

NATS and Docker



- [NATS Server Containerization](#)
 - [Nightly](#)
 - [Usage](#)
 - [Creating a NATS Cluster](#)
 - [Creating a NATS Cluster with Docker Compose](#)
 - [Testing the Clusters](#)
 - [Tutorial](#)
-

NATS Server Containerization

The NATS server is provided as a Docker image on [Docker Hub](#) that you can run using the Docker daemon. The NATS server Docker image is extremely lightweight, coming in under 10 MB in size.

[Synadia](#) actively maintains and supports the NATS server Docker image.

Nightly

The nightly build container can be found [here](#)

Usage

To use the Docker container image, install Docker and pull the public image:

```
docker pull nats
```

Run the NATS server image:

```
docker run nats
```

By default the NATS server exposes multiple ports:

- 4222 is for clients.
- 8222 is an HTTP management port for information reporting.
- 6222 is a routing port for clustering.

The default ports may be customized by providing either a `-p` or `-P` option on the `docker run` command line.

The following steps illustrate how to run a server with the ports exposed on a `docker network`.

First, create the `docker network` 'nats':

```
docker network create nats
```

Then start the server:

```
docker run --name nats --network nats --rm -p 4222:4222 -p 8222:8222 nats --http_port 8222
```

Creating a NATS Cluster

First, run a server with the ports exposed on the 'nats' `docker network`:

```
docker run --name nats --network nats --rm -p 4222:4222 -p 8222:8222 nats --http_port 8222 --cluster_name NATS --cluster nats://0.0.0.0:6222
```

```
[1] 2021/09/28 09:21:56.554756 [INF] Starting nats-server
[1] 2021/09/28 09:21:56.554864 [INF]   Version:  2.6.1
[1] 2021/09/28 09:21:56.554878 [INF]    Git:      [c91f0fe]
[1] 2021/09/28 09:21:56.554894 [INF]   Name:
NDIQLLD2UGGPSAEYBKHW3S2JB2DXIAFHMIWWRUBAX7FC4RTQX4ET2JNQ
[1] 2021/09/28 09:21:56.555001 [INF]    ID:
NDIQLLD2UGGPSAEYBKHW3S2JB2DXIAFHMIWWRUBAX7FC4RTQX4ET2JNQ
[1] 2021/09/28 09:21:56.557658 [INF] Starting http monitor on 0.0.0.0:8222
[1] 2021/09/28 09:21:56.557967 [INF] Listening for client connections on 0.0.0.0:4222
[1] 2021/09/28 09:21:56.559224 [INF] Server is ready
[1] 2021/09/28 09:21:56.559375 [INF] Cluster name is NATS
[1] 2021/09/28 09:21:56.559433 [INF] Listening for route connections on 0.0.0.0:6222
```

Next, start additional servers pointing them to the seed server to cause them to form a cluster:

```
docker run --name nats-1 --network nats --rm nats --cluster_name NATS --cluster
nats://0.0.0.0:6222 --routes=nats://ruser:T0pS3cr3t@nats:6222
docker run --name nats-2 --network nats --rm nats --cluster_name NATS --cluster
nats://0.0.0.0:6222 --routes=nats://ruser:T0pS3cr3t@nats:6222
```

NOTE Since the Docker image protects routes using credentials we need to provide them above.

Extracted [from Docker image configuration](#)

To verify the routes are connected, you can make a request to the monitoring endpoint on `/routez` and confirm that there are now 2 routes:

```
curl http://127.0.0.1:8222/routez
```

```
{
  "server_id": "NDIQLLD2UGGPSAEYBKHW3S2JB2DXIAFHMIWWRUBAX7FC4RTQX4ET2JNQ",
  "now": "2021-09-28T09:22:15.8019785Z",
  "num_routes": 2,
  "routes": [
    {
      "rid": 5,
      "remote_id": "NBRAUY3YSVFYU7BFWI2YF5VPQFG02XCCKAHYZ7ETCMGB3SQY3FDFTY0Q",
      "did_solicit": false,
      "is_configured": false,
      "ip": "172.18.0.3",
      "port": 59092,
      "pending_size": 0,
      "rtt": "1.2505ms",
      "in_msgs": 4,
      "out_msgs": 3,
      "in_bytes": 2714,
      "out_bytes": 1943,
      "subscriptions": 35
    },
    {
      "rid": 6,
      "remote_id": "NA5STTST5GYFCD22M2I3VDJ57LQKOU35ZVWKQY305QRF0PC3RFDIDVJ",
      "did_solicit": false,
      "is_configured": false,
      "ip": "172.18.0.4",
      "port": 47424,
```

```
    "pending_size": 0,  
    "rtt": "1.2008ms",  
    "in_msgs": 4,  
    "out_msgs": 1,  
    "in_bytes": 2930,  
    "out_bytes": 833,  
    "subscriptions": 35  
  }  
]  
}
```

Creating a NATS Cluster with Docker Compose

It is also straightforward to create a cluster using Docker Compose. Below is a simple example that uses a network named 'nats' to create a full mesh cluster.

```
version: "3.5"  
services:  
  nats:  
    image: nats  
    ports:  
      - "8222:8222"  
    command: "--cluster_name NATS --cluster nats://0.0.0.0:6222 --http_port 8222 "  
    networks: ["nats"]  
  nats-1:  
    image: nats  
    command: "--cluster_name NATS --cluster nats://0.0.0.0:6222 --  
routes=nats://ruser:T0pS3cr3t@nats:6222"  
    networks: ["nats"]  
    depends_on: ["nats"]  
  nats-2:  
    image: nats  
    command: "--cluster_name NATS --cluster nats://0.0.0.0:6222 --  
routes=nats://ruser:T0pS3cr3t@nats:6222"  
    networks: ["nats"]  
    depends_on: ["nats"]
```

```
networks:
  nats:
    name: nats
```

Now we use Docker Compose to create the cluster that will be using the 'nats' network:

```
docker-compose -f nats-cluster.yaml up
```

```
[+] Running 3/3
  0 Container xxx_nats_1    Created
  0 Container xxx_nats-1_1  Created
  0 Container xxx_nats-2_1  Created
Attaching to nats-1_1, nats-2_1, nats_1
nats_1      | [1] 2021/09/28 10:42:36.742844 [INF] Starting nats-server
nats_1      | [1] 2021/09/28 10:42:36.742898 [INF]   Version:  2.6.1
nats_1      | [1] 2021/09/28 10:42:36.742913 [INF]   Git:      [c91f0fe]
nats_1      | [1] 2021/09/28 10:42:36.742929 [INF]   Name:
NCZIIQ6QT4KT5K5WBP7H2RRBM4MSYD4C2TVSRZ0ZN57EHX6VTF4EWXAU
nats_1      | [1] 2021/09/28 10:42:36.742954 [INF]   ID:
NCZIIQ6QT4KT5K5WBP7H2RRBM4MSYD4C2TVSRZ0ZN57EHX6VTF4EWXAU
nats_1      | [1] 2021/09/28 10:42:36.745289 [INF] Starting http monitor on 0.0.0.0:8222
nats_1      | [1] 2021/09/28 10:42:36.745737 [INF] Listening for client connections on
0.0.0.0:4222
nats_1      | [1] 2021/09/28 10:42:36.750381 [INF] Server is ready
nats_1      | [1] 2021/09/28 10:42:36.750669 [INF] Cluster name is NATS
nats_1      | [1] 2021/09/28 10:42:36.751444 [INF] Listening for route connections on
0.0.0.0:6222
nats-1_1    | [1] 2021/09/28 10:42:37.709888 [INF] Starting nats-server
nats-1_1    | [1] 2021/09/28 10:42:37.709977 [INF]   Version:  2.6.1
nats-1_1    | [1] 2021/09/28 10:42:37.709999 [INF]   Git:      [c91f0fe]
nats-1_1    | [1] 2021/09/28 10:42:37.710023 [INF]   Name:
NBHTXXY3HYZVPXITYQ73BSDA5CQZINTKYRM23XFI46RWTTUP5TAXQMB
nats-1_1    | [1] 2021/09/28 10:42:37.710042 [INF]   ID:
NBHTXXY3HYZVPXITYQ73BSDA5CQZINTKYRM23XFI46RWTTUP5TAXQMB
nats-1_1    | [1] 2021/09/28 10:42:37.711646 [INF] Listening for client connections on
0.0.0.0:4222
nats-1_1    | [1] 2021/09/28 10:42:37.712197 [INF] Server is ready
nats-1_1    | [1] 2021/09/28 10:42:37.712376 [INF] Cluster name is NATS
nats-1_1    | [1] 2021/09/28 10:42:37.712469 [INF] Listening for route connections on
0.0.0.0:6222
```

```
nats_1      | [1] 2021/09/28 10:42:37.718918 [INF] 172.18.0.4:52950 - rid:4 - Route connection
created
nats-1_1    | [1] 2021/09/28 10:42:37.719906 [INF] 172.18.0.3:6222 - rid:4 - Route connection
created
nats-2_1    | [1] 2021/09/28 10:42:37.731357 [INF] Starting nats-server
nats-2_1    | [1] 2021/09/28 10:42:37.731518 [INF]   Version:  2.6.1
nats-2_1    | [1] 2021/09/28 10:42:37.731531 [INF]   Git:      [c91f0fe]
nats-2_1    | [1] 2021/09/28 10:42:37.731543 [INF]   Name:
NCG6UQ2N3IHE60S76TL46RNZBAPHNUCQSA64FDFHG5US2LLJ0QLD5ZK2
nats-2_1    | [1] 2021/09/28 10:42:37.731554 [INF]   ID:
NCG6UQ2N3IHE60S76TL46RNZBAPHNUCQSA64FDFHG5US2LLJ0QLD5ZK2
nats-2_1    | [1] 2021/09/28 10:42:37.732893 [INF] Listening for client connections on
0.0.0.0:4222
nats-2_1    | [1] 2021/09/28 10:42:37.733431 [INF] Server is ready
nats-2_1    | [1] 2021/09/28 10:42:37.733491 [INF] Cluster name is NATS
nats-2_1    | [1] 2021/09/28 10:42:37.733835 [INF] Listening for route connections on
0.0.0.0:6222
nats_1      | [1] 2021/09/28 10:42:37.740860 [INF] 172.18.0.5:54616 - rid:5 - Route connection
created
nats-2_1    | [1] 2021/09/28 10:42:37.741557 [INF] 172.18.0.3:6222 - rid:4 - Route connection
created
nats-1_1    | [1] 2021/09/28 10:42:37.743981 [INF] 172.18.0.5:6222 - rid:5 - Route connection
created
nats-2_1    | [1] 2021/09/28 10:42:37.744332 [INF] 172.18.0.4:40250 - rid:5 - Route connection
created
```

Testing the Clusters

Now, the following should work: make a subscription on one of the nodes and publish it from another node. You should be able to receive the message without problems.

```
docker run --network nats --rm -it natsio/nats-box
```

Inside the container:

```
nats sub -s nats://nats:4222 hello &
nats pub -s "nats://nats-1:4222" hello first
nats pub -s "nats://nats-2:4222" hello second
```

Stopping the seed node, which received the subscription, should trigger an automatic failover to the other nodes:

```
docker-compose -f nats-cluster.yaml stop nats
```

Output extract:

```
...
16e55f1c4f3c:~# 10:47:28 Disconnected due to: EOF, will attempt reconnect
10:47:28 Disconnected due to: EOF, will attempt reconnect
10:47:28 Reconnected [nats://172.18.0.4:4222]
```

Publishing again will continue to work after the reconnection:

```
nats pub -s "nats://nats-1:4222" hello again
nats pub -s "nats://nats-2:4222" hello again
```

Tutorial

See the [NATS Docker tutorial](#) for more instructions on using the NATS server Docker image.

📄 #3

📄 cocobook-sync 📄	2026-10-28 11:13:21 UTC
📄 cocobook-sync 📄	2026-10-29 01:42:37 UTC