NATS Server Clients

- Embedding NATS
- Installing the nats CLI Tool
- Testing your setup
- Testing Against a Remote Server
  - NATS URLs
  - Example

A NATS client is an application making a connection to one of the nats servers pointed to by its connection URL, and uses a credential file to authenticate and indicate its authorization to the server and the whole NATS infrastructure.

The nats-server doesn't come bundled with any clients, but its companion is the `nats` CLI tool that you should install (even if you don't intend to run your own servers) as it is the best tool to use to test, monitor, manage and generally interact with a NATS infrastructure (regardless of that infrastructure being an isolated local server, a leaf node server, a cluster or even a global super-cluster).

Other NATS client tools to know about are the `nsc` CLI tool (to manage accounts attributes and user JWT tokens) and the 'nk' tool (and library) to manage Nkeys.

Also, most client libraries come with sample programs that allow you to publish, subscribe, send requests and reply messages.

Embedding NATS

If your application is in Go, and if it fits your use case and deployment scenarios, you can even embed a NATS server inside your application.

Embedding NATS in Go

Installing the `nats` CLI Tool

Please refer to the installation section in the readme.

Testing your setup

Open a terminal and start a nats-server:

```
nats-server
```

```
[45695] 2021/09/29 02:22:53.570667 [INF] Starting nats-server
[45695] 2021/09/29 02:22:53.570796 [INF]   Version:  2.6.1
[45695] 2021/09/29 02:22:53.570799 [INF]   Git:      [not set]
[45695] 2021/09/29 02:22:53.570804 [INF]   Name:
NAAACXGWSD6ZW5KVHOTSGGPU2JCMZUDSMY5GVZZP27DMRPWYINC2X6ZI
[45695] 2021/09/29 02:22:53.570807 [INF]   ID:
NAAACXGWSD6ZW5KVHOTSGGPU2JCMZUDSMY5GVZZP27DMRPWYINC2X6ZI
[45695] 2021/09/29 02:22:53.571747 [INF] Listening for client connections on 0.0.0.0:4222
[45695] 2021/09/29 02:22:53.572051 [INF] Server is ready
```

On another terminal session first check the connection to the server

```
nats server check connection -s nats://0.0.0.0:4222
```

```
OK Connection OK:connected to nats://127.0.0.1:4222 in 790.28µs OK:rtt time 69.896µs OK:round
trip took 0.000102s | connect_time=0.0008s;0.5000;1.0000 rtt=0.0001s;0.5000;1.0000
request_time=0.0001s;0.5000;1.0000
```

Next, start a subscriber using the `nats` CLI tool:

```
nats subscribe ">" -s nats://0.0.0.0:4222
```

Note that when the client connected, the server didn't log anything interesting because server output is relatively quiet unless something interesting happens.

To make the server output more lively, you can specify the `-V` flag to enable logging of server protocol tracing messages. Go ahead and `<ctrl>+c` the process running the server, and restart the server with the `-V` flag:

```
nats-server -V
```

```
[45703] 2021/09/29 02:23:05.189377 [INF] Starting nats-server
[45703] 2021/09/29 02:23:05.189489 [INF]   Version:  2.6.1
[45703] 2021/09/29 02:23:05.189493 [INF]   Git:      [not set]
[45703] 2021/09/29 02:23:05.189497 [INF]   Name:
NAIBOVQLOZSDIUFQYZOQUGV3PNZUT66D4WF5MKS2G7N423UGJDH2DFWG
[45703] 2021/09/29 02:23:05.189500 [INF]   ID:
NAIBOVQLOZSDIUFQYZOQUGV3PNZUT66D4WF5MKS2G7N423UGJDH2DFWG
[45703] 2021/09/29 02:23:05.190236 [INF] Listening for client connections on 0.0.0.0:4222
[45703] 2021/09/29 02:23:05.190504 [INF] Server is ready
[45703] 2021/09/29 02:23:07.111053 [TRC] 127.0.0.1:51653 - cid:4 - <<- [CONNECT
{"verbose":false,"pedantic":false,"tls_required":false,"name":"NATS CLI Version
0.0.26","lang":"go","version":"1.12.0","protocol":1,"echo":true,"headers":true,"no_responders"
:true}]
[45703] 2021/09/29 02:23:07.111282 [TRC] 127.0.0.1:51653 - cid:4 - "v1.12.0:go:NATS CLI
Version 0.0.26" - <<- [PING]
[45703] 2021/09/29 02:23:07.111301 [TRC] 127.0.0.1:51653 - cid:4 - "v1.12.0:go:NATS CLI
Version 0.0.26" - ->> [PONG]
[45703] 2021/09/29 02:23:07.111632 [TRC] 127.0.0.1:51653 - cid:4 - "v1.12.0:go:NATS CLI
Version 0.0.26" - <<- [SUB >  1]
[45703] 2021/09/29 02:23:07.111679 [TRC] 127.0.0.1:51653 - cid:4 - "v1.12.0:go:NATS CLI
Version 0.0.26" - <<- [PING]
[45703] 2021/09/29 02:23:07.111689 [TRC] 127.0.0.1:51653 - cid:4 - "v1.12.0:go:NATS CLI
Version 0.0.26" - ->> [PONG]
```

If you had created a subscriber, you should notice output on the subscriber telling you that it disconnected, and reconnected. The server output above is more interesting. You can see the subscriber send a `CONNECT` protocol message and a `PING` which was responded to by the server with a `PONG`.

> You can learn more about the NATS protocol here, but more interesting than the protocol description is an interactive demo.

On a third terminal, publish your first message:

```
nats pub hello world -s nats://0.0.0.0:4222
```

On the subscriber window you should see:

```
[#1] Received on "hello"
world
```

Testing Against a Remote Server

If the NATS server were running in a different machine or a different port, you'd have to specify that to the client by specifying a *NATS URL* (either in a `nats context` or using the `-s` flag).

NATS URLs

NATS URLs take the form of: `nats://<server>:<port>` and `tls://<server>:<port>`. URLs with a `tls` protocol sport a secured TLS connection.

If you are connecting to a cluster you can specify more than one URL (comma separated). e.g. `nats://localhost:4222,nats://localhost:5222,nats://localhost:6222` if you are running a test cluster of 3 nats servers on your local machine, listening at ports 4222, 5222, and 6222 respectively.

Example

```
nats sub -s nats://server:port ">"
```

If you want to try on a remote server, the NATS team maintains a demo server you can reach at `demo.nats.io`.

```
nats sub -s nats://demo.nats.io ">"
```

□□□ #3
□ cocobook-sync □□□ 2026-10-28 11:13:29 UTC
□ cocobook-sync □□□ 2026-10-29 01:42:37 UTC