zapier.georgelambert.org · Markdown indexes
VERAE TIME × ZAPIER

NATS subjects

site/docs/02-architecture/nats-subjects.md

NATS Subject Topology

All subjects are prefixed with `verae.zapier.` to isolate this platform from other Verae messaging.

Streams

Stream	Subjects	Retention	Purpose
`ZAPIER_JOBS`	`verae.zapier.jobs.watch`	Work queue	Poll Verae for job status
`ZAPIER_EVENTS`	`verae.zapier.jobs.events`	Limits (time)	Terminal job outcomes
`ZAPIER_WEBHOOKS`	`verae.zapier.webhooks.deliver`	Work queue	POST to Zapier hook URLs
`ZAPIER_USAGE` (optional)	`verae.zapier.usage`	Limits	Billing export

Subjects

`verae.zapier.jobs.watch`

Published by: HTTP edge after successful `POST /api/timestamp` (or batch item).
Consumed by: `job-poller` durable consumer (queue group).

Payload

Field	Type	Required	Description
`tenantId`	string	yes	Owning tenant
`jobId`	string	yes	Verae job id
`tokenRef`	string	preferred	Opaque ref to resolve Verae credentials (avoid raw JWT)
`veraeToken`	string	discouraged	Only if tokenRef unavailable; redacted in logs
`enqueuedAt`	ISO-8601	yes	Enqueue time
`attempt`	number	yes	Delivery attempt (0-based)
`maxAttempts`	number	yes	Stop after this many polls
`intervalMs`	number	yes	Suggested delay between polls
`traceId`	string	yes	Correlation id for debug

`verae.zapier.jobs.events`

Published by: Job poller when status is `completed`, `failed`, or `timeout`.
Consumed by: Event router → webhook enqueue; optional waiters on HTTP edge.

Payload

Field	Type	Required	Description
`event`	string	yes	`timestamp.completed` | `timestamp.failed` | `timestamp.timeout`
`tenantId`	string	yes	Tenant id
`jobId`	string	yes	Job id
`status`	object	yes	Verae `StatusResponse` shape (or synthetic timeout)
`traceId`	string	yes	Correlation id
`emittedAt`	ISO-8601	yes	Event time

`verae.zapier.webhooks.deliver`

Published by: Event router for each matching subscription.
Consumed by: `webhook-deliver` queue group.

Payload

Field	Type	Required	Description
`hookId`	string	yes	Stored subscription id
`tenantId`	string	yes	Tenant
`targetUrl`	string	yes	Zapier REST Hook URL
`event`	string	yes	Event name
`payload`	object	yes	Body POSTed to Zapier
`attempt`	number	yes	Attempt count
`traceId`	string	yes	Correlation id

`verae.zapier.usage` (optional)

Field	Type	Description
`tenantId`	string	Tenant
`action`	string	`timestamp` | `verify` | `status` | …
`amount`	number	Increment
`at`	ISO-8601	Timestamp

Consumers

Name	Stream	Mode	Notes
`job-poller`	`ZAPIER_JOBS`	Pull, queue	Nak with delay when still pending
`event-webhook-router`	`ZAPIER_EVENTS`	Push/pull	Fan-out to deliver subjects
`webhook-deliver`	`ZAPIER_WEBHOOKS`	Pull, queue	HTTP POST with backoff
`usage-writer`	`ZAPIER_USAGE`	Optional	Persist metering

Ack semantics

Situation	Action
Job still `pending`	`Nak` with delay ≈ `intervalMs` or republish with `attempt+1`
Job terminal	Publish event, `Ack` watch message
Webhook HTTP 2xx	`Ack`
Webhook HTTP 5xx / network	`Nak` / redelivery until `max_deliver`
Poison message	Term after max_deliver; write DLQ log with `DEBUG_VERAE=webhooks`

Security rules

1. Prefer `tokenRef` over embedding Verae JWTs in messages.
2. NATS must use private network + auth (Phase 15: mTLS).
3. Debug logs must redact tokens and `targetUrl` query secrets if any.
4. Treat `targetUrl` as untrusted egress (timeouts, size limits, SSRF allowlist later).