OMNARETH



WHITEPAPER

Provable Custody

Rules-based fraud prevention and tamper-evident records for high-value cargo, by road and sea

October 2026 · Technical whitepaper

OVERVIEW

Executive summary

Many of the most costly cargo losses, on the road and at sea, involve no force at all, just a lie: a driver who was never dispatched, a release order or bill of lading that was never issued, a seal that is not the one applied at origin. Each lie succeeds because the person at the gate cannot see the whole custody record, and because nobody can later prove what that record said.

Omnareth closes both gaps. Every custody handoff, release request and tracker alert is checked against the shipment's recorded state and a set of explicit custody rules before anything moves. The decision (allow, flag or deny) comes back with the exact reasons. Omnareth also remembers across shipments: who was responsible for each refusal, which release orders have been seen where, and which identities analysts have confirmed as fraudulent, so a crew that is turned away at one gate is recognized at the next. And every decision is sealed with a cryptographic integrity hash and anchored on a ledger, so the custody record can later be shown to be complete and unaltered.

This paper describes how Omnareth models custody, which threats its reference rule set blocks, how it recognizes repeat fraud across shipments, how its tamper evidence works, how fast it is, and where its guarantees end. Every capability and figure in it was verified against the running system.

23	22 µs	75 ms	3 / 3
custody threats covered by the reference rule set	median rule evaluation, 34 rules	median end-to-end decision, stored and audited, lab test	database tampering methods caught in testing

What you can prove with Omnareth

Who held the cargo, when, and on whose authority: every handoff and release is a decision tied to the shipment.

That every unauthorized attempt was refused, with the rule and reason that refused it.

That repeat attempts were recognized: which identities were implicated, which cases were opened, and what was known about them at each decision.

That the record has not been edited since: each decision is recomputable from its audit snapshot and matched against its on-chain anchor.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 1

The custody problem

A container crossing an ocean passes through several independent parties: the shipper, a drayage carrier, the ocean line, one or more terminals, a customs broker, and the consignee's trucking company. Each keeps its own records in its own system. Custody is handed over many times, usually by people who have never met and who verify each other with paperwork.

That structure creates a predictable set of attacks:

Deception at the gate

- **Fictitious pickup.** A driver arrives with a plausible story and the real release order number, which is often visible to many parties. Nothing at the gate ties the driver to the consignee's actual dispatch.
- **Forged or stale release orders.** A release order that looks right but was never issued, or was superseded, is accepted because the gate cannot check it against the carrier's system in the moment.
- **Double release.** The same release order is presented twice, at different times or gates.

Tampering in transit

- **Seal swap.** Cargo is removed and the container resealed with a different seal. The swap is only caught if someone compares the seal number with the one applied at origin.
- **Container swap.** A container with the right paperwork is not the container that was booked.
- **Pilferage.** Part of the load disappears; the seal may even be intact. A scale reading is the only signal.

At sea

- **Phantom or switched vessels.** Cargo is loaded on a vessel other than the one booked, or arrives on a different one.
- **Diversion.** A vessel goes dark, calls at a port that is not on its rotation, transfers cargo ship to ship, or discharges where the cargo was never meant to land.
- **Forged bills of lading.** Whoever presents the original bill at the port of discharge can claim the cargo. A forged or copied bill turns a document fraud into a theft.

Gaps in the record

- **Ghost handoffs.** A party that no longer holds the cargo, or never did, hands it on.
- **Records that change after the fact.** When a loss is investigated, the custody log is held in databases that the parties involved can edit. There is no way to show it is the log as it was.

> None of these attacks needs to defeat a lock. Each succeeds because a decision was made without the full custody record, or because the record could not be trusted later. Those are the two problems Omnareth is built to solve.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 2

Design principles

DETERMINISTIC

Same inputs, same decision

Rules are explicit conditions evaluated by a deterministic engine, not a model. A decision can be replayed and explained to an auditor, an insurer or a court.

EXPLAINABLE

Every decision says why

Each decision lists every rule evaluated, which ones fired, a machine-readable code, a human-readable reason and a remediation step.

FAIL-CLOSED

Missing data means no

Authorization checks are written so that a missing list or field denies rather than allows. A release cannot slip through because a consignee list was never loaded.

TAMPER-EVIDENT

The record proves itself

Every decision carries an integrity hash over exactly what was decided on, anchored on a ledger. Edits to the record, even covered-up ones, fail verification.

SYSTEM-AGNOSTIC

Works beside what you run

Omnareth does not replace your TMS, WMS or terminal system. They send events and act on decisions over a REST API.

FAST

In the transaction path

Rule evaluation takes microseconds and a full stored, audited decision about 75 ms in our lab test, so checks can run at the gate, not in a nightly report.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 3

Architecture

Omnareth sits beside the systems that already record custody. They send events (a handoff, a release request, a tracker reading) and receive a decision. Omnareth holds the current state of each shipment, evaluates the rules, persists the decision and its audit record, and anchors the record on the Omnareth chain.

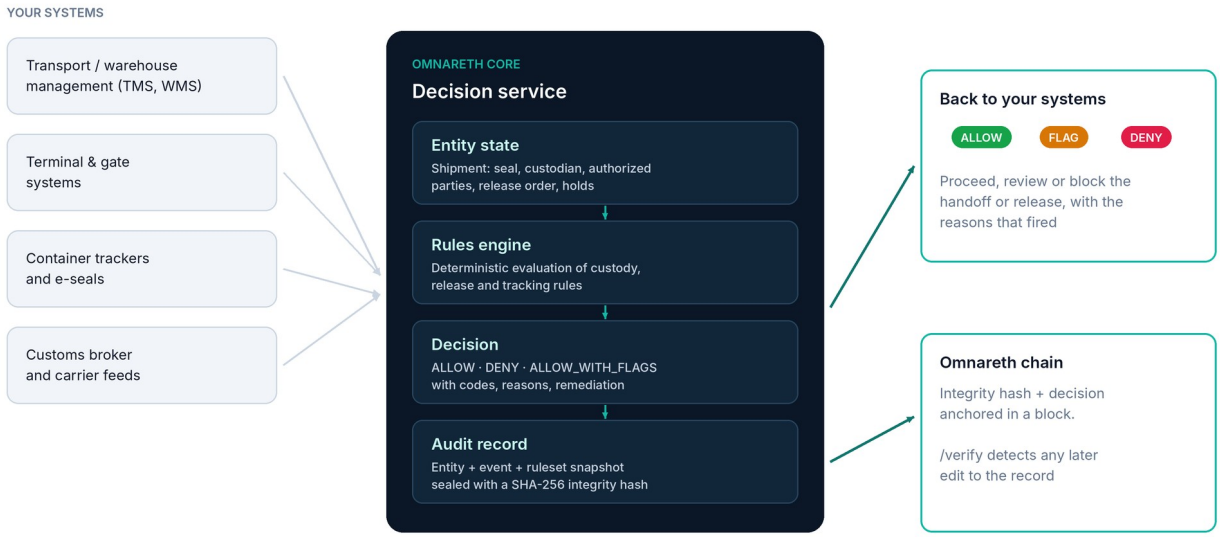


Figure 1. Events flow from your systems through the decision service; decisions flow back, and evidence is anchored on chain.

Components

COMPONENT	ROLE
Core service	REST API (port 3001). Stores entity types, shipments, rules, events, decisions and audit records in PostgreSQL, and runs the rules engine.
Rules engine	Deterministic evaluator for conditions (and, or, not, comparisons). Pure library with no I/O; also usable on its own.
Fraud prevention	Part of the core service. Adds fraud context to each event before rules run, records who was implicated in each decision, runs detectors, and keeps the watchlist and fraud cases.
Omnareth chain node	Ledger (port 4040) that records each decision's integrity hash and status in a block, accepted through PACT Protocol's Proof-of-Activity consensus. Blocks can be persisted to disk.
Verification	`GET /v1/audits/:id/verify` recomputes a decision's hash and compares it with its anchor.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 4

Modeling custody

The shipment is an entity

A shipment is stored as an entity whose shape is defined by a JSON Schema and validated on every write. The reference model records what a custody decision needs to know:

FIELD	PURPOSE
`sealNumber`, `containerNumber`	The high-security seal applied at origin and the booked container
`currentCustodian`	The party that holds the cargo right now
`authorizedCustodians`	Parties approved to take custody along the route
`authorizedConsignees`	Parties approved to take final release
`releaseOrderNumber`	The valid release order issued by the carrier
`dispatchedDrivers`	Drivers pre-registered by the consignee's trucking company
`customsStatus`, `holdActive`	Clearance and any commercial, legal or security hold
`minWeightKg`, `maxWeightKg`	Accepted scale tolerance set at booking
`vesselImo`, `portOfLoading`	For sea: the booked vessel and port of loading
`authorizedDischargePorts`, `scheduledPortCalls`	Where the cargo may land, and the vessel's published rotation
`billOfLadingNumber`, `billOfLadingType`	The bill issued by the ocean carrier; an ORIGINAL bill must be surrendered at release

Events describe what is happening

Six event types cover the custody lifecycle by road and sea. Each names the shipment it concerns, so rules can read both the event and the shipment's current state.

EVENT	CARRIES
CUSTODY_HANDOFF	From and to party, location, container and seal number, seal condition, scale weight
CARGO_RELEASE	Releasing party, recipient, release order, driver and truck, seal at the gate
TRACKING_UPDATE	Geofence status, door sensor, whether at an authorized facility, unscheduled stop time
VESSEL_LOAD	Vessel IMO and voyage, port, seal number and condition at loading

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

EVENT	CARRIES
VESSEL_DISCHARGE	Vessel IMO, port, seal number and condition at discharge
VOYAGE_UPDATE	AIS gap in hours, ship-to-ship transfer, any port call

Rules compare the two

A rule is a condition plus a consequence. Most custody rules compare a field of the event with a field of the shipment. The rule below denies a handoff whose seal is not the seal applied at origin:

```
{
  "name": "Seal Number Mismatch at Handoff",
  "appliesToEntityType": "Shipment",
  "appliesToEventType": "CUSTODY_HANDOFF",
  "severity": "CRITICAL",
  "condition": { "type": "compare", "field": "handoff.sealNumber",
                 "operator": "!=", "valueField": "entity.sealNumber" },
  "consequence": { "decision": "DENY", "code": "SEAL_MISMATCH",
    "message": "Seal number does not match the seal applied at origin",
    "remediation": "Hold the container and request a physical inspection" }
}
```

Authorization rules are written to fail closed. Instead of "deny if the driver is *not in* the list", which would pass if the list were missing, the rule says "deny unless the driver *is in* the list":

```
{ "type": "not", "condition":
  { "type": "compare", "field": "release.driverId",
    "operator": "in", "valueField": "entity.dispatchedDrivers" } }
```

Your system records approved transfers

Omnareth decides; the system of record acts. When a handoff or release is allowed, the TMS, WMS or terminal system updates the shipment's custodian and status through the API. Denied attempts never change custody, so an attacker cannot use a refused request to move the record.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 5

Threat coverage

The reference rule set has 29 rules covering 23 threats across handoff, release and transit. DENY blocks the action. FLAG allows it but returns ALLOW_WITH_FLAGS with the reasons, so an operator can follow up without stopping the flow.

THREAT	WHAT OMNARETH CHECKS	RULE CODE	DECISION
Seal tampering or seal swap	Seal at handoff differs from the origin seal, or is reported broken	SEAL_MISMATCH · SEAL_BROKEN	DENY
Container swap	Container number differs from the booking	CONTAINER_MISMATCH	DENY
Custody gap / ghost handoff	Party handing over or releasing is not the recorded custodian	CUSTODY_GAP	DENY
Unauthorized custodian	Receiving party is not approved for the route	UNAUTHORIZED_CUSTODIAN	DENY
Unauthorized release	Release recipient is not an authorized consignee	UNAUTHORIZED_RELEASE	DENY
Forged or stale release order	Release order at the gate does not match the order issued	INVALID_RELEASE_ORDER	DENY
Fictitious pickup	Driver is not on the consignee's dispatch list	FICTITIOUS_PICKUP	DENY
Double release	Shipment has already been released	DUPLICATE_RELEASE	DENY
Premature release	Customs not cleared, or a hold is active	CUSTOMS_NOT_CLEARED · HOLD_ACTIVE	DENY
Seal compromised at release	Seal missing, different or broken at the gate	SEAL_COMPROMISED	DENY
Door breach in transit	Doors opened outside an authorized facility	DOOR_BREACH	DENY
Pilferage or contraband	Scale weight outside the booked tolerance	WEIGHT_DISCREPANCY	FLAG
Hijack precursors	Route deviation or unscheduled stop over 30 minutes	ROUTE_DEVIATION · UNSCHEDULED_STOP	FLAG
Skipped inspection	Handoff recorded without a seal check	SEAL_NOT_INSPECTED	FLAG
Phantom or switched vessel	Container loaded on, or arriving on, a vessel other than the booked one	VESSEL_MISMATCH	DENY

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

THREAT	WHAT OMNARETH CHECKS	RULE CODE	DECISION
Wrong port of loading	Loading at a port other than the booked port	LOAD_PORT_MISMATCH	DENY
Seal tampering at sea	Seal missing, different or broken at loading or discharge	SEAL_COMPROMISED	DENY
Cargo diversion	Discharge at a port that is not an authorized port of discharge	UNAUTHORIZED_DISCHARGE	DENY
Ship-to-ship transfer	A transfer between vessels at sea reported for the cargo	SHIP_TO_SHIP_TRANSFER	DENY
Forged bill of lading	Bill of lading presented at release is not the one the carrier issued	INVALID_BILL_OF_LADING	DENY
Release without the original bill	Original bill of lading not surrendered at release	BL_NOT_SURRENDERED	DENY
AIS dark period	Vessel stopped transmitting its position for over 6 hours	AIS_GAP	FLAG
Unscheduled port call	Call at a port not on the vessel's published rotation	UNSCHEDULED_PORT_CALL	FLAG

Validated against a reference scenario

The rule set is exercised end to end by a 22-event scenario across 3 shipments. A clean journey from Shenzhen to Long Beach is attacked six times on the way and released once, correctly; a second shipment shows a theft pattern in transit; a third sails from Antofagasta to Rotterdam through a switched vessel, an AIS dark period, a ship-to-ship transfer, a diversion attempt and a forged bill of lading before its correct release. Every decision must match its expected outcome or the scenario fails.

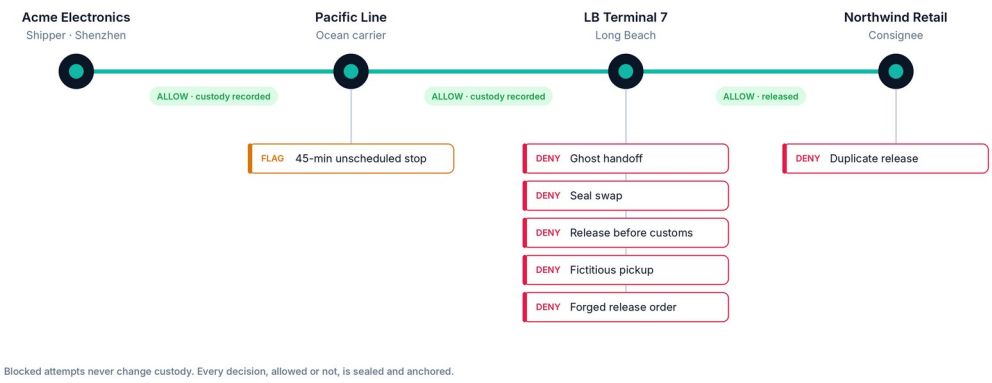


Figure 2. Shipment 1: the legitimate custody chain proceeds while every attack against it is refused.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 6

Fraud prevention across shipments

Custody rules judge one event at a time against one shipment. Theft crews do not stop after one refusal: the same driver tries another shipment, a release order copied from one booking is presented for another, or someone on the inside adds an impostor to a dispatch list. Omnareth keeps a memory across shipments so that each new attempt is judged with what earlier attempts revealed.

Holding the right party responsible

An impostor names a real consignee and quotes a real release order; both are innocent. So a rule states which identifiers in the event it holds responsible when it fires, in `implicates`. Only those are recorded against the denial:

```
"consequence": { "decision": "DENY", "code": "FICTITIOUS_PICKUP",
  "message": "Driver is not on the consignee's dispatch list",
  "implicates": ["release.driverId", "release.truckPlate"] }
```

Which event fields are identifiers is declared with the entity type. The Shipment type tracks drivers, trucks, parties, containers, vessels, release orders and bills of lading. A legitimate driver turned away because customs has not cleared is not implicated, and so is never counted as suspicious.

Fraud context in every decision

Before rules run, Omnareth computes `context.fraud` for the event from its watchlist and from earlier decisions. Fraud rules act on it like any other field:

FIELD	MEANING
`blocked`, `watched`, `matches`	Watchlist entries for any driver, truck, party, container or release order in the event
`priorImplicatedDenials`	Denied events in the last 30 days in which one of these identities was implicated
`referenceOtherEntities`	Other shipments a single-use reference in the event (release order, bill of lading) has been presented for
`actorImplicatedEntities24h`	Other shipments where a hopping actor in the event (a driver) was implicated in a denial in the last 24 hours

The server always computes this context; anything a client sends under `context.fraud` is replaced. It is stored in the audit record and covered by its integrity hash, so the evidence shows exactly what was known about each identity when the decision was made.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

Fraud rules

The reference set adds five rules over the fraud context:

THREAT	WHAT OMNARETH CHECKS	RULE CODE	DECISION
Known fraudster returns	A driver, truck or party in the event is blocked on the fraud watchlist	WATCHLIST_BLOCK	DENY
Watched identity	An identity is under watch, added by an analyst or automatically	WATCHLIST_MATCH	FLAG
Repeat offender	An identity was implicated in 2 or more denied events in the last 30 days	REPEAT_OFFENDER	FLAG
Compromised release document	The release order or bill of lading has also been presented for another shipment	REFERENCE_REUSE	FLAG
Driver hopping	The driver was implicated in denied pickups on 2 or more other shipments in 24 hours	DRIVER_HOPPING	FLAG

Detectors and fraud cases

After every decision, detectors look for patterns no single event shows. Each pattern opens one case per subject; later events update it rather than opening duplicates.

PATTERN	TRIGGER	WHAT HAPPENS
Repeat offender	Any identity (driver, vessel, party, bank account…) implicated in 2 denied events within 30 days	Case opened (high); the identity is watched for 90 days
Reference reuse	A single-use reference (release order, bill of lading) presented for 2 or more entities	Case opened (critical); every further use is flagged
Actor hopping	An actor (a driver, a bank account) implicated in denials on 3 or more entities within 24 hours	Case opened (high)
Burst	3 or more denied events at one place within 60 minutes	Case opened (high)

Analysts work the cases through the fraud API, which requires an analyst or administrator account. Their decision changes what Omnareth does next:

OUTCOME	EFFECT
Confirmed fraud	The subject is blocked on the watchlist: every later event involving it is denied, on any shipment.
False positive	The automatic watch is lifted and the subject's part in the denials behind the case is cleared, so they no longer count against it.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

OUTCOME	EFFECT
Investigating · Closed	Track the case without changing the watchlist.

An analyst's watchlist entry always takes precedence: detectors never downgrade a block or override a manual entry.

Validated against a fraud scenario

A second reference scenario follows one impostor across 3 cleared shipments. All 5 decisions must match their expected outcome and verify on chain, or the scenario fails.

#	WHAT HAPPENED	DECISION AND RULES	FRAUD CASES
1	Impostor quotes shipment A's real release order	DENY · FICTITIOUS_PICKUP	
2	Same driver and truck try shipment B	DENY · FICTITIOUS_PICKUP	Repeat offender opened for driver and truck; both watched
3	Impostor presents A's release order for shipment C	DENY · INVALID_RELEASE_ORDER · FICTITIOUS_PICKUP · RELEASE_ORDER_REUSE · WATCHLIST_MATCH · REPEAT_OFFENDER · DRIVER_HOPPING	Release-order reuse and driver hopping opened
–	Analyst confirms the repeat-offender case (3 shipments, 3 denials)		Driver blocked
–	An insider adds the impostor to shipment C's dispatch list		
4	Impostor returns with C's real order, a clean truck and a request claiming fraud.blocked = false	DENY · WATCHLIST_BLOCK	Audit record shows blocked = true
5	Dispatched driver collects shipment C	ALLOW	

The insider case

In step 4 every custody check passes: someone has added the impostor to the dispatch list, the release order is genuine and the truck is new. The request even claims the driver is not blocked. Omnareth denies it on the watchlist, and the audit record shows the block it acted on.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

Beyond cargo

Nothing in fraud prevention is specific to cargo. Any entity type can declare its identifiers, and the same detectors, watchlist and analyst workflow apply. Each identifier is a SUBJECT (it can be held responsible) or CONTEXT (a place); a `singleUse` reference belongs to one entity only; a `hopping` actor reveals a scheme when it is implicated across several entities. A vendor master, for example:

```
"fraud": { "identifiers": [
  { "path": "payment.bankAccount", "type": "BANK_ACCOUNT", "role": "SUBJECT", "hopping":
true },
  { "path": "change.senderDomain", "type": "EMAIL_DOMAIN", "role": "SUBJECT", "hopping":
true },
  { "path": "payment.requestedBy", "type": "EMPLOYEE", "role": "SUBJECT" } ] }
```

The payments reference set built on it has 11 rules covering bank-detail change fraud, lookalike domains, payment diversion, duplicate invoices and self-approval. Its scenario follows a business email compromise campaign: one fraud account pushed onto 3 vendors is held responsible (never the vendors or the clerk), recognized across them, confirmed by an analyst and refused even after an insider marks it verified. All 9 decisions match their expected outcome and verify on chain.

Reference sets for insurance claims, e-commerce refunds, lending, healthcare claims, telecom, government benefits, cybersecurity and identity verification follow the same pattern, each with its own identifiers: the device behind a claims ring across policies and copied police reports; the gift card collecting refunds across orders and reused return labels; the phone behind synthetic identities across loan applications, and one person applying many times; the clinic billing phantom visits across members and copied prior authorizations; the rogue store agent behind SIM swaps, port-outs and drop-address orders across accounts; the account collecting benefits for stolen and deceased identities across cases; the address behind credential stuffing and MFA push bombing across user accounts; the emulator behind forged documents and deepfake selfies across onboardings, and one passport presented for two people. Each scenario follows a ring to an analyst's confirmation and an insider's override, with all 8 decisions verified on chain.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 7

Evidence and tamper detection

What is recorded for each decision

Every evaluated event produces a decision and an audit record. The audit record captures exactly what the decision was based on:

- **Entity snapshot**: the shipment's state at the moment of the decision.
- **Event snapshot**: the handoff, release or tracking payload as received.
- **Ruleset snapshot**: the name, version, condition and consequence of every rule that was evaluated.
- **Context**: what was known beyond the shipment, including the fraud context computed for the event.
- **Integrity hash**: SHA-256 over a canonical JSON encoding (keys sorted) of all four, so the same content always produces the same hash regardless of how a database stores it.

Anchoring on the Omnareth chain

The Omnareth chain accepts records through PACT Protocol, the Proof-of-Activity consensus technology developed by Habify Labs, Inc.

When anchoring is enabled, the core service writes the decision id, event id, integrity hash and decision status to the Omnareth chain node and records the transaction hash and block height with the decision. Anchoring never blocks a decision: if the node is unreachable, the decision is still returned and the receipt records the failure.

Verification

`GET /v1/audits/:id/verify` recomputes the hash from the stored snapshots and compares the result, the decision id and the decision status with what was anchored.

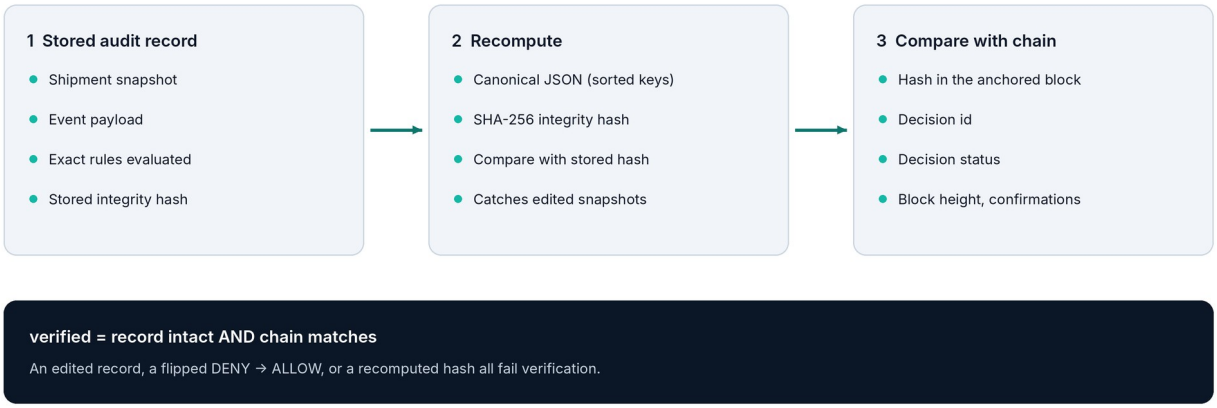


Figure 3. A record verifies only if it still hashes to its stored value and matches its anchor.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

We tested this by editing the database directly, the way an insider with database access could:

CHANGE MADE IN THE DATABASE	RECORD INTACT	CHAIN	RESULT
None	true	match	Verified
A DENY decision flipped to ALLOW	true	mismatch	Fails
Seal number edited in the audit record	false	mismatch	Fails
Seal number edited and the stored hash recomputed to hide it	true	mismatch	Fails

Why the last case matters

An insider who edits a record and then recomputes its stored hash defeats any check that lives only in the database. The anchored copy is outside their reach, so the record still fails verification.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 8

Performance

Custody checks are only useful if they run where the decision is made: at the gate, during the handoff, as the tracker reports. Omnareth is fast enough to sit in that path.

22 µs
rule evaluation, median (34 rules)

~46,000
evaluations per second, single core

75 ms
decision via API, fraud-checked, stored and audited (median)

531 ms
the same, waiting for the PACT commit (median)

MEASUREMENT	P50	P95	P99
Rule evaluation only (34 rules, in process)	22 µs	38 µs	71 µs
Full API decision with fraud checks, stored with audit record	75 ms	93 ms	117 ms
The same, waiting for the PACT commit	531 ms	685 ms	687 ms

Measured on 4 vCPU Intel Xeon @ 2.1 GHz, Node.js 22, PostgreSQL 16 and a four-validator PACT network on the same host, production settings (sign-in, field encryption). API figures are from a fresh install and include request validation, entity lookup, fraud context, evaluation, persistence of the event, decision, every rule evaluation and the audit record, and fraud detection. The benchmark repeats one attack, so the attacker's history grows throughout; latency stays flat because every fraud query is bounded. One client sending requests in sequence. By default the answer returns before the record commits on PACT; it commits about 0.4 s later at the median. With 8 to 32 concurrent clients, one service process handled 30 to 44 decisions per second.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 9

Deployment and integration

RUNTIME

Node.js services

Node.js 20.12 or later

PostgreSQL 15 or later

Docker Compose file included for the full stack

INTEGRATION

REST and JSON

Send events, receive decisions

Custody history per shipment

Verification endpoint for auditors

API surface

ENDPOINT	PURPOSE
POST /v1/entity-types	Register an entity type (e.g. Shipment, Vendor), its JSON Schema and its fraud identifiers
POST /v1/entities	Create a record: a shipment, a vendor, any entity
GET · PUT /v1/entities/:id	Read or update its state (custodian, holds, verified accounts…)
POST /v1/rules	Create a rule; rules are versioned
POST /v1/events/process	Evaluate an event and return the decision, audit id and anchor receipt
GET /v1/decisions?entityId=…	History: every decision for a shipment, vendor or other entity
GET /v1/decisions/:id	Decision detail with every rule evaluation and its reason
GET /v1/audits/:id	Audit record: entity, event and ruleset snapshots plus integrity hash
GET /v1/audits/:id/verify	Recompute the hash and compare it with the on-chain anchor
GET /v1/fraud/cases	Fraud cases by status and pattern (analyst or admin)
GET · PATCH /v1/fraud/cases/:id	A case with every signal behind it; confirm fraud or mark a false positive
GET · POST · PATCH /v1/fraud/watchlist	Block or watch any identity: a driver, vessel, bank account, domain…

Rolling out

1. **Model** your shipments and parties, starting from the reference Shipment schema.
2. **Run in shadow mode**: send events from one lane or terminal and record decisions without acting on them.
3. **Tune** rule thresholds (weight tolerance, stop duration) against real traffic.
4. **Enforce** at the gate: act on DENY, route FLAG to an operator, and turn on anchoring.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

5. **Review fraud cases** as they open, and confirm or clear them so the watchlist reflects what your team knows.

> **Built and tested**
>
> The system described here, including the reference rules and scenario, is covered by 1,683 automated tests, and the reference scenarios (22 custody decisions, 5 fraud decisions across shipments and 9 payment decisions) run end to end against the live services with every decision checked and every record verified on chain.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth

SECTION 10

Trust model and limitations

Tamper evidence is only as strong as its weakest assumption. We state ours plainly so that deployments can be designed around them.

Who takes part in the network

The Omnareth chain accepts records through PACT Protocol's Proof-of-Activity consensus. Anchoring proves that the decision database was not edited after the fact, which is the most common way records are disputed. How far the evidence holds against the operator depends on who takes part in the network: if one operator controls both the database and every participant, the chain does not, by itself, protect against that operator. Deployments that need evidence that holds against the operator should include independent parties in the network.

Garbage in

Omnareth decides on the data it is sent. Seal numbers, driver ids and tracker readings come from your systems and devices. The rules make inconsistencies visible and refuse to proceed on them; they cannot detect a lie that is consistent everywhere.

State changes come from your system of record

Omnareth does not move custody on its own; your system updates the shipment after an allowed transfer. The integration must update state only on ALLOW, which the reference integration does.

Fraud memory is local to a deployment

Omnareth recognizes repeat fraud across the shipments and events it sees. It does not share watchlists or signals with other organizations, so a crew active elsewhere is unknown until it is seen here or an analyst adds it.

Memory depends on stable identifiers

Cross-shipment detection follows driver, truck, party, container and release-order identifiers as your systems report them. An impostor who presents a new driver ID each time is still refused by the custody rules, but not recognized as the same person. Detectors use fixed thresholds and open cases for human review; they do not decide fraud on their own.

SUMMARY

Conclusion

Cargo fraud by deception works because each gate decides with part of the picture, and because the record of what happened can be rewritten afterwards. Omnareth gives every gate the whole custody record, refuses anything that does not fit it, explains why, remembers who tried, and seals each decision so it can be proven later. It does so in milliseconds, beside the systems you already run.

© 2026 Habify Labs, Inc. All Rights Reserved. · Omnareth