

Docker Swarm



- [Step 1:](#)
 - [Step 2:](#)
 - [Step 3:](#)
 - [And so forth...](#)
-

Step 1:

Create an overlay network for the cluster (in this example, `nats-cluster-example`), and instantiate an initial NATS server.

First create an overlay network:

```
docker network create --driver overlay nats-cluster-example
```

Next instantiate an initial "seed" server for a NATS cluster listening for other servers to join route to it on port 6222:

```
docker service create --network nats-cluster-example --name nats-cluster-node-1 nats:1.0.0 -  
cluster nats://0.0.0.0:6222 -DV
```

Step 2:

The 2nd step is to create another service which connects to the NATS server within the overlay network. Note that we connect to the server at `nats-cluster-node-1`:

```
docker service create --name ruby-nats --network nats-cluster-example wallyqs/ruby-nats:ruby-  
2.3.1-nats-v0.8.0 -e '  
  NATS.on_error do |e|  
    puts "ERROR: #{e}"  
  end  
  NATS.start(:servers => ["nats://nats-cluster-node-1:4222"]) do |nc|
```

```

inbox = NATS.create_inbox
puts "[#{Time.now}] Connected to NATS at #{nc.connected_server}, inbox: #{inbox}"

nc.subscribe(inbox) do |msg, reply|
  puts "[#{Time.now}] Received reply - #{msg}"
end

nc.subscribe("hello") do |msg, reply|
  next if reply == inbox
  puts "[#{Time.now}] Received greeting - #{msg} - #{reply}"
  nc.publish(reply, "world")
end

EM.add_periodic_timer(1) do
  puts "[#{Time.now}] Saying hi (servers in pool: #{nc.server_pool})"
  nc.publish("hello", "hi", inbox)
end
end'

```

Step 3:

Now you can add more nodes to the Swarm cluster via more docker services, referencing the seed server in the `-routes` parameter:

```

docker service create --network nats-cluster-example --name nats-cluster-node-2 nats:1.0.0 -
cluster nats://0.0.0.0:6222 -routes nats://nats-cluster-node-1:6222 -DV

```

In this case, `nats-cluster-node-1` is seeding the rest of the cluster through the autodiscovery feature. Now NATS servers `nats-cluster-node-1` and `nats-cluster-node-2` are clustered together.

Add in more replicas of the subscriber:

```

docker service scale ruby-nats=3

```

Then confirm the distribution on the Docker Swarm cluster:

```

docker service ps ruby-nats

```

ID	NAME	IMAGE	NODE
DESIRED STATE	CURRENT STATE	ERROR	

```
25skxso8honyhuznu15e4989m  ruby-nats.1  wallyqs/ruby-nats:ruby-2.3.1-nats-v0.8.0  node-1
Running          Running 2 minutes ago
0017lut0u3wj153yvp0uxr8yo  ruby-nats.2  wallyqs/ruby-nats:ruby-2.3.1-nats-v0.8.0  node-1
Running          Running 2 minutes ago
2sxl8rw6vm99x622efbdkb96  ruby-nats.3  wallyqs/ruby-nats:ruby-2.3.1-nats-v0.8.0  node-2
Running          Running 2 minutes ago
```

The sample output after adding more NATS server nodes to the cluster, is below - and notice that the client is *dynamically* aware of more nodes being part of the cluster via auto discovery!

```
[2016-08-15 12:51:52 +0000] Saying hi (servers in pool: [{:uri=>#<URI::Generic
nats://10.0.1.3:4222>, :was_connected=>true, :reconnect_attempts=>0}])
[2016-08-15 12:51:53 +0000] Saying hi (servers in pool: [{:uri=>#<URI::Generic
nats://10.0.1.3:4222>, :was_connected=>true, :reconnect_attempts=>0}])
[2016-08-15 12:51:54 +0000] Saying hi (servers in pool: [{:uri=>#<URI::Generic
nats://10.0.1.3:4222>, :was_connected=>true, :reconnect_attempts=>0}])
[2016-08-15 12:51:55 +0000] Saying hi (servers in pool: [{:uri=>#<URI::Generic
nats://10.0.1.3:4222>, :was_connected=>true, :reconnect_attempts=>0}, {:uri=>#<URI::Generic
nats://10.0.1.7:4222>, :reconnect_attempts=>0}, {:uri=>#<URI::Generic nats://10.0.1.6:4222>,
:reconnect_attempts=>0}])
```

Sample output after adding more workers which can reply back (since ignoring own responses):

```
[2016-08-15 16:06:26 +0000] Received reply - world
[2016-08-15 16:06:26 +0000] Received reply - world
[2016-08-15 16:06:27 +0000] Received greeting - hi - _INBOX.b8d8c01753d78e562e4dc561f1
[2016-08-15 16:06:27 +0000] Received greeting - hi - _INBOX.4c35d18701979f8c8ed7e5f6ea
```

And so forth...

From here you can experiment adding to the NATS cluster by simply adding servers with new service names, that route to the seed server `nats-cluster-node-1`. As you've seen above, clients will automatically be updated to know that new servers are available in the cluster.

```
docker service create --network nats-cluster-example --name nats-cluster-node-3 nats:1.0.0 -
cluster nats://0.0.0.0:6222 -routes nats://nats-cluster-node-1:6222 -DV
```

☐☐ #1

☐ cocobook-sync ☐☐ 2026-10-28 11:13:23 UTC

☐ cocobook-sync ☐☐ 2026-10-28 11:13:23 UTC