

zapier.georgelambert.org · Markdown indexes

VERAE TIME × ZAPIER

modules and nats

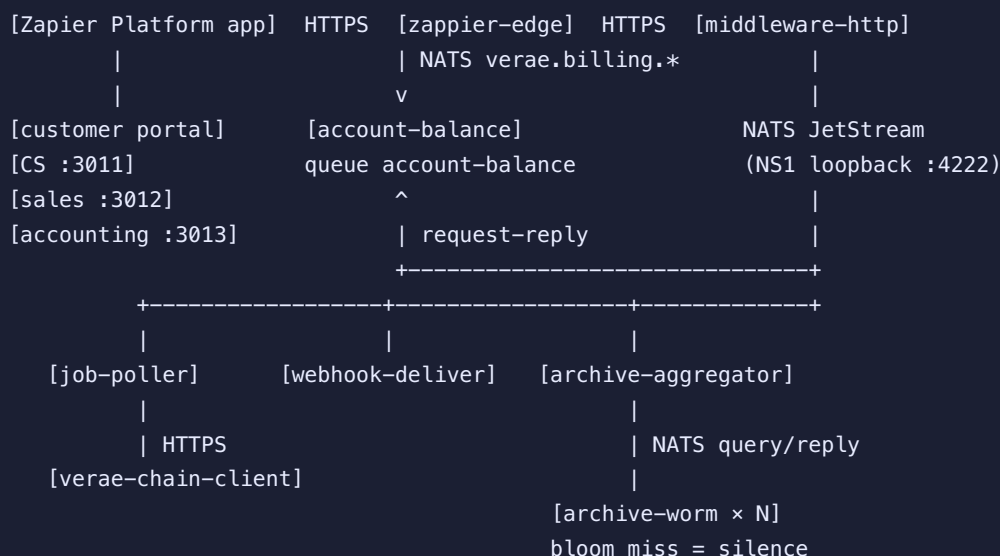
site/packages/docs-master/modules-and-nats.md

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, zapier-edge billing, CS, sales, and accounting.

Who talks to whom (HTTPS vs NATS)



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