Edit on GitHub

NATS Introduction

NATS is an open-source, cloud-native messaging system. In addition to functioning as the “nervous system” for the Apcera platform, companies like Baidu, Siemens, VMware, HTC, and Ericsson rely on NATS for its highly performant and resilient messaging capabilities.

NATS server

NATS provides a lightweight server that is written in the Go programming language. Apcera actively maintains and supports the NATS server source code, binary distributions, and Docker image.

NATS clients

There are several client libraries for NATS. Apcera actively maintains and supports the Go, Node, Ruby, Java, C, C# and NGINX C clients, and there are several community-provided clients.

You can write your own client in any language you choose. NATS provides a simple, text-based protocol that makes writing clients a breeze.

NATS design goals

The core principles underlying NATS are performance, scalability, and ease-of-use. Based on these principles, NATS is designed to be:

- Highly performant (fast)
- Always on and available (dial tone)
- Extremely lightweight (small footprint)
- At most once delivery (fire and forget)
- Support for various messaging models and use cases (flexible)

NATS use cases

NATS is a fire-and-forget messaging system designed to natively support modern cloud architectures. Because complexity does not scale, NATS is designed to be easy to use and implement.

Some of the types of use cases that are ideal for NATS include:

- Addressing, discovery
- Command and control (control plane)
- Load balancing
- N-way scalability
- Location transparency
- Fault tolerance

NATS philosophy holds that high levels of quality-of-service should be built into the client. Only request-reply is built in. NATS does not provide:

- Persistence
- Transactions
- Enhanced delivery modes
- Enterprise queueing

NATS messaging models

NATS supports various messaging models, including:

- Publish Subscribe
- Request Reply
- Queueing

NATS features

NATS provides the following unique features:

- Pure pub sub
  - Never assumes the audience.
  - Always “on” dial tone.
- Clustered mode server
  - NATS servers can be clustered together.
  - Distributed queueing across clusters.
  - Cluster-aware clients.
- Auto-pruning of subscribers
  - To support scaling, NATS provides for auto-pruning of client connections.
  - If a client app is slow consuming messages, NATS will cut off the client.
  - If a client is not responsive within the ping-pong interval, the server cuts it off.
  - Clients implement retry logic.
- Text-based protocol
  - Makes it easy to get started with new clients.
  - Does not affect server performance.
  - Can Telnet directly to the server and send messages across the wire.

NATS FAQs

See our FAQ page.

NATS is open-source software (View License), as is this site.

×

NATS License (The MIT License)

Close