OMNARETH

TECHNICAL DATASHEET

Omnareth Decision Service

Specifications, interfaces, performance and deployment

Omnareth is a fraud prevention and decision service. It evaluates events against the state of the record they concern (a shipment, a vendor, any entity you define), explicit rules and what it knows about everyone involved, returns an explainable decision, and records tamper-evident evidence of every decision.

Decision model

ITEM	SPECIFICATION
Inputs	Events of any type referencing an entity of any type. Reference sets: cargo (`CUSTODY_HANDOFF`, `CARGO_RELEASE`, `TRACKING_UPDATE`, `VESSEL_LOAD`, `VESSEL_DISCHARGE`, `VOYAGE_UPDATE`) payments (`VENDOR_BANK_CHANGE`, `PAYMENT_REQUEST`), insurance (`CLAIM_SUBMITTED`, `PAYOUT_REQUEST`), refunds (`REFUND_REQUEST`), lending (`APPLICATION_SUBMITTED`, `DISBURSEMENT_REQUEST`), healthcare (`CLAIM_SUBMITTED`), telecom (`SIM_SWAP_REQUEST`, `PORT_OUT_REQUEST`, `DEVICE_ORDER`) benefits (`BENEFIT_CLAIM`, `PAYMENT_UPDATE`) cybersecurity (`LOGIN_ATTEMPT`, `SECURITY_CHANGE`, `PRIVILEGED_ACTION`) and identity verification (`DOCUMENT_CHECK`, `SELFIE_CHECK`)
Outcomes	ALLOW · ALLOW_WITH_FLAGS · DENY
Per rule	Result (PASS, FAIL, NOT_APPLICABLE), code, message, remediation, severity (LOW to CRITICAL), evaluation time
Precedence	Any DENY denies; otherwise any FLAG gives ALLOW_WITH_FLAGS; otherwise ALLOW
Determinism	Pure evaluation: identical event, entity and rules always give an identical decision

Rule language

ITEM	SPECIFICATION
Format	JSON: name, description, entity and event type scope, optional jurisdiction and effective dates, severity, condition, consequence, tags
Logic	`and`, `or`, `not`, nested to any depth
Operators	`==` `!=` `<` `<=` `>` `>=` in not_in contains not_contains starts_with ends_with matches exists not_exists
Comparison targets	A literal `value`, or another field via `valueField` (e.g. `entity.sealNumber`)

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

ITEM	SPECIFICATION
Data in scope	Event payload, the referenced entity under `entity.`, and evaluation context under `context.` (including `context.fraud`)
Responsibility	Optional `consequence.implicates`: the event fields (e.g. `release.driverId`) held responsible when the rule fires
Lifecycle	Rules are versioned; status DRAFT, ACTIVE, INACTIVE or DEPRECATED

```
{ "type": "compare", "field": "release.releaseOrderNumber",
  "operator": "!=", "valueField": "entity.releaseOrderNumber" }
```

Entities

ITEM	SPECIFICATION
Schema	Each entity type is defined by a JSON Schema; entities are validated on create and update
Reference models	Shipment: seal and container, custodian, authorized custodians and consignees, release order, dispatched drivers, customs, holds, weight tolerance; for sea, vessel IMO, ports of loading and discharge, port rotation and bill of lading. Vendor: verified email domain, verified bank accounts, invoice limit, paid invoices. Policy: insured parties, covered losses, limit, payout accounts, claimed incidents. Order: customer, payment method, delivery, return label and receipt. LoanApplication: affordability, income tolerance, status, verified accounts. Member: eligibility, covered services, prior authorizations, paid services. Subscriber: status, recent SIM swap, verified addresses, device limit. BenefitCase: status, weekly benefit, paid weeks, verified accounts. UserAccount: status, known devices, granted roles, export limit
Updates	Your system of record updates state after an allowed transfer (`PUT /v1/entities/:id`)

Fraud prevention

ITEM	SPECIFICATION
Fraud identifiers	Declared per entity type under `fraud.identifiers`: event field path, identifier type (any upper-case token, e.g. `DRIVER`, `VESSEL`, `BANK_ACCOUNT`), role `SUBJECT` or `CONTEXT`, and optional `singleUse` (one entity only, e.g. a bill of lading) or `hopping` (scheme across entities, e.g. one bank account on many vendors). Types without a declaration use the cargo defaults

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

ITEM	SPECIFICATION
Fraud context	`context.fraud` computed by the server for every event: watchlist matches (`blocked`, `watched`, `matches`), `priorImplicatedDenials` (30 days), `referenceOtherEntities` (single-use references seen on other entities), `actorImplicatedEntities24h` (hopping actors). Client-supplied values are replaced; the context is sealed in the audit record
Reference fraud rules	Cargo: WATCHLIST_BLOCK (DENY) · WATCHLIST_MATCH (FLAG) · REPEAT_OFFENDER (FLAG) · REFERENCE_REUSE (FLAG) · DRIVER_HOPPING (FLAG). Payments, insurance, refunds, lending, healthcare, telecom, benefits, cybersecurity and identity verification: the same watchlist, repeat-offender and hopping checks for their own identifiers, plus reuse checks for police reports, return labels, national IDs, prior authorizations, SIMs, session tokens, identity documents and faces
Detectors	Repeat offender: any identity (driver, vessel, party, bank account…) implicated in 2 denied events within 30 days. Reference reuse: a single-use reference (release order, bill of lading) presented for 2 or more entities. Actor hopping: an actor (a driver, a bank account) implicated in denials on 3 or more entities within 24 hours. Burst: 3 or more denied events at one place within 60 minutes
Cases	One open case per pattern and subject, updated by later events; keeps the latest 100 shipments and decisions and a count of attempts. Statuses OPEN, INVESTIGATING, CONFIRMED_FRAUD, FALSE_POSITIVE, CLOSED
Watchlist	BLOCK or WATCH per identifier, with optional expiry. Sources: analyst, confirmed case, detector (90-day watch). Analyst entries always take precedence
Access	Fraud API requires an `admin` or `fraud_analyst` account

Evidence

ITEM	SPECIFICATION
Audit record	Entity snapshot, event snapshot, ruleset snapshot and context (including fraud context) for every decision
Integrity hash	SHA-256 over canonical JSON (sorted keys), stable across database storage
Anchoring	Decision id, event id, integrity hash and status written to the Omnareth chain; receipt (transaction hash, block) stored with the decision. Never blocks a decision
Verification	`GET /v1/audits/:id/verify` recomputes the hash and compares hash, decision id and status with the anchor
History	`GET /v1/decisions?entityId=…` returns every decision for a shipment, vendor or other entity

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

API

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…

Performance

MEASUREMENT	P50	P95	P99
Rule evaluation, 34 rules	22 µs	38 µs	71 µs
API decision, fraud-checked, stored and audited	75 ms	93 ms	117 ms
The same, waiting for the PACT commit	531 ms	685 ms	687 ms

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). Rule evaluation throughput ~46,000 per second on one core. 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.

Deployment

ITEM	SPECIFICATION
Services	Core service (port 3001) · PACT validators (`pact node` or `pact multi`, one per operator)
Runtime	Node.js 20.12 or later · PostgreSQL 15 or later

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

ITEM	SPECIFICATION
Packaging	Docker Compose file for the full stack; services also run directly with `pnpm start`
Chain storage	PostgreSQL (`PACT_DATABASE_URL`), a local file, or memory for tests; stored blocks are re-verified at start-up
Consensus	PACT Protocol (Proof-of-Activity), developed by Habify Labs, Inc. Independent parties in the network recommended where evidence must hold against the operator
Access control	Bearer token on every `/v1` route except sign-in, health and the public pilot form. Roles: `admin` (rules, entity types, everything), `integration` (send events, keep entities in sync, read decisions and evidence), `fraud_analyst` (read; work cases and the watchlist). Any other account has no access; self-registration is off

Configuration

VARIABLE	PURPOSE
`DATABASE_URL`	PostgreSQL connection for the core service
`CHAIN_NODE_URL`	Enables anchoring to the given chain node
`CHAIN_AUTO_MINE`	Seal each anchor into a block immediately (default) or leave it pending
`OMNARETH_CHAIN_DATA_FILE`	Persist chain blocks to this file
`JWT_SECRET`	Signing secret for API access tokens: wherever authentication is required, at least 32 characters and not a published default, or the service does not start
`JWT_ISSUER`, `JWT_AUDIENCE`	Claims set on and required of every access token
`AUTH_REQUIRED`	Require tokens regardless of `NODE_ENV` (required unless `NODE_ENV` is development or test)
`ALLOW_SELF_REGISTRATION`	Open sign-up where authentication is required; new accounts get no role
`RATE_LIMIT_MAX_REQUESTS`, `RATE_LIMIT_WINDOW_MS`	Request rate limit per window

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