but not indexed at premium prices. That option cannot be provided by the platform that charges for indexing, because it is precisely the option that reduces what the platform charges.

Consideration	Filtering inside the SIEM	Classifying at the edge
Point of metering	Usually after the event has been ingested and counted	Before the event reaches the indexer
Possible outcomes	Keep or discard	Forward, archive in full, or drop with a recorded reason
Form of the logic	Vendor console or proprietary query language	Plain-text rule files under version control
Testing before release	Limited and platform-specific	Cases, near misses, engine validation and replay against real traffic
Portability	Tied to one platform	Applies to any downstream SIEM, hub or archive
Audit answer to 'why is this event missing?'	Depends on configuration history	A lookup: the rule that decided it is on the event's own record

03

The approach: classify at the edge, keep everything

The SAFI approach places a classification pipeline between the systems that produce security events and the platform that bills for them. It reads every event, decides what it is using rules a person can read, and routes it accordingly. Nothing on the source side changes: forwarders, syslog, Windows event collection, firewalls and endpoints continue to send what they already send, and the collector accepts it.

The pipeline runs on Vector, an open-source, high-throughput observability data pipeline. Vector parses, enriches, masks, aggregates and routes events, and its configuration is plain text that can live in version control. Downstream, OpenSearch acts as the data hub: the archives become indexes with typed templates written before the first event lands, so that a field is a field rather than free text, and archived data stays searchable without passing through the licensed indexer. The organisation's existing SIEM keeps receiving the stream it needs. Where an organisation wants detection content running over the archived estate as well, Wazuh can be added as an embedded SIEM reading from the same store; like the other add-ons, it is deployed only when a client asks for it.

Three design commitments run through the whole approach. The first is that every event is classified, not sampled; the pipeline makes a decision about each one and records which rule made it. The second is that every event is kept: a raw archive receives everything that arrived, whatever the verdict, so a classification mistake costs a correction rather than lost evidence. The third is that the rules belong to the organisation. They are plain text files, readable without special tooling, changed through a defined process and handed over in full.

Two further stages protect the quality of what is classified. Masking runs first, on the collector, before classification and before any archive is written. Card numbers, credentials, bearer tokens, national identifiers, IBANs and email addresses are replaced in place, so no downstream copy holds the original. The mask preserves shape: a card number keeps its last four digits and its field length, because many records are parsed by counting separators and a mask that shifted one would break a parser further down. Schema-drift detection runs alongside classification. When a source changes the shape of what it sends, for example after a vendor update adds or renames a field, the affected event is kept and flagged rather than classified, because nothing downstream should be trusted to judge an event whose structure has changed.

Parsing once at the edge has a benefit beyond cost. A log line arrives as a sentence, and a SIEM that stores the sentence makes every analyst extract the same values with their own pattern at search time. When the collector parses once, before anything is written, what lands downstream is typed: source, event identifier, action, bytes, zone, the rule that classified the event and a lineage identifier that joins it to its own raw record. Two analysts asking the same question get the same answer.

- Classify every event, and record the rule that decided it on the event's own record.
- Keep every event in a raw archive, whatever its verdict.
- Mask sensitive values before classification and before any archive exists.
- Flag schema drift rather than silently misclassifying a changed event.
- Keep the rules in plain text, owned and versioned by the organisation.

04

Three endings and one archive

Every event that leaves the classification chain has one of three endings. The distinction between them is the heart of the approach, because it replaces the binary keep-or-delete decision with a choice about where an event lives and what it costs to keep it there.

Forwarded events are indexed by the SIEM and billed. These are the events that feed detections, dashboards and the searches analysts run every day. Some categories should never be reduced by any rule: process creation, logon sessions, privilege use, service installation and account changes are examples of telemetry that detection content depends on, and a sound rule set keeps them on the forwarded path explicitly.

Archived events are kept in full at storage prices, searchable in the data hub, and off the ingest line. An event in the archive tier has not been judged worthless; it has been judged not worth indexing at indexer prices. It remains available for an investigation next month or an audit next year, and it can be brought back into the SIEM when there is a reason to do so.

Dropped events are not sent anywhere downstream, but they are not silently lost either. The drop stage only ever sees an event that an earlier rule has already moved off the default of keep. An event must therefore be named twice before it can be discarded: once as low value, and once more as noise. Nothing is dropped by omission, and every dropped event is written to its own archive with the rule that dropped it beside it. When somebody asks why an event is not in the SIEM, the answer is a lookup rather than an argument.

Beneath all three endings sits the raw archive. Every event reaches it, before anything has collapsed or re-routed it, carrying the class it was given and the rule that decided it. This is what makes the other decisions auditable rather than asserted. It is also what makes replay possible: when an investigation needs a window of archived events in the SIEM, the Replay add-on reads that window from the raw archive and resends it downstream, and nothing else. Alongside the raw archive the collector writes a record of what was forwarded, so the reduction is arithmetic between two archives on the collector's own disk, not a figure in a report.

In the SAFI design the collector writes four archives: raw (everything that arrived), sent (what was forwarded), the archive tier (full fidelity, off the ingest line) and dropped (with the reason). Every event is in at least one of them, and in the raw archive always.

> **Data is not deleted; it stops being metered.**

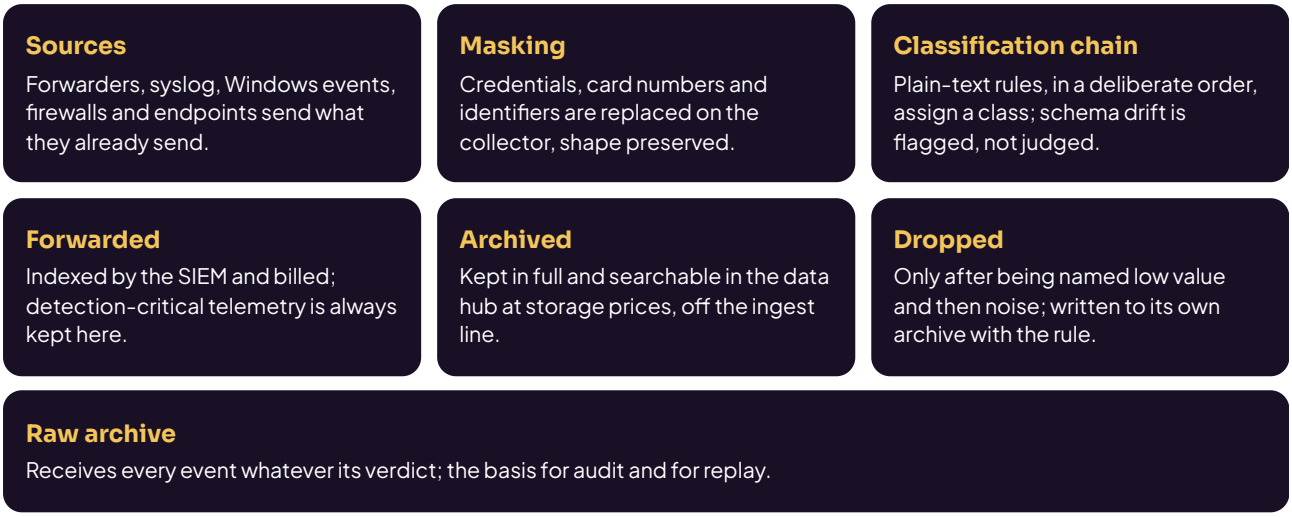


Figure 1. The edge-classification pipeline: every event is masked, classified and routed to one of three endings, and every event also reaches the raw archive.

05

Rules you can read and test

When a pipeline decides which events reach the SIEM, its rules decide what the security operations centre can see. That is a serious responsibility to hand to anything the organisation cannot inspect. A SAFI rule is a plain text file on the collector that states what it matches and what class it assigns. There is no compiled logic, no vendor judgement hidden in a binary, and no need for special tooling to read it.

Order is part of the rule. Precedence is the order in which the pipeline wires the rule files together, and it is deliberately not the order in which filenames happen to sort. A rule that runs later overrules one that ran earlier, so reading the chain from top to bottom is reading what actually happens to an event.

Readability alone is not enough. A rule that can be edited but not tested is a risk rather than a feature, because it invites changes nobody can verify. The process below exists because changing a classification rule changes what a security team can see, and that change has to be provable before it reaches a collector and reversible after it does.

The near miss deserves particular attention. A case proves that a rule fires. A near miss, a log line that looks similar but must not hit the rule, proves that it does not fire too widely. An over-broad drop or archive rule is the failure nobody notices: a missing detection eventually shows up as a gap, but a rule that quietly diverts evidence can do so for months. Running the full catalogue of

cases and near misses on every change turns this from a matter of care into a matter of build success or failure.

Where a rule set is agreed with a security team event by event, each case can carry two verdicts: what the chain produces and what was agreed it should produce. Where those differ, the difference is reported as a divergence on every build until somebody resolves it, rather than being hidden inside a passing test suite. The direction of a divergence matters most. A chain that keeps something the team asked to drop has cost a little money and said so; a chain that drops something the team asked to keep has lost evidence. The design should make the second kind structurally hard, and the double-naming gate on the drop stage does exactly that.



Figure 2. The seven-step rule process: every change is written, tested, validated, measured, released, and reversible.

Property	What it means	Why it matters
Plain text	Each rule is a readable file stating what it matches and the class it assigns	Anyone with access can answer why an event was classified as it was
Deliberate order	Precedence is set by how the chain is wired, not by filename sorting	Reading the chain top to bottom is reading what happens
Case and near miss	Every rule has a log it must catch and a similar log it must not	Over-broad rules are caught before they divert evidence
Engine-validated	The pipeline engine validates the real configuration	Reading a file and asking the engine are different acts
Measured before release	Replay against last week's raw archive shows moved verdicts and volume	The effect on real traffic is known before anyone is affected
Versioned and pushed	Rules travel as a versioned configuration pack from the Platform Console	No ad hoc changes over remote sessions; every collector runs a known version
Reversible and audited	The previous pack is retained and each change records who made it	A regretted change costs a reload, not a reconstruction

06

Measuring the effect

Claims about SIEM cost reduction are easy to make and hard to check, so we hold ourselves to a simple discipline: every figure is labelled Verified or Illustrative, and the two are never blended. A Verified figure has been measured on real traffic. An Illustrative figure is arithmetic on stated, hypothetical inputs, offered to show the shape of a calculation and nothing more.

The Verified figure is this. On one production estate, classifying events at the edge sent 86.4% less volume to the SIEM, measured across 7,959,818 events. That figure is Verified. It is not a guarantee for every estate. The mix of sources, the detections in use, existing filtering at source and retention obligations all change the result, and anyone who promises a fixed percentage before seeing an organisation's traffic is guessing.

The shape of the saving is more predictable than its size. Estates with many chatty infrastructure sources, verbose application logging or duplicated forwarding tend to have more room. Estates that already filter carefully at source tend to have less. Aggregation also contributes: where the same event arrives many times, collapsing the repeats on the collector shows up as bytes that arrived and never left.

Because the collector writes both the raw archive and the record of what was forwarded, the measurement does not depend on a report or an estimate. The reduction is the difference

between two archives on the collector's own disk, over a defined period, on the organisation's own data. That is the only figure an organisation should base a decision on, and it is what a proof of concept should produce.

The table below is an Illustrative cost model. Every input is hypothetical, prices are expressed in neutral cost units rather than any currency or vendor price list, and the split between endings is an assumption chosen for readability, not a prediction and not derived from the Verified figure above. Its purpose is to show which inputs drive the result: the share forwarded, the ratio between indexer and storage prices, and the cost of running the pipeline itself.

> **The only honest version of the saving is the one measured on your own traffic, between your own raw archive and what you actually forwarded.**

Illustrative line item (hypothetical inputs)	Forward everything (Illustrative)	Classify at the edge (Illustrative)
Daily volume arriving (hypothetical)	500 GB	500 GB
Share forwarded to the SIEM (assumed)	100%	25%
Share archived in full (assumed)	0%	65%
Share dropped with a reason (assumed)	0%	10%
Volume indexed per day	500 GB	125 GB
SIEM ingest at a hypothetical 1.00 cost unit per GB	500 CU per day	125 CU per day
Archive storage at a hypothetical 0.05 CU per GB (raw archive plus archive tier: 500 GB + 325 GB)	0 CU per day	41.25 CU per day
Pipeline operation (hypothetical allowance)	0 CU per day	20 CU per day
Illustrative total	500 CU per day	186.25 CU per day
Illustrative change	Baseline	About 63% lower, on these hypothetical inputs only

07

Governance, evidence and reversibility

Reducing log volume is only defensible if an organisation can say what it kept, where, for how long, and why. The edge-classification approach is designed so that each of those questions has a

documentary answer rather than a verbal one.

Retention becomes independent of indexing. Because the raw archive keeps every event whatever its verdict, the period for which records are retained is set by the archive's retention policy, not by what the SIEM was sent. An organisation can keep a short, fast, expensive index and a long, complete, inexpensive archive, and decide each according to its own obligations.

Every decision is attributable. Each event carries the rule that classified it, each rule change carries who made it and when, and each configuration pack is versioned. When an auditor or an investigator asks why an event is not in the SIEM, the answer is the rule on the event's record and the change history of that rule. When a change is regretted, the previous pack is restored with a reload; the raw archive was never touched, so nothing has to be reconstructed.

The security operations team must own the classification, not merely be consulted on it. A cost project that removes visibility will be reversed after the first difficult incident, and rightly so. In practice this means walking through event identifiers with the team, recording their agreed verdict beside the pipeline's verdict, and treating every divergence as a finding to resolve rather than a detail to hide. It also means making retrieval easy: if bringing a week of archived events back into the SIEM takes a ticket and a long wait, analysts will reasonably resist anything leaving the index.

The rule set is designed to sit within widely published references on log management, so that a reviewer has texts to hold the design to. This is not a claim of certification, and it is not legal or compliance advice; each organisation should confirm its own obligations with its own advisers.

- NIST SP 800–92, Guide to Computer Security Log Management: plan what is logged and manage volume deliberately rather than collecting everything or nothing.
- NIST SP 800–53 Rev. 5, Audit and Accountability family: records must be generated, retained long enough and reviewable; the raw archive keeps retention independent of indexing.
- CIS Controls v8, Control 8, Audit Log Management: collection, adequate storage, centralisation, retention and review, of which adequate storage is the control volume reduction most directly serves.
- ISO/IEC 27001:2022, Annex A 8.15 Logging and 8.16 Monitoring activities: produce, keep and protect logs, and monitor for anomalous behaviour.
- PCI DSS v4.0, Requirement 3.4.1: masking of card numbers when displayed; the collector mask keeps only the last four digits and is applied before any archive or index exists.
- MITRE ATT&CK data sources: the telemetry detections depend on, such as process creation, logon, privilege use, service installation and account change, which no rule should reduce.

08

Implementation roadmap

The approach is delivered in four stages, each with a defined outcome and an exit criterion agreed before it starts. The principle throughout is that the organisation should be able to stop at the end of any stage with something it owns and can use.

A proof of concept is time-bounded and milestone-gated, and it is measured on the organisation's own data. A collector is placed alongside the existing forwarding path for a representative set of sources, the security team walks through the event identifiers that matter, and the reduction is measured between the raw archive and the forwarded record. The output is a figure the organisation measured itself, and a draft rule set its analysts have reviewed.

The build installs collectors and the data hub by script, verified on the machine, in a defined scope. Rules are completed with cases and near misses, the archives are set up with typed templates and retention, masking is confirmed against the organisation's own sensitive fields, and the Platform Console is configured to push versioned configuration packs. Every release of the console pack is checked by 3,151 automated checks per release (Verified).

Handover transfers ownership in substance, not only in name. The organisation receives the configuration, the rules with the reason for each, the documentation and training for its own team, so that it can run, change or replace what was delivered without Alin Li. There is no lock-in, including to us.

In production, support and planned maintenance releases are agreed separately after go-live. Rule changes continue to follow the seven-step process, whoever makes them, and the organisation's team makes them from the console.

Phase	Outcome	Exit criteria
Proof of concept	Edge classification running beside the existing path on a representative set of sources; reduction measured on the organisation's own data	Measured reduction agreed by both parties; draft rule set reviewed by the SOC; divergences listed; go or no-go decision recorded
Build	Collectors, masking, classification chain, archives and data hub installed by script and verified on the machine in the agreed scope	Every rule has a case and a near miss; full catalogue passes; engine validation passes; replay of a recent period reviewed; retrieval from archive demonstrated
Handover	Configuration, rules with reasons, documentation and training transferred to the organisation's team	The organisation's team makes, tests, replays, pushes and reverses a rule change unaided; ownership of all artefacts confirmed
In production	The pipeline runs as part of normal security operations, with support and maintenance agreed separately	Agreed service terms in place; monitoring of collectors and sinks; scheduled review of rules and divergences

09

Risks and how to manage them

Any change to what reaches a SIEM carries risk, and it is better to name those risks plainly than to suggest there are none. The table below sets out the principal risks of edge classification and the controls the approach uses to manage each. None of these controls removes the need for judgement; they make that judgement visible, testable and reversible.

Risk	How it shows up	How it is managed
Loss of evidence through an over-broad rule	Events a detection or investigation needs are archived or dropped	Near misses for every rule; double-naming gate before any drop; raw archive keeps every event; replay before release
Detection gaps	Detection content depends on telemetry no longer indexed	Detection-critical categories kept on the forwarded path by explicit rule; SOC ownership of verdicts; Replay add-on for investigation windows
Schema drift after a source update	A changed event shape is misclassified silently	Drift detection flags and keeps the event rather than judging it
Sensitive data in archives	Credentials or card numbers stored in cheaper tiers	Masking on the collector before classification and before any archive is written
Untested or ad hoc rule changes	A change made under pressure behaves unexpectedly	Seven-step process; engine validation; versioned packs pushed from the console; no remote-session edits
Overstated savings	A business case rests on a percentage measured elsewhere	Only Verified or Illustrative figures; decision based on measurement on the organisation's own data
Dependence on the supplier	The organisation cannot change or run the pipeline itself	Open-source components; full handover of rules, configuration and documentation; training for the organisation's team
Analyst resistance	The SOC sees the project as a loss of visibility	Verdicts agreed event by event; divergences reported on every build; easy retrieval of archived data

CHECKLIST

Before you change what reaches your SIEM

- [ ] Identify which sources drive your ingest volume and which detections, dashboards and searches actually use them.
- [ ] Confirm that a raw archive will receive every event, whatever its verdict, and that its retention meets your obligations.
- [ ] List the telemetry categories that no rule may reduce, and make keeping them on the forwarded path explicit.
- [ ] Require every rule to be plain text, with a case it must catch and a near miss it must not.
- [ ] Check that dropping an event requires two separate decisions, low value and then noise, and that every drop is recorded with its rule.
- [ ] Confirm that sensitive values are masked on the collector before classification and before any archive is written.
- [ ] Ensure schema drift is detected and flagged rather than silently classified.
- [ ] Replay a recent period through any proposed change and review every moved verdict and its volume before release.
- [ ] Push changes only as versioned configuration packs, with an audit trail and a tested route back to the previous pack.
- [ ] Measure the reduction on your own data, between the raw archive and what was forwarded, before making any business case.
- [ ] Agree classification verdicts with your SOC and track divergences until they are resolved.
- [ ] Confirm that you own the rules, configuration, documentation and data, and could run or replace the pipeline without the supplier.

CONCLUSION

Where to start

The cost of a SIEM does not have to rise in step with the volume of everything an estate can produce. Classifying every event at the edge, with rules the organisation can read, test and reverse, lets the SIEM carry what is worth indexing while everything else is kept in full, searchable and inexpensive. Because a raw archive receives every event and every decision is recorded beside the event it decided, reducing cost does not reduce what the organisation can prove.

The approach is built on open-source components, handed over to the organisation that runs it and owned by that organisation. Data is not deleted; it stops being metered. The right first step is a

time-bounded proof of concept on your own traffic, which produces a figure you measured yourself and a rule set your analysts have reviewed, and leaves you free to decide what happens next.

GLOSSARY

Terms used

SIEM	Security information and event management: a platform that indexes, correlates and alerts on security events, commonly priced on ingest volume.
Edge classification	Deciding what each event is, and where it should live, on a collector before it reaches the indexer.
Forwarded	An event sent to the SIEM, indexed and therefore billed.
Archived	An event kept at full fidelity and searchable in the data hub at storage prices, off the ingest line.
Dropped	An event not sent downstream after being named both low value and noise, recorded with the rule that dropped it.
Raw archive	The archive that receives every event, whatever its verdict, with its class and deciding rule.
Near miss	A test log line that resembles what a rule catches but must not be caught by it.
Schema drift	A change in the structure of events a source sends, such as an added or renamed field.
Masking	Replacing sensitive values on the collector while preserving the shape of the record.
Replay	An add-on that resends a chosen time window from the raw archive to a downstream platform.
Configuration pack	A versioned bundle of rules and pipeline configuration pushed to collectors from the Platform Console.
Vector	An open-source observability data pipeline used here to parse, mask, classify, aggregate and route events.
OpenSearch	An open-source search and analytics engine used here as the data hub for the archives.
Wazuh	An open-source security platform, available as an add-on embedded SIEM reading from the data hub.
Verified	A figure measured on real traffic.
Illustrative	A figure calculated from stated, hypothetical inputs to show the shape of a calculation.

About Alin Li

Alin Li OÜ builds platforms on open-source software, hands them over to the organisation that runs them, and supports them in production, with engineers in Europe and the Gulf, working in English and Arabic.

- **No lock-in**, including to Alin Li.
- **You own what is delivered**: configuration, code, rules, documentation and data.
- **Every figure is labelled** Verified or Illustrative.

Alin Li OÜ · Registry code 17184688 · Ahtri tn 12, 10151 Tallinn, Estonia
Info@AlinLi.com · +372 5705 1771 · alinli.com

This white paper is for general information. It is not legal, regulatory or financial advice. Figures marked Verified were measured on one estate and are not a guarantee for any other; figures marked Illustrative are hypothetical. Product names belong to their owners. © 2026 Alin Li OÜ. You may share this document unchanged.