zapier.georgelambert.org · Markdown indexes
VERAE TIME × ZAPIER
modules and nats
site/docs-master/modules-and-nats.md

Modules and NATS message contracts

Master map of every service in the Zapier ? middleware ? chain ? WORM-archive path.

Zapier cloud **never** connects to NATS. Internal services do: middleware workers, archives, account-balance, zappier-edge billing, CS, sales, and accounting.

Who talks to whom (HTTPS vs NATS)

```
[Zapier Platform app]  HTTPS  [zappier-edge]  HTTPS  [middleware-http]
        |                     | NATS verae.billing.*          |
        |                     v                               |
[customer portal]      [account-balance]                NATS JetStream
[CS :3011]             queue account-balance            (NS1 loopback :4222)
[sales :3012]                 ^                               |
[accounting :3013]            | request-reply                 |
                              +-------------------------------+
          +------------------+------------------+-------------+
          |                  |                  |
   [job-poller]      [webhook-deliver]   [archive-aggregator]
          |                                     |
          | HTTPS                               | NATS query/reply
   [verae-chain-client]                         |
                                         [archive-worm × N]
                                         bloom miss = silence
```

Module catalog

Module	Repo (Forgejo)	Runtime	Listens	Sends
zapier-platform-app	`verae-zapier-app`	Zapier cloud	User Zap steps	HTTPS to zappier-edge
zappier-edge	`zappier-edge`	Public HTTPS :3000	Zapier, portal, admin	HTTPS to middleware; NATS billing after authz
access-authz	`verae-access-authz`	:3020 + NATS	`verae.access.authz.check`	allow/deny
access-web	`verae-access-web`	:3021	customer browser	billing statement/reload
access-api	`verae-access-api`	:3022	x-api-key	statement; HTTPS to middleware
access-leaf	`verae-access-leaf`	:3023	S2S leaf	archive/jobs only
access-zapier	`verae-access-zapier`	:3024	Zapier HTTPS	jobs.watch after authz
account-balance	`zappier-account-balance`	:3010 + NATS	`verae.billing.*`	statement/adjust replies
customer-service	`zappier-customer-service`	:3011	CS staff HTTP	NATS balance.adjust / statement.get
sales-pricing	`zappier-sales-pricing`	:3012	Sales HTTP	NATS statement.get; HTTPS to edge for multipliers
accounting-export	`zappier-accounting-export`	:3013	Accounting HTTP	reads invoices; NATS statement

Module	Repo (Forgejo)	Runtime	Listens	Sends
middleware-http	`verae-middleware`	Public HTTPS :3100	zappier-edge	NATS jobs.watch; HTTP to chain; wait on jobs.events
job-poller	`verae-job-poller`	Worker	`verae.zapier.jobs.watch`	`verae.zapier.jobs.events`; HTTP GET chain status
webhook-deliver	`verae-webhook-deliver`	Worker	`verae.zapier.webhooks.deliver` + events	HTTPS POST Zapier REST Hook
request-splitter	`verae-request-splitter`	In middleware	HTTP body / multipart	chain hash; `verae.archive.put`
archive-aggregator	`verae-archive-aggregator`	Worker / in wait	`includeAttached` on wait	`verae.archive.query`; reads `verae.archive.reply.<id>`
archive-worm	`verae-archive-worm`	N copies	`verae.archive.query`, `verae.archive.put`	`verae.archive.reply.<id>` if bloom hits
tree-node	`verae-tree-node`	N copies (WORM role)	`verae.archive.query`, `verae.archive.put` kind `tree`	`verae.archive.reply.<id>` if bloom hits
zapier-simulator	`verae-zapier-simulator`	Local HTTP :3847	operator browser	in-process replay of all addresses
zapier-user-docs	`zapier-user-docs`	Static	—	catalog `/user-docs/`
verae-chain-client	`verae-chain-client`	Library	—	HTTPS `api.veraetime.net` or MOCK
fleet	`verae-fleet`	Local HTTP :3850	operator	keepFloor + SSH hosts

Module	Repo (Forgejo)	Runtime	Listens	Sends
staff-iam	`verae-staff-iam`	:3028	staff login	none (HTTP)
keep	`verae-keep`	:3860 / :3861	host supervisor	none
bootstrap	`verae-bootstrap`	clone/ install	—	—
overview	`overview`	Static	—	high-level map
nats-process	`verae-nats-process`	Template worker	`verae.example.process.in`	`.out` / `.reply.*`
docs-master	`zapier-docs-master`	Static	—	published on zapier.georgelambert.org

Addresses (subjects)

Address	Kind	Payload (required fields)
`verae.zapier.jobs.watch`	JetStream work queue	`tenantId`, `veraeUserId`, `jobId`, `tokenRef`, `enqueuedAt`, `attempt`, `maxAttempts`, `intervalMs`, `traceId`
`verae.zapier.jobs.events`	JetStream events	`event` (`timestamp.completed|failed|timeout`), `tenantId`, `jobId`, `status`, `traceId`, `emittedAt`
`verae.zapier.webhooks.deliver`	JetStream work queue	`hookId`, `tenantId`, `targetUrl`, `event`, `payload`, `attempt`, `traceId`
`verae.zapier.usage`	optional	`tenantId`, `action`, `amount`, `at`
`verae.billing.statement.get`	request-reply	`customerId` → prepaid, credits, usage, payments
`verae.billing.balance.adjust`	request-reply	`customerId`, `cents`, `reason`, `agent`
`verae.billing.usage.recorded`	pub	meter events from zappier-edge
`verae.billing.payment.recorded`	pub	portal reload / invoice paid
`verae.billing.credit.applied`	pub	CS goodwill
`verae.access.authz.check`	request-reply	`{ plane, subject, principal }` → `{ allow, reason }`
`verae.access.web.billing.statement.get`	ingress	web plane only; mapped after authz
`verae.access.api.*`	ingress	customer API plane
`verae.access.zapier.*`	ingress	Zapier plane
`verae.access.leaf.in`	ingress	S2S leaf; never billing
`verae.archive.put`	JetStream	`sha256`, `tenantId`, `kind` (`publicMeta|privateMeta|file|tree`), `record`, `traceId`
`verae.archive.query`	pub to **all** archives	`correlationId`, `sha256`, `tenantId`, `kinds[]`, `traceId`
`verae.archive.reply.<correlationId>`	replies	

Address	Kind	Payload (required fields)
		`archiveId`, `sha256`, `records[]`, `traceId`

Completed job JSON (back to Zapier)

See `docs/02-architecture/archive-nats.md`. Seal receipts from chain; extra receipts and `files[]` from archive aggregation when `includeAttached` is true.