

Administration



- [Account Level](#)
 - [Creating clustered streams](#)
 - [Viewing Stream Placement and Stats](#)
 - [Forcing Stream and Consumer leader election](#)
 - [System Level](#)
 - [Viewing the cluster state](#)
 - [Forcing Meta Group leader election](#)
 - [Evicting a peer](#)
-

Once a JetStream cluster is operating interactions with the CLI and with `nats` CLI is the same as before. For these examples, lets assume we have a 5 server cluster, n1-n5 in a cluster named C1.

Account Level

Within an account there are operations and reports that show where users data is placed and which allow them some basic interactions with the RAFT system.

Creating clustered streams

When adding a stream using the `nats` CLI the number of replicas will be asked, when you choose a number more than 1, (we suggest 1, 3 or 5), the data will be stored on multiple nodes in your cluster using the RAFT protocol as above.

```
nats stream add ORDERS --replicas 3
```

Example output extract:

```
....  
Information for Stream ORDERS created 2021-02-05T12:07:34+01:00
```

....

Configuration:

....

Replicas: 3

Cluster Information:

Name: C1

Leader: n1-c1

Replica: n4-c1, current, seen 0.07s ago

Replica: n3-c1, current, seen 0.07s ago

Above you can see that the cluster information will be reported in all cases where Stream info is shown such as after add or using `nats stream info`.

Here we have a stream in the NATS cluster `C1`, its current leader is a node `n1-c1` and it has 2 followers - `n4-c1` and `n3-c1`.

The `current` indicates that followers are up to date and have all the messages, here both cluster peers were seen very recently.

The replica count can be edited once configured.

Viewing Stream Placement and Stats

Users can get overall statistics about their streams and also where these streams are placed:

```
nats stream report
```

Obtaining Stream stats

Stream	Consumers	Messages	Bytes	Storage	Lost	Deleted	Cluster
ORDERS	4	0	0 B	File	0	0	n1-c1*, n2-c1, n3-c1
ORDERS_3	4	0	0 B	File	0	0	n1-c1*, n2-c1, n3-c1
ORDERS_4	4	0	0 B	File	0	0	n1-c1*, n2-c1, n3-c1
ORDERS_5	4	0	0 B	File	0	0	n1-c1, n2-c1, n3-c1*
ORDERS_2	4	1,385	13 MiB	File	0	1	n1-c1, n2-c1, n3-c1*
ORDERS_0	4	1,561	14 MiB	File	0	0	n1-c1, n2-c1*, n3-c1

Forcing Stream and Consumer leader election

Every RAFT group has a leader that's elected by the group when needed. Generally there is no reason to interfere with this process, but you might want to trigger a leader change at a convenient time. Leader elections will represent short interruptions to the stream so if you know you will work on a node later it might be worth moving leadership away from it ahead of time.

Moving leadership away from a node does not remove it from the cluster and does not prevent it from becoming a leader again, this is merely a triggered leader election.

```
nats stream cluster step-down ORDERS
```

```
14:32:17 Requesting leader step down of "n1-c1" in a 3 peer RAFT group
14:32:18 New leader elected "n4-c1"
```

```
Information for Stream ORDERS created 2021-02-05T12:07:34+01:00
```

```
...
```

```
Cluster Information:
```

```
    Name: c1
```

```
    Leader: n4-c1
```

```
    Replica: n1-c1, current, seen 0.12s ago
```

```
    Replica: n3-c1, current, seen 0.12s ago
```

The same is true for consumers, `nats consumer cluster step-down ORDERS NEW`.

System Level

Systems users can view state of the Meta Group - but not individual Stream or Consumers.

Viewing the cluster state

We have a high level report of cluster state:

```
nats server report jetstream --user admin --password s3cr3t!
```

```
+-----+
|-----+
|
|                                     JetStream
```

Summary									
Server	Cluster	Streams	Consumers	Messages	Bytes	Memory	File	API Req	API Err
n3-c2	c2	0	0	0	0 B	0 B	0 B	1	
0									
n3-c1	c1	6	24	2,946	27 MiB	0 B	27 MiB	3	
0									
n2-c2	c2	0	0	0	0 B	0 B	0 B	3	
0									
n1-c2	c2	0	0	0	0 B	0 B	0 B	14	
2									
n2-c1	c1	6	24	2,946	27 MiB	0 B	27 MiB	15	
0									
n1-c1*	c1	6	24	2,946	27 MiB	0 B	27 MiB	31	
0									
		18	72	8,838	80 MiB	0 B	80 MiB	67	
2									
RAFT Meta Group Information									
Name	Leader	Current	Offline	Active	Lag				
n1-c1	yes	true	false	0.00s	0				
n1-c2		true	false	0.05s	0				
n2-c1		false	true	9.00s	2				
n2-c2		true	false	0.05s	0				
n3-c1		true	false	0.05s	0				
n3-c2		true	false	0.05s	0				

This is a full cluster wide report, the report can be limited to a specific account using `--account`.

Here we see the distribution of streams, messages, api calls etc by across 2 super clusters and an overview of the RAFT meta group.

In the Meta Group report the server `n2-c1` is not current and has not been seen for 9 seconds, it's also behind by 2 raft operations.

This report is built using raw data that can be obtained from the monitor port on the `/jsz` url, or over nats using:

```
nats server req jetstream --user admin --password s3cr3t! --help
```

```
usage: nats server request jetstream [<flags>] [<wait>]
```

Show JetStream details

Flags:

<code>-h, --help</code>	Show context-sensitive help (also try <code>--help-long</code> and <code>--help-man</code>).
<code>--version</code>	Show application version.
<code>-s, --server=NATS_URL</code>	NATS server urls
<code>--user=NATS_USER</code>	Username or Token
<code>--password=NATS_PASSWORD</code>	Password
<code>--creds=NATS_CREDS</code>	User credentials
<code>--nkey=NATS_NKEY</code>	User NKEY
<code>--tlscert=NATS_CERT</code>	TLS public certificate
<code>--tlskey=NATS_KEY</code>	TLS private key
<code>--tlscacert=NATS_CA</code>	TLS certificate authority chain
<code>--timeout=NATS_TIMEOUT</code>	Time to wait on responses from NATS
<code>--js-api-prefix=PREFIX</code>	Subject prefix for access to JetStream API
<code>--js-event-prefix=PREFIX</code>	Subject prefix for access to JetStream Advisories
<code>--js-domain=DOMAIN</code>	JetStream domain to access
<code>--context=CONTEXT</code>	Configuration context
<code>--trace</code>	Trace API interactions
<code>--limit=2048</code>	Limit the responses to a certain amount of records
<code>--offset=0</code>	Start at a certain record
<code>--name=NAME</code>	Limit to servers matching a server name
<code>--host=HOST</code>	Limit to servers matching a server host name
<code>--cluster=CLUSTER</code>	Limit to servers matching a cluster name
<code>--tags=TAGS ...</code>	Limit to servers with these configured tags
<code>--account=ACCOUNT</code>	Show statistics scoped to a specific account
<code>--accounts</code>	Include details about accounts

<code>--streams</code>	Include details about Streams
<code>--consumer</code>	Include details about Consumers
<code>--config</code>	Include details about configuration
<code>--leader</code>	Request a response from the Meta-group leader only
<code>--all</code>	Include accounts, streams, consumers and configuration

Args:

`[<wait>]` Wait for a certain number of responses

```
nats server req jetstream --user admin --password s3cr3t! --leader
```

This will produce a wealth of raw information about the current state of your cluster - here requesting it from the leader only.

Forcing Meta Group leader election

Similar to Streams and Consumers above the Meta Group allows leader stand down. The Meta Group is cluster wide and spans all accounts, therefore to manage the meta group you have to use a `SYSTEM` user.

```
nats server cluster step-down --user admin --password s3cr3t!
```

```
17:44:24 Current leader: n2-c2
```

```
17:44:24 New leader: n1-c2
```

Evicting a peer

Generally when shutting down NATS, including using `Lame Duck Mode`, the cluster will notice this and continue to function.

There might be a case though where you know a node will never return, and you want to signal to JetStream that the node will not return. A `peer-remove` will remove that node from the Stream in question and all its Consumers.

After the node is removed the cluster will notice that the replica count of a stream is not honored anymore and will immediately pick a new node and start replicating data to it. The new node will be selected using the same placement rules as the existing stream.

```
nats server cluster peer-remove n4-c1 --user admin --password s3cr3t!
```

```
? Really remove offline peer n4-c1 (y/N)
```





Peer-removing nodes from the cluster is a destructive operation, and decreases the size of the cluster. A server that is to be peer-removed should ideally already be offline. It can be performed with nodes that are still online, but in this case, JetStream will be disabled on those nodes. The server should be shut down and not restarted, or if it is, JetStream should be disabled. The server may not come back under the same `server_name` if it was peer-removed and its disk wiped. This means the configured `server_name` will need to be changed to a new value before restarting.

Alternatively, if you're intending for the node to remain and only move a stream off of a specific node, you can peer-remove a node on the stream-level.

```
nats stream cluster peer-remove ORDERS
```

```
? Select a Peer n4-c1
```

```
14:38:50 Removing peer "n4-c1"
```

```
14:38:50 Requested removal of peer "n4-c1"
```

At this point the stream and all consumers will have removed `n4-c1` from the group. A new node will be selected, and data will be replicated to it. In this case `n2-c1` is selected as a new peer.

```
$ nats stream info ORDERS
```

```
....
```

```
Cluster Information:
```

```
    Name: c1
```

```
    Leader: n3-c1
```

```
    Replica: n1-c1, current, seen 0.02s ago
```

```
    Replica: n2-c1, outdated, seen 0.42s ago
```

We can see a new replica was picked, the stream is back to replication level of 3 and `n4-c1` is not active any more in this Stream or any of its Consumers.

```
## #1
```

```
□ cocobook-sync ## 2026-10-28 11:13:46 UTC
```

```
□ cocobook-sync ## 2026-10-28 11:13:46 UTC
```