NATS Cluster Protocol

- NATS Cluster Protocol
- NATS Cluster protocol conventions
- NATS Cluster protocol messages
- INFO
  - Description
  - Syntax
  - Example
- CONNECT
  - Description
  - Syntax
  - Example
- RS+
  - Description
  - Syntax
- RS-
  - Description
  - Syntax
- RMSG
  - Description
  - Syntax
- PING/PONG
  - Description
  - Syntax
- -ERR
  - Description

NATS Cluster Protocol

The NATS server clustering protocol describes the protocols passed between NATS servers within a cluster to share accounts, subscriptions, forward messages, and share cluster topology regarding new servers. It is a simple text-based protocol. Servers communicate with each other through a regular TCP/IP or TLS socket using a small set of protocol operations that are terminated by newline.

The NATS server implements a zero allocation byte parser that is fast and efficient.

The NATS cluster protocol is very similar to that of the NATS client protocol. In the context of a cluster, it can be helpful to visualize a server being a proxy operating on behalf of its connected clients, subscribing, unsubscribing, sending and receiving messages.

NATS Cluster protocol conventions

Subject names and wildcards: The NATS cluster protocol has the same features and restrictions as the client with respect to subject names and wildcards. Clients are bound to a single account, however the cluster protocol handles all accounts.

Field Delimiters: The fields of NATS protocol messages are delimited by whitespace characters ' `` `'\(space\) or`\t` (tab). Multiple whitespace characters will be treated as a single field delimiter.

Newlines: Like other text-based protocols, NATS uses `CR` followed by `LF` (`CR+LF`, `\r`, `0x0D0A`) to terminate protocol messages. This newline sequence is also used to mark the beginning of the actual message payload in a `RMSG` protocol message.

NATS Cluster protocol messages

The following table briefly describes the NATS cluster protocol messages. As in the client protocol, the NATS protocol operation names are case insensitive, thus `SUB foo 1\r` and `sub foo 1\r` are equivalent.

Click the name to see more detailed information, including syntax:

OP Name	Sent By	Description
`INFO`	All Servers	Sent after initial TCP/IP connection and to update cluster knowledge
`CONNECT`	All Servers	Sent to establish a route

OP Name	Sent By	Description
`RS+`	All Servers	Subscribes to a subject for a given account on behalf of interested clients.
`RS-`	All Servers	Unsubscribe (or auto-unsubscribe) from subject for a given account.
`RMSG`	Origin Server	Delivers a message for a given subject and account to another server.
`PING`	All Servers	PING keep-alive message
`PONG`	All Servers	PONG keep-alive response
`-ERR`	All Servers	Indicates a protocol error. May cause the remote server to disconnect.

The following sections explain each protocol message.

INFO

Description

As soon as the server accepts a connection from another server, it will send information about itself and the configuration and security requirements that are necessary for the other server to successfully authenticate with the server and exchange messages.

The connecting server also sends an `INFO` message. The accepting server will add an `ip` field containing the address and port of the connecting server, and forward the new server's `INFO` message to all servers it is routed to.

Any servers in a cluster receiving an `INFO` message with an `ip` field will attempt to connect to the server at that address, unless already connected. This propagation of `INFO` messages on behalf of a connecting server provides automatic discovery of new servers joining a cluster.

Syntax

`INFO {["option_name":option_value],...}`

The valid options are as follows:

- `server_id`: The unique identifier of the NATS server
- `version`: The version of the NATS server

- `go`: The version of golang the NATS server was built with
- `host`: The host specified in the cluster parameter/options
- `port`: The port number specified in the cluster parameter/options
- `auth_required`: If this is set, then the server should try to authenticate upon connect.
- `tls_required`: If this is set, then the server must authenticate using TLS.
- `max_payload`: Maximum payload size that the server will accept.
- `connect_urls` : A list of server urls that a client can connect to.
- `ip`: Optional route connection address of a server, `nats-route://<hostname>:<port>`

Example

Below is an example of an `INFO` string received by a NATS server, with the `ip` field.

```
INFO
{"server_id":"KP19vTlB417XElnv8kKaC5","version":"2.0.0","go":"","host":"localhost","port":5222
,"auth_required":false,"tls_required":false,"tls_verify":false,"max_payload":1048576,"ip":"nat
s-route://127.0.0.1:5222/","connect_urls":["localhost:4222"]}
```

CONNECT

Description

The `CONNECT` message is analogous to the `INFO` message. Once the NATS server has established a TCP/IP socket connection with another server, and an `INFO` message has been received, the server will send a `CONNECT` message to provide more information about the current connection as well as security information.

Syntax

`CONNECT {["option_name":option_value],...}`

The valid options are as follows:

- `tls_required`: Indicates whether the server requires an SSL connection.
- `auth_token`: Authorization token
- `user`: Connection username (if `auth_required` is set)
- `pass`: Connection password (if `auth_required` is set)
- `name`: Generated Server Name
- `lang`: The implementation language of the server (go).

- `version`: The version of the server.

Example

Here is an example from the default string from a server.

```
CONNECT {"tls_required":false,"name":"wt0vffeQyoDGMVBC2aKX0b"}\r
```

RS+

Description

`RS+` initiates a subscription to a subject on on a given account, optionally with a distributed queue group name and weighting factor. Note that queue subscriptions will use RS+ for increases and decreases to queue weight except when the weighting factor is 0.

Syntax

Subscription: `RS+ <account> <subject>\r`

Queue Subscription: `RS+ <account> <subject> <queue> <weight>\r`

where:

- `account`: The account associated with the subject interest
- `subject`: The subject
- `queue`: Optional queue group name
- `weight`: Optional queue group weight representing how much interest/subscribers

RS-

Description

`RS-` unsubcribes from the specified subject on the given account. It is sent by a server when it no longer has interest in a given subject.

Syntax

Subscription: `RS- <account> <subject>\r`

where:

- `account`: The account associated with the subject interest
- `subject`: The subject

RMSG

Description

The `RMSG` protocol message delivers a message to another server.

Syntax

`RMSG <account> <subject> [reply-to] <#bytes>\r\n[payload]\r`

where:

- `account`: The account associated with the subject interest
- `subject`: Subject name this message was received on
- `reply-to`: The optional reply subject
- `#bytes`: Size of the payload in bytes
- `payload`: The message payload data

PING/PONG

Description

`PING` and `PONG` implement a simple keep-alive mechanism between servers. Once two servers establish a connection with each other, the NATS server will continuously send `PING` messages to other servers at a configurable interval. If another server fails to respond with a `PONG` message within the configured response interval, the server will terminate its connection. If your connection stays idle for too long, it is cut off.

If the another server sends a ping request, a server will reply with a pong message to notify the other server that it is still present.

Syntax

`PING\r` `PONG\r`

-ERR

Description

The `-ERR` message is used by the server to indicate a protocol, authorization, or other runtime connection error to another server. Most of these errors result in the remote server closing the connection.

□□□ #3
□ cocobook-sync □□□ 2026-10-28 11:15:01 UTC
□ cocobook-sync □□□ 2026-10-29 01:42:41 UTC