ElastiCache :- Redis

What Is Amazon ElastiCache?

Elastic Cache is a web Service that makes it easy to deploy, Operate and scale in-memory Cache in Cloud. The service improves the Performance of the web application by allowing retrieving information from fast, managed, in-memory cache, instead of relying entirely on slower disk based databases.

Types of ElastiCache:

1) Memcached :-

   A widely adopted memory caching object system. Elastic cache is a protocol complaint with Memcached, so popular tools that you use today with existing memcached environments will work seamlessly with this service.

   Usage in environments with High-performance, distributed memory object caching system, intended for use in speeding up dynamic web applications.

2) Redis :-

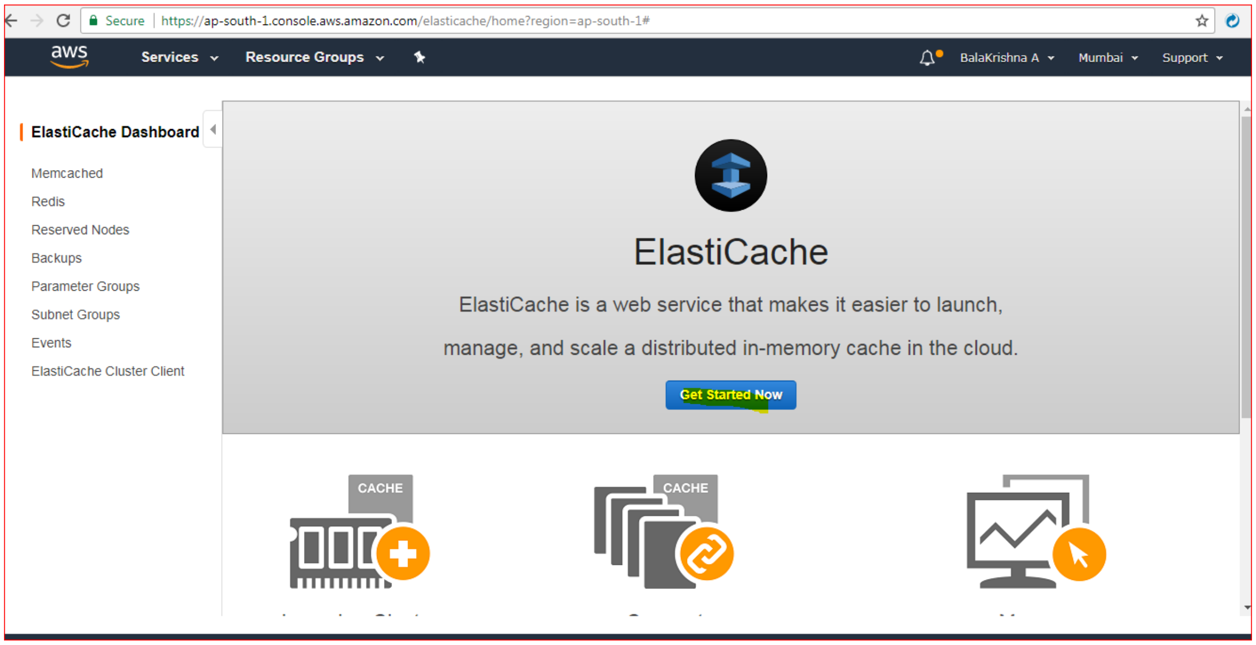
   A popular open source in-memory key value store that supports data structures such as sorted sets and lists. ElastiCache for Redis supports Master/Slave replication and Multi-AZ which can be used to achieve cross AZ redundancy.

   Usage in environments with In-memory data structure store used as database, cache and message broker. ElastiCache for Redis offers Multi-AZ with Auto-Failover and enhanced robustness.

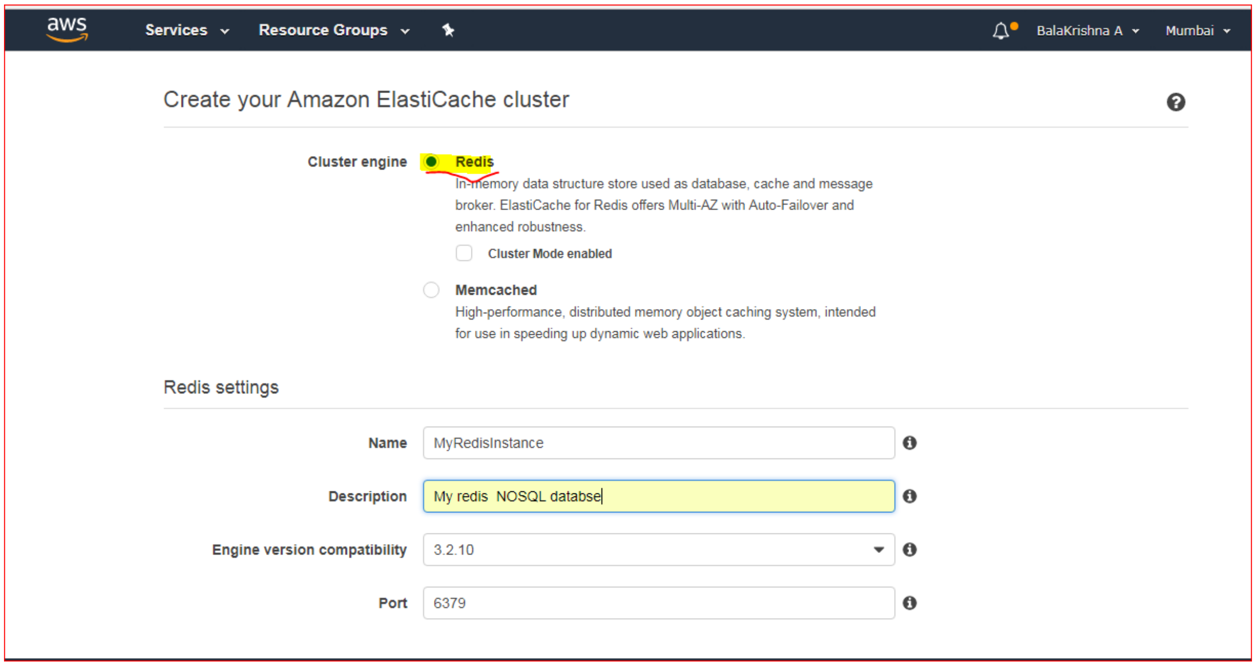
ElastiCache has multiple features to enhance reliability for critical production deployments:

a) Automatic detection and recovery from cache node failures.
b) Multi-AZ with Automatic Failover of a failed primary cluster to a read replica in Redis clusters that support replication (called *replication groups* in the ElastiCache API and AWS CLI.
c) Flexible Availability Zone placement of nodes and clusters.
d) Integration with other AWS services such as Amazon EC2, Amazon CloudWatch, AWS CloudTrail, and Amazon SNS to provide a secure, high-performance, managed in-memory caching solution.
e) ElastiCache is a good choice if your database is read heavy and not prone to frequent changing.

LAB Practice session for Redis



ElastiCache is a web service that makes it easier to launch, manage, and scale a distributed in-memory cache in the cloud.



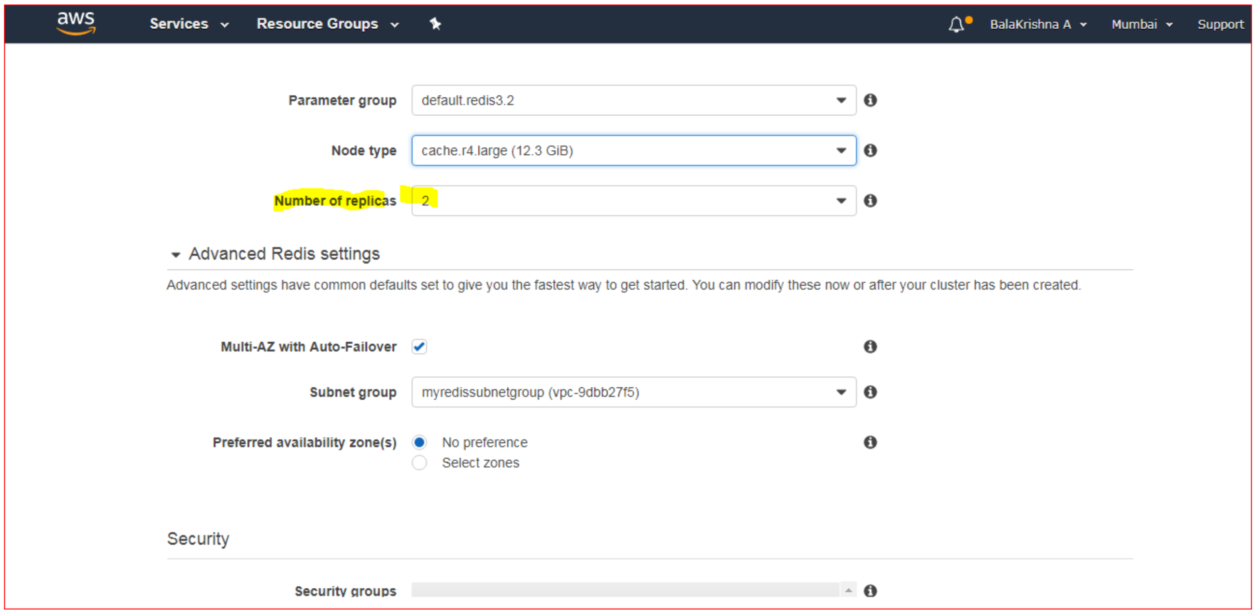
Create your Amazon ElastiCache cluster

Cluster engine (●) Redis
In-memory data structure store used as database, cache and message broker. ElastiCache for Redis offers Multi-AZ with Auto-Failover and enhanced robustness.
☐ Cluster Mode enabled

( ) Memcached
High-performance, distributed memory object caching system, intended for use in speeding up dynamic web applications.

Redis settings

Name	MyRedisInstance
Description	My redis NOSQL databse
Engine version compatibility	3.2.10
Port	6379



Parameter group	default.redis3.2
Node type	cache.r4.large (12.3 GiB)
Number of replicas	2

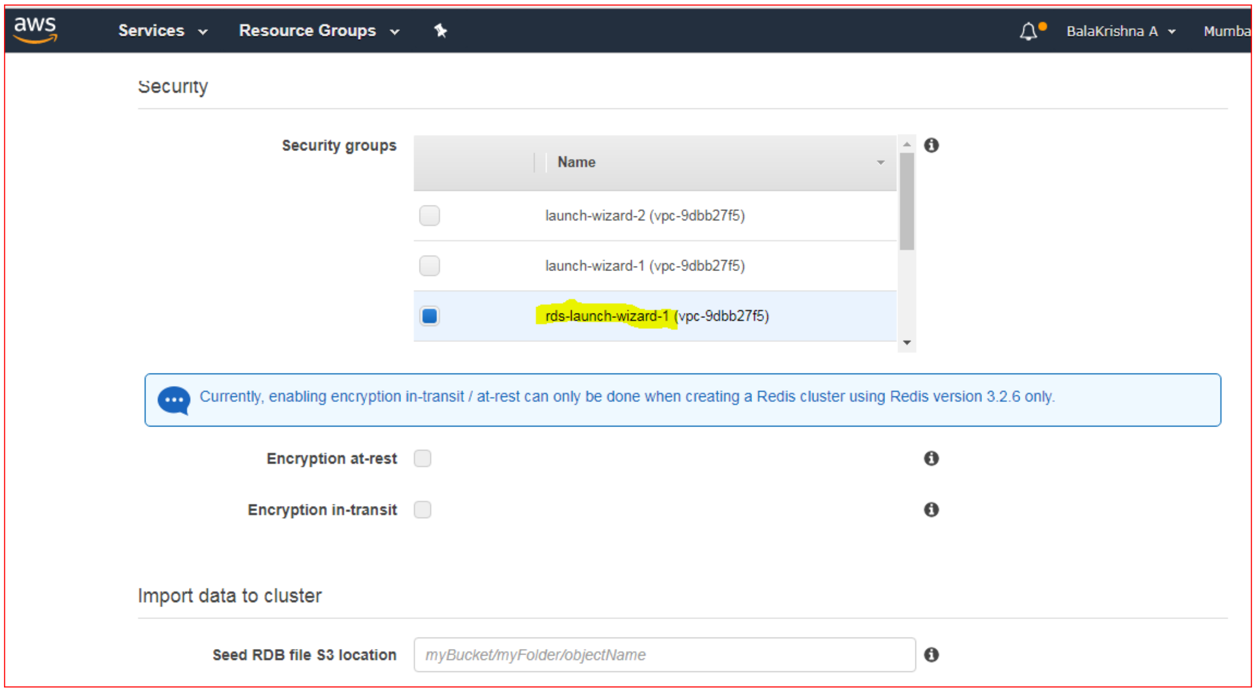
Advanced Redis settings

Advanced settings have common defaults set to give you the fastest way to get started. You can modify these now or after your cluster has been created.

Multi-AZ with Auto-Failover	✓
Subnet group	myredissubnetgroup (vpc-9dbb27f5)
Preferred availability zone(s)	(●) No preference ( ) Select zones

Security

Security groups



Security

Security groups

	Name
☐	launch-wizard-2 (vpc-9dbb27f5)
☐	launch-wizard-1 (vpc-9dbb27f5)
☑	rds-launch-wizard-1 (vpc-9dbb27f5)

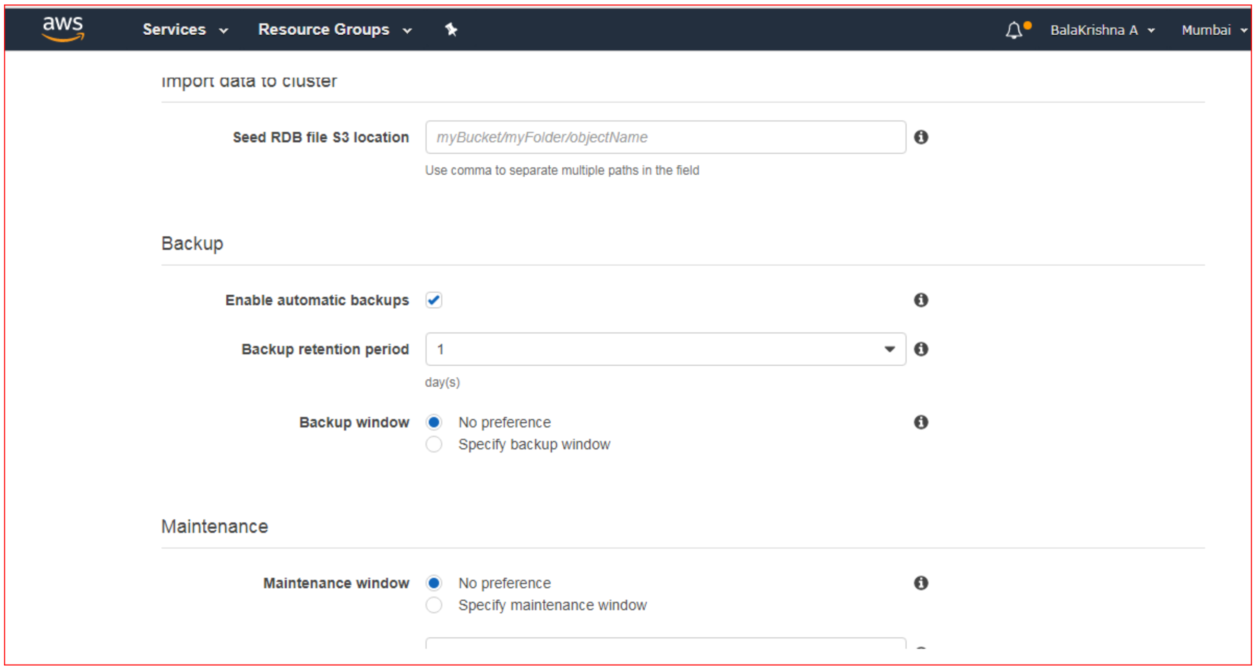
Currently, enabling encryption in-transit / at-rest can only be done when creating a Redis cluster using Redis version 3.2.6 only.

Encryption at-rest ☐

Encryption in-transit ☐

Import data to cluster

Seed RDB file S3 location: myBucket/myFolder/objectName



Import data to cluster

Seed RDB file S3 location: myBucket/myFolder/objectName
Use comma to separate multiple paths in the field

Backup

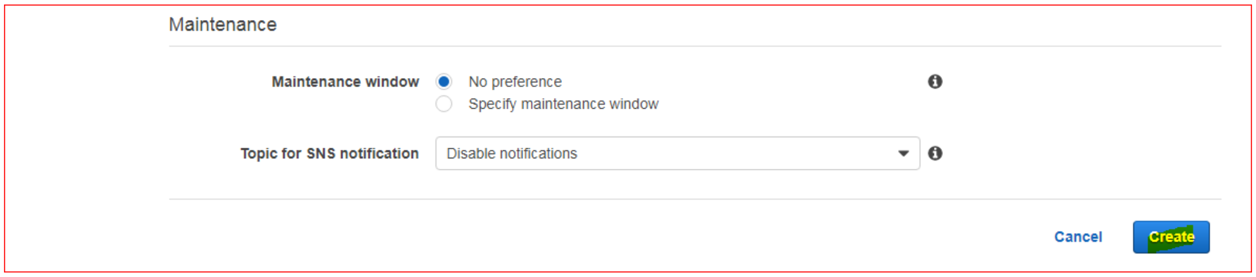
Enable automatic backups ✓

Backup retention period: 1
day(s)

Backup window: [✓ No preference] Specify backup window

Maintenance

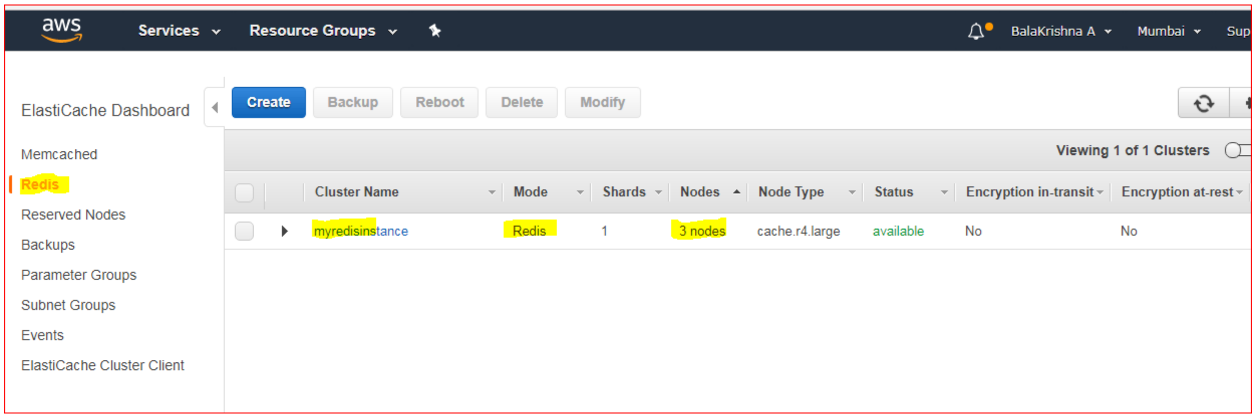
Maintenance window: [✓ No preference] Specify maintenance window



Maintenance

Maintenance window: [✓ No preference] Specify maintenance window

Topic for SNS notification: Disable notifications

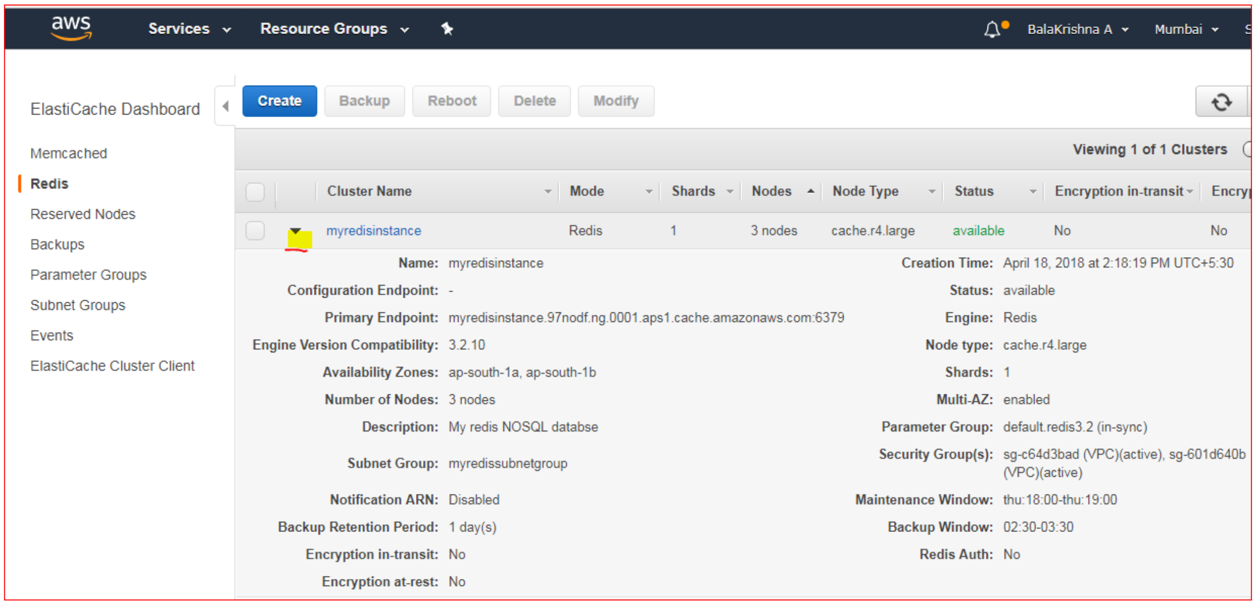


ElastiCache Dashboard

- Memcached
- Redis
- Reserved Nodes
- Backups
- Parameter Groups
- Subnet Groups
- Events
- ElastiCache Cluster Client

Viewing 1 of 1 Clusters

Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encryption at-rest
myredisinstance	Redis	1	3 nodes	cache.r4.large	available	No	No



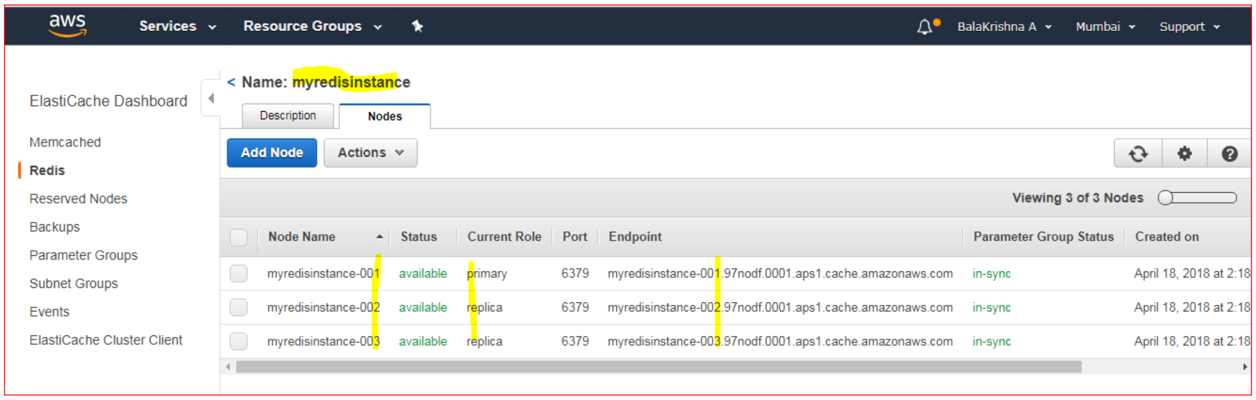
ElastiCache Dashboard

- Memcached
- Redis
- Reserved Nodes
- Backups
- Parameter Groups
- Subnet Groups
- Events
- ElastiCache Cluster Client

Viewing 1 of 1 Clusters

Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encry
myredisinstance	Redis	1	3 nodes	cache.r4.large	available	No	No

Name:	myredisinstance	Creation Time:	April 18, 2018 at 2:18:19 PM UTC+5:30
Configuration Endpoint:	-	Status:	available
Primary Endpoint:	myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379	Engine:	Redis
Engine Version Compatibility:	3.2.10	Node type:	cache.r4.large
Availability Zones:	ap-south-1a, ap-south-1b	Shards:	1
Number of Nodes:	3 nodes	Multi-AZ:	enabled
Description:	My redis NOSQL databse	Parameter Group:	default.redis3.2 (in-sync)
Subnet Group:	myredissubnetgroup	Security Group(s):	sg-c64d3bad (VPC)(active), sg-601d640b (VPC)(active)
Notification ARN:	Disabled	Maintenance Window:	thu:18:00-thu:19:00
Backup Retention Period:	1 day(s)	Backup Window:	02:30-03:30
Encryption in-transit:	No	Redis Auth:	No
Encryption at-rest:	No		



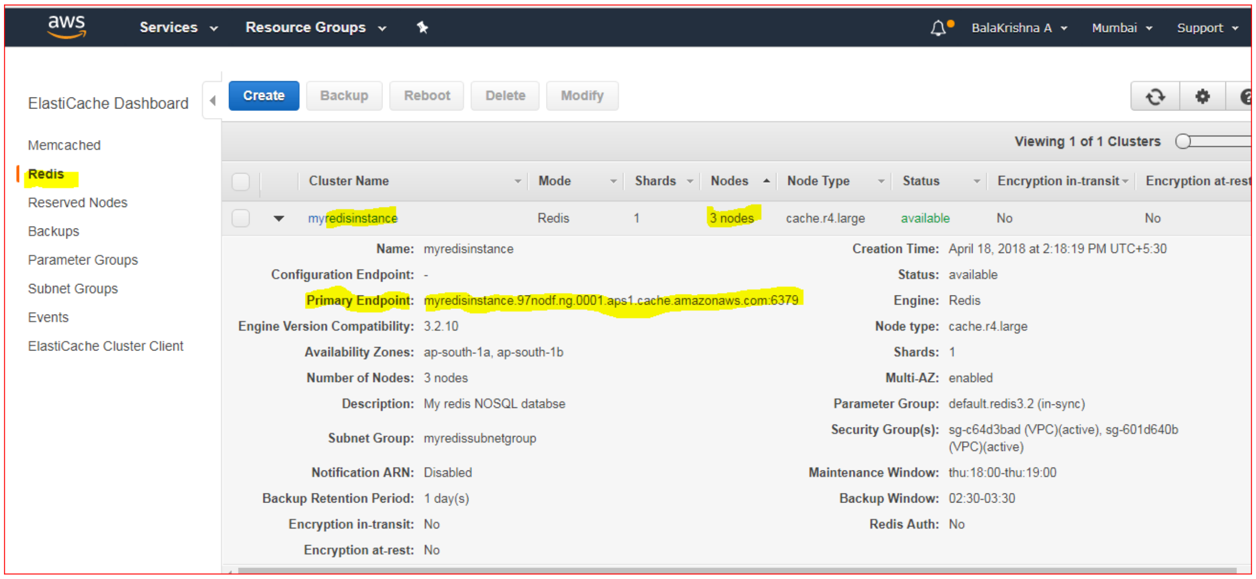
ElastiCache Dashboard

- Memcached
- Redis
- Reserved Nodes
- Backups
- Parameter Groups
- Subnet Groups
- Events
- ElastiCache Cluster Client

< Name: myredisinstance

Viewing 3 of 3 Nodes

Node Name	Status	Current Role	Port	Endpoint	Parameter Group Status	Created on
myredisinstance-001	available	primary	6379	myredisinstance-001.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:18
myredisinstance-002	available	replica	6379	myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:18
myredisinstance-003	available	replica	6379	myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:18



Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encryption at-rest
myredisinstance	Redis	1	3 nodes	cache.r4.large	available	No	No

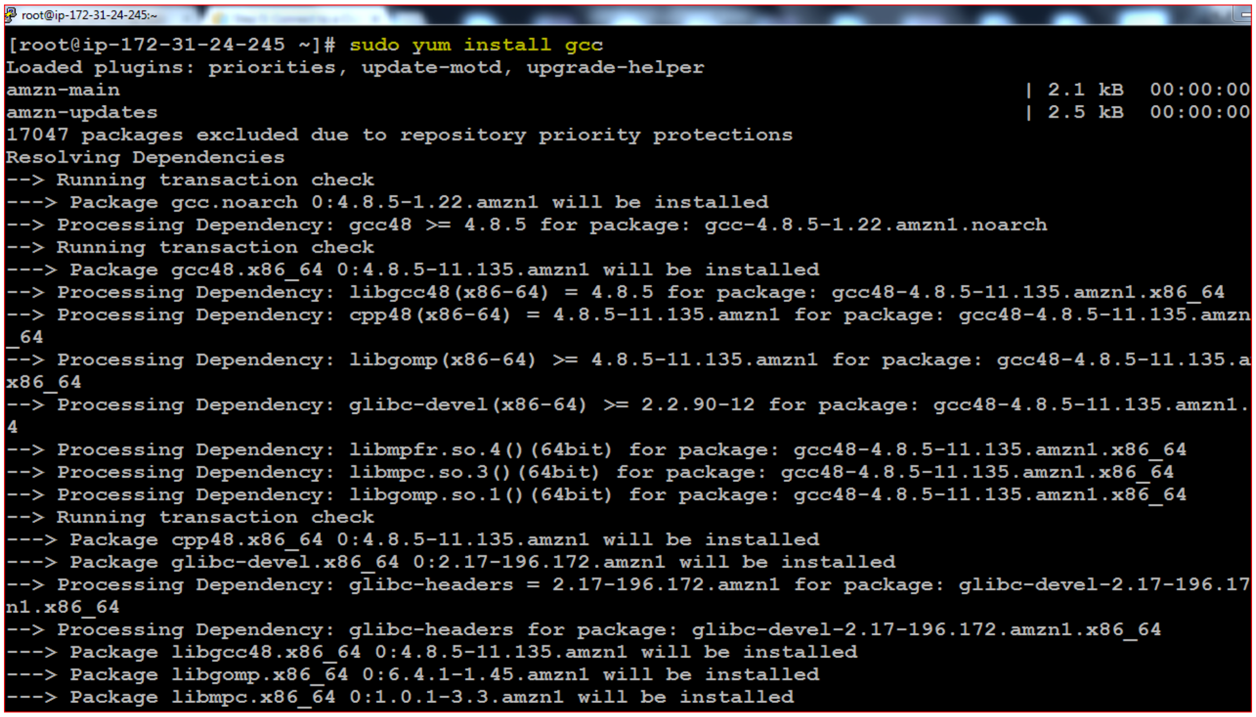
Name:	myredisinstance	Creation Time:	April 18, 2018 at 2:18:19 PM UTC+5:30
Configuration Endpoint:	-	Status:	available
Primary Endpoint:	myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379	Engine:	Redis
Engine Version Compatibility:	3.2.10	Node type:	cache.r4.large
Availability Zones:	ap-south-1a, ap-south-1b	Shards:	1
Number of Nodes:	3 nodes	Multi-AZ:	enabled
Description:	My redis NOSQL databse	Parameter Group:	default.redis3.2 (in-sync)
Subnet Group:	myredissubnetgroup	Security Group(s):	sg-c64d3bad (VPC)(active), sg-601d640b (VPC)(active)
Notification ARN:	Disabled	Maintenance Window:	thu:18:00-thu:19:00
Backup Retention Period:	1 day(s)	Backup Window:	02:30-03:30
Encryption in-transit:	No	Redis Auth:	No
Encryption at-rest:	No		

Connect to a Redis cluster that is not encryption enabled using the *redis-cli*

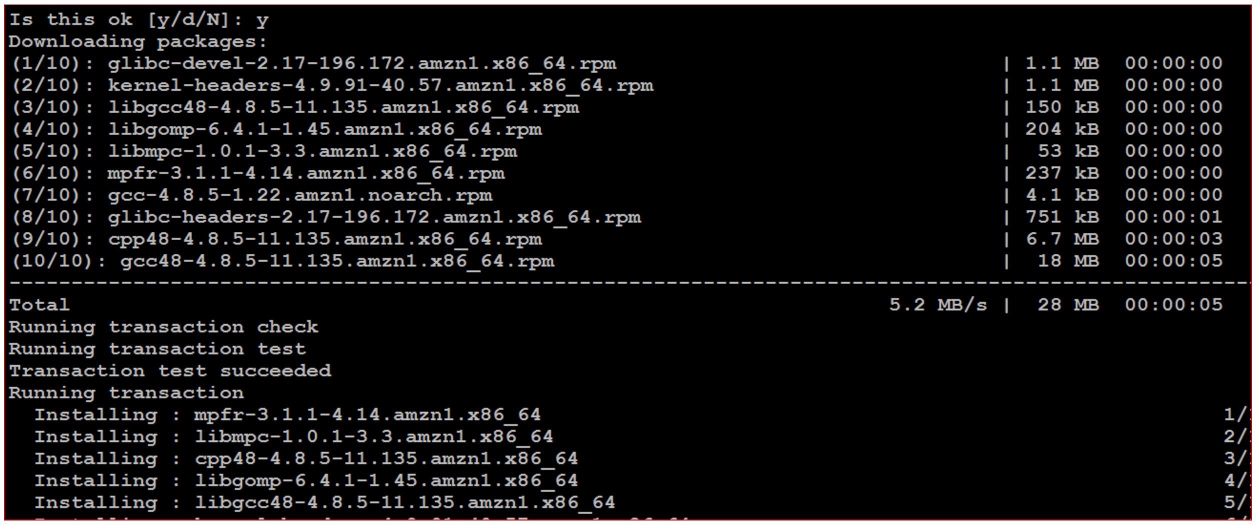
1. Connect to your Amazon EC2 instance using the connection utility of your choice.
2. Download and install the GNU Compiler Collection (gcc).

At the command prompt of your EC2 instance, type the following command then, at the confirmation prompt, type `y` .

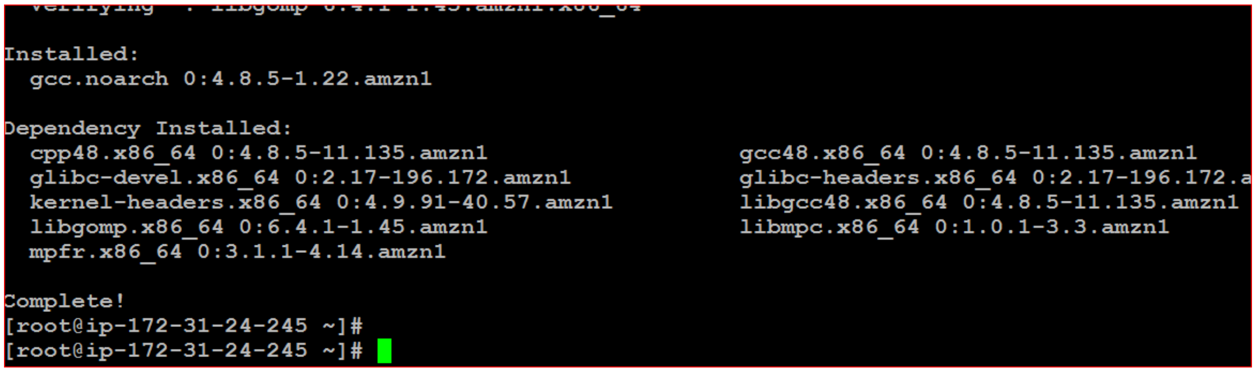
```
sudo yum install gcc
```


root@ip-172-31-24-245:~

```
[root@ip-172-31-24-245 ~]# sudo yum install gcc
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-main                                                                                   | 2.1 kB  00:00:00
amzn-updates                                                                                | 2.5 kB  00:00:00
17047 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package gcc.noarch 0:4.8.5-1.22.amzn1 will be installed
--> Processing Dependency: gcc48 >= 4.8.5 for package: gcc-4.8.5-1.22.amzn1.noarch
--> Running transaction check
---> Package gcc48.x86_64 0:4.8.5-11.135.amzn1 will be installed
--> Processing Dependency: libgcc48(x86-64) = 4.8.5 for package: gcc48-4.8.5-11.135.amzn1.x86_64
--> Processing Dependency: cpp48(x86-64) = 4.8.5-11.135.amzn1 for package: gcc48-4.8.5-11.135.amzn
_64
--> Processing Dependency: libgomp(x86-64) >= 4.8.5-11.135.amzn1 for package: gcc48-4.8.5-11.135.a
x86_64
--> Processing Dependency: glibc-devel(x86-64) >= 2.2.90-12 for package: gcc48-4.8.5-11.135.amzn1.
4
--> Processing Dependency: libmpfr.so.4()(64bit) for package: gcc48-4.8.5-11.135.amzn1.x86_64
--> Processing Dependency: libmpc.so.3()(64bit) for package: gcc48-4.8.5-11.135.amzn1.x86_64
--> Processing Dependency: libgomp.so.1()(64bit) for package: gcc48-4.8.5-11.135.amzn1.x86_64
--> Running transaction check
---> Package cpp48.x86_64 0:4.8.5-11.135.amzn1 will be installed
---> Package glibc-devel.x86_64 0:2.17-196.172.amzn1 will be installed
--> Processing Dependency: glibc-headers = 2.17-196.172.amzn1 for package: glibc-devel-2.17-196.17
n1.x86_64
--> Processing Dependency: glibc-headers for package: glibc-devel-2.17-196.172.amzn1.x86_64
---> Package libgcc48.x86_64 0:4.8.5-11.135.amzn1 will be installed
---> Package libgomp.x86_64 0:6.4.1-1.45.amzn1 will be installed
---> Package libmpc.x86_64 0:1.0.1-3.3.amzn1 will be installed
```



```
Is this ok [y/d/N]: y
Downloading packages:
(1/10): glibc-devel-2.17-196.172.amzn1.x86_64.rpm                                        | 1.1 MB  00:00:00
(2/10): kernel-headers-4.9.91-40.57.amzn1.x86_64.rpm                                     | 1.1 MB  00:00:00
(3/10): libgcc48-4.8.5-11.135.amzn1.x86_64.rpm                                           | 150 kB  00:00:00
(4/10): libgomp-6.4.1-1.45.amzn1.x86_64.rpm                                              | 204 kB  00:00:00
(5/10): libmpc-1.0.1-3.3.amzn1.x86_64.rpm                                                |  53 kB  00:00:00
(6/10): mpfr-3.1.1-4.14.amzn1.x86_64.rpm                                                 | 237 kB  00:00:00
(7/10): gcc-4.8.5-1.22.amzn1.noarch.rpm                                                  | 4.1 kB  00:00:00
(8/10): glibc-headers-2.17-196.172.amzn1.x86_64.rpm                                      | 751 kB  00:00:01
(9/10): cpp48-4.8.5-11.135.amzn1.x86_64.rpm                                              | 6.7 MB  00:00:03
(10/10): gcc48-4.8.5-11.135.amzn1.x86_64.rpm                                             |  18 MB  00:00:05
--------------------------------------------------------------------------------------------------------------
Total                                                                          5.2 MB/s |  28 MB  00:00:05
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : mpfr-3.1.1-4.14.amzn1.x86_64                                                              1/
  Installing : libmpc-1.0.1-3.3.amzn1.x86_64                                                             2/
  Installing : cpp48-4.8.5-11.135.amzn1.x86_64                                                           3/
  Installing : libgomp-6.4.1-1.45.amzn1.x86_64                                                           4/
  Installing : libgcc48-4.8.5-11.135.amzn1.x86_64                                                        5/
```

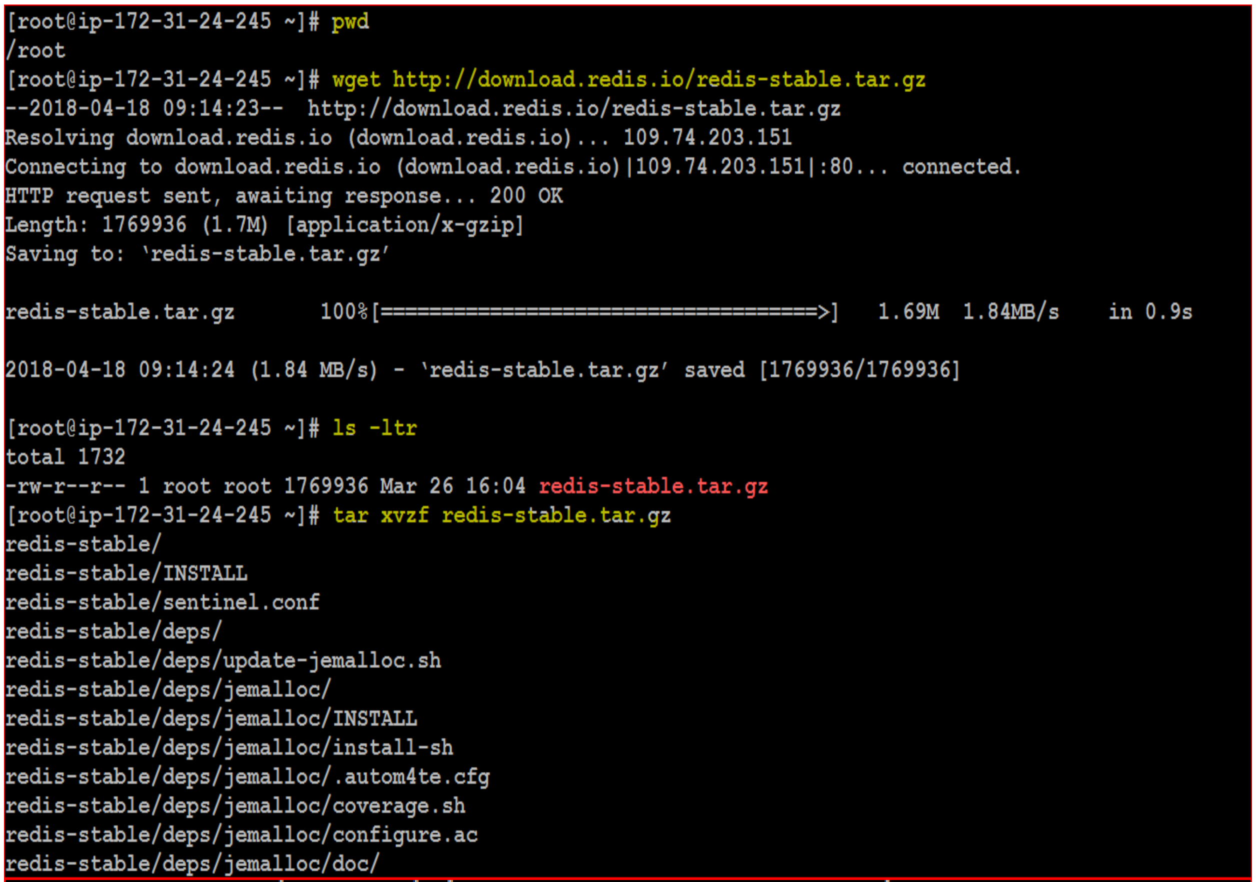


```
  Verifying  : libgomp-6.4.1-1.45.amzn1.x86_64

Installed:
  gcc.noarch 0:4.8.5-1.22.amzn1

Dependency Installed:
  cpp48.x86_64 0:4.8.5-11.135.amzn1                    gcc48.x86_64 0:4.8.5-11.135.amzn1
  glibc-devel.x86_64 0:2.17-196.172.amzn1              glibc-headers.x86_64 0:2.17-196.172.a
  kernel-headers.x86_64 0:4.9.91-40.57.amzn1           libgcc48.x86_64 0:4.8.5-11.135.amzn1
  libgomp.x86_64 0:6.4.1-1.45.amzn1                    libmpc.x86_64 0:1.0.1-3.3.amzn1
  mpfr.x86_64 0:3.1.1-4.14.amzn1

Complete!
[root@ip-172-31-24-245 ~]#
[root@ip-172-31-24-245 ~]#
```

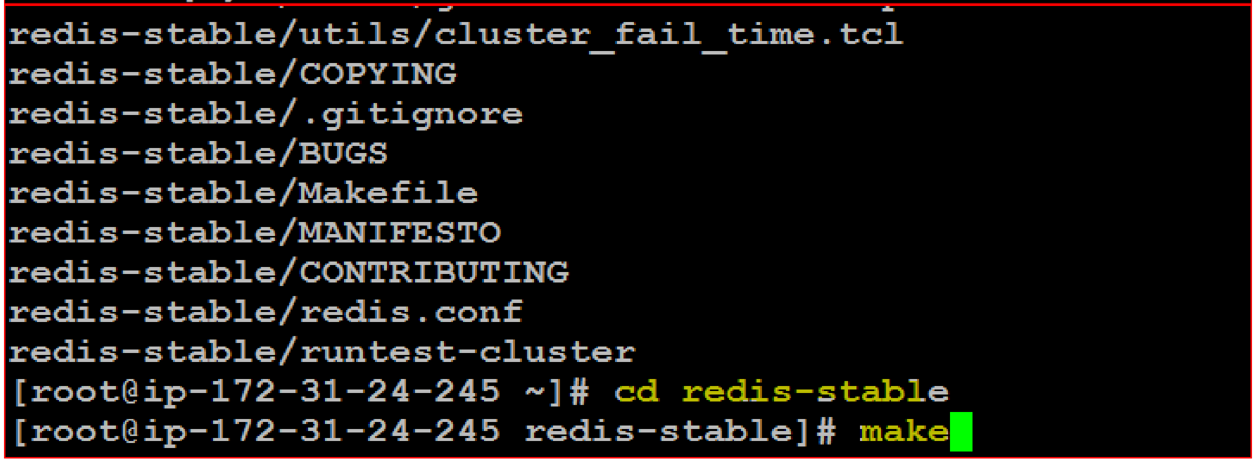
3. Download and compile the `redis-cli` utility. This utility is included in the Redis software distribution.

   At the command prompt of your EC2 instance, type the following commands:

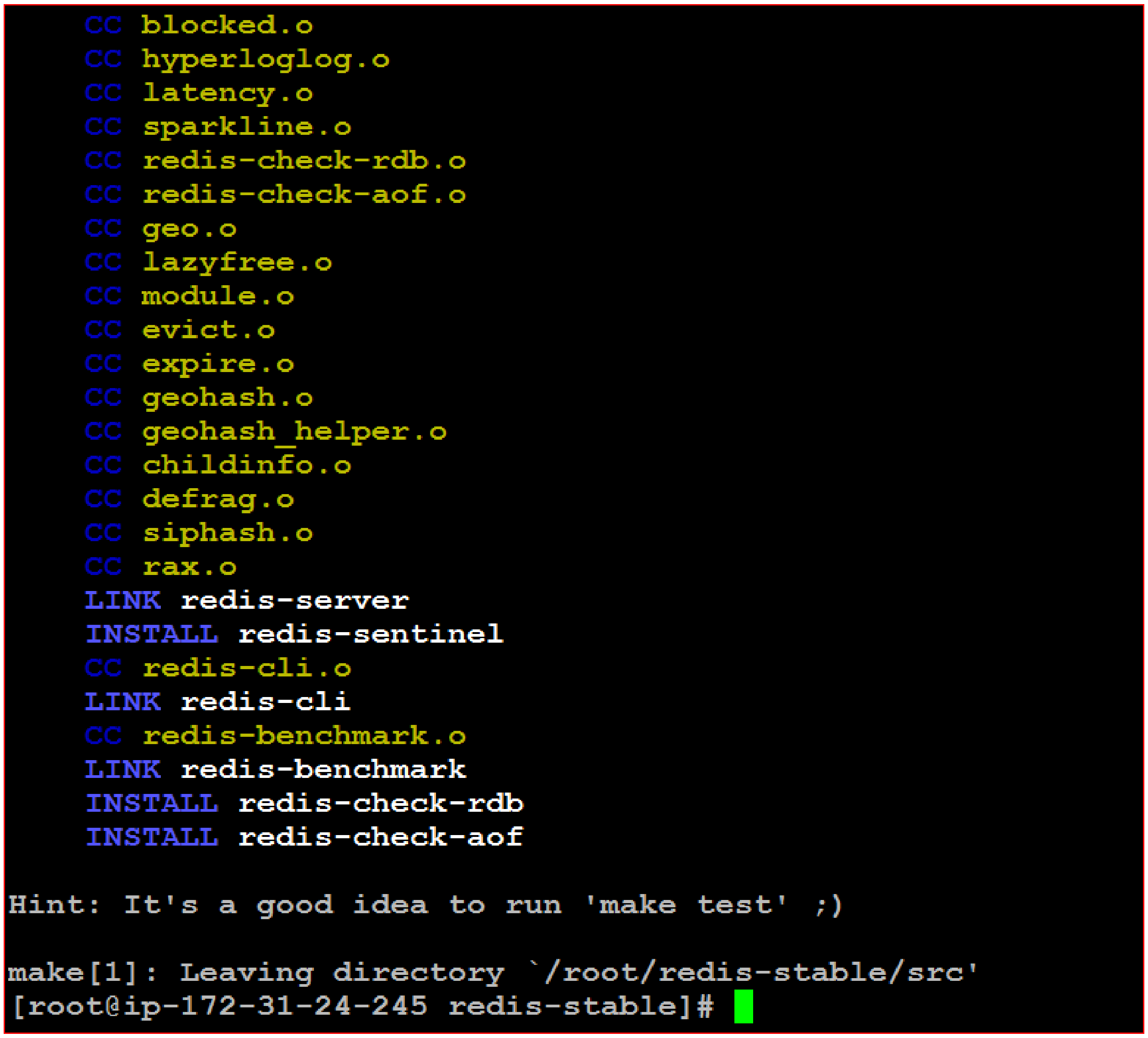
```
        wget http://download.redis.io/redis-stable.tar.gz
        tar xvzf redis-stable.tar.gz
        cd redis-stable
        make distclean  // this command need to execute only for Ubuntu systems only
        make
```



```
[root@ip-172-31-24-245 ~]# pwd
/root
[root@ip-172-31-24-245 ~]# wget http://download.redis.io/redis-stable.tar.gz
--2018-04-18 09:14:23--  http://download.redis.io/redis-stable.tar.gz
Resolving download.redis.io (download.redis.io)... 109.74.203.151
Connecting to download.redis.io (download.redis.io)|109.74.203.151|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1769936 (1.7M) [application/x-gzip]
Saving to: 'redis-stable.tar.gz'

redis-stable.tar.gz       100%[=====================================>]   1.69M  1.84MB/s    in 0.9s

2018-04-18 09:14:24 (1.84 MB/s) - 'redis-stable.tar.gz' saved [1769936/1769936]

[root@ip-172-31-24-245 ~]# ls -ltr
total 1732
-rw-r--r-- 1 root root 1769936 Mar 26 16:04 redis-stable.tar.gz
[root@ip-172-31-24-245 ~]# tar xvzf redis-stable.tar.gz
redis-stable/
redis-stable/INSTALL
redis-stable/sentinel.conf
redis-stable/deps/
redis-stable/deps/update-jemalloc.sh
redis-stable/deps/jemalloc/
redis-stable/deps/jemalloc/INSTALL
redis-stable/deps/jemalloc/install-sh
redis-stable/deps/jemalloc/.autom4te.cfg
redis-stable/deps/jemalloc/coverage.sh
redis-stable/deps/jemalloc/configure.ac
redis-stable/deps/jemalloc/doc/
```

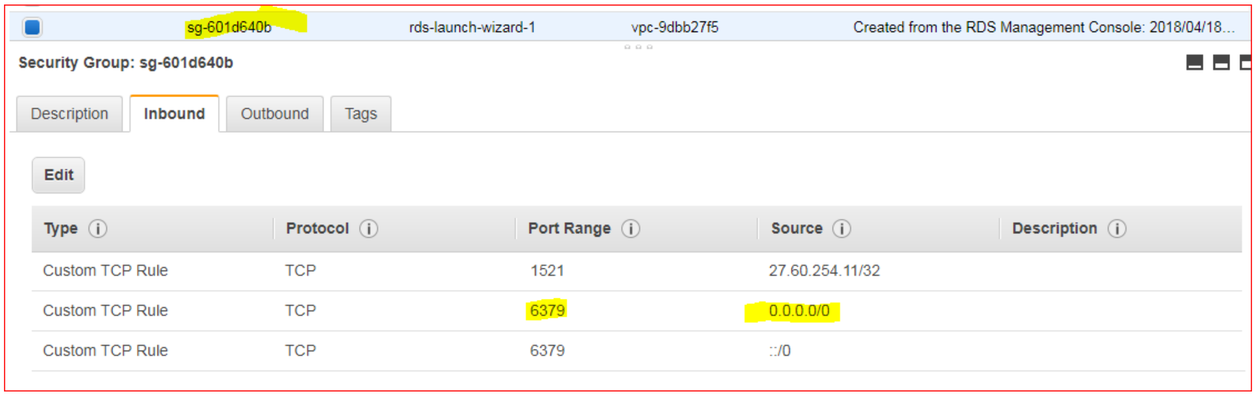


```
redis-stable/utils/cluster_fail_time.tcl
redis-stable/COPYING
redis-stable/.gitignore
redis-stable/BUGS
redis-stable/Makefile
redis-stable/MANIFESTO
redis-stable/CONTRIBUTING
redis-stable/redis.conf
redis-stable/runtest-cluster
[root@ip-172-31-24-245 ~]# cd redis-stable
[root@ip-172-31-24-245 redis-stable]# make
```



```
    CC blocked.o
    CC hyperloglog.o
    CC latency.o
    CC sparkline.o
    CC redis-check-rdb.o
    CC redis-check-aof.o
    CC geo.o
    CC lazyfree.o
    CC module.o
    CC evict.o
    CC expire.o
    CC geohash.o
    CC geohash_helper.o
    CC childinfo.o
    CC defrag.o
    CC siphash.o
    CC rax.o
    LINK redis-server
    INSTALL redis-sentinel
    CC redis-cli.o
    LINK redis-cli
    CC redis-benchmark.o
    LINK redis-benchmark
    INSTALL redis-check-rdb
    INSTALL redis-check-aof

Hint: It's a good idea to run 'make test' ;)

make[1]: Leaving directory `/root/redis-stable/src'
[root@ip-172-31-24-245 redis-stable]#
```

Allow/Add port No: 6379 to the security Group (Inboud rule) of redis instance as show below .



sg-601d640b rds-launch-wizard-1 vpc-9dbb27f5 Created from the RDS Management Console: 2018/04/18...

Security Group: sg-601d640b

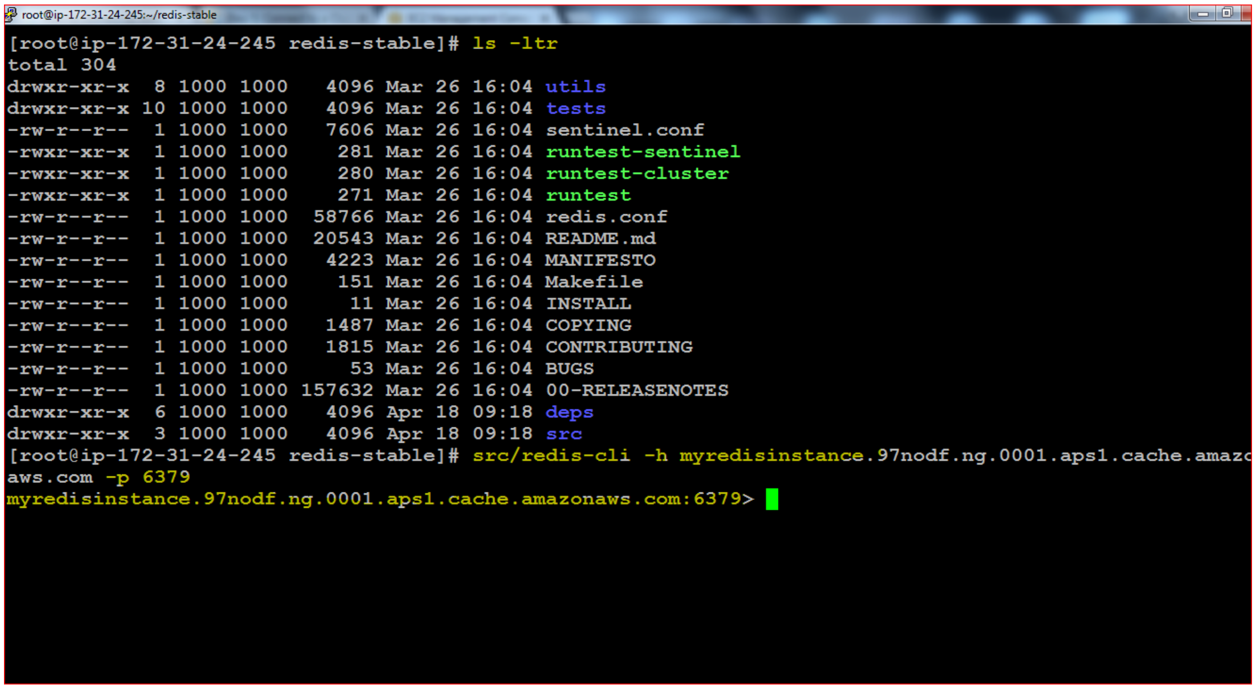
Type	Protocol	Port Range	Source	Description
Custom TCP Rule	TCP	1521	27.60.254.11/32	
Custom TCP Rule	TCP	6379	0.0.0.0/0	
Custom TCP Rule	TCP	6379	::/0	

4. At the command prompt of your EC2 instance, type the following command, substituting the endpoint of your cluster and port for what is shown in this example.

```
src/redis-cli -c -h mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com -p 6379
```

This results in a Redis command prompt similar to the following.

```
redis mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com 6379>
```



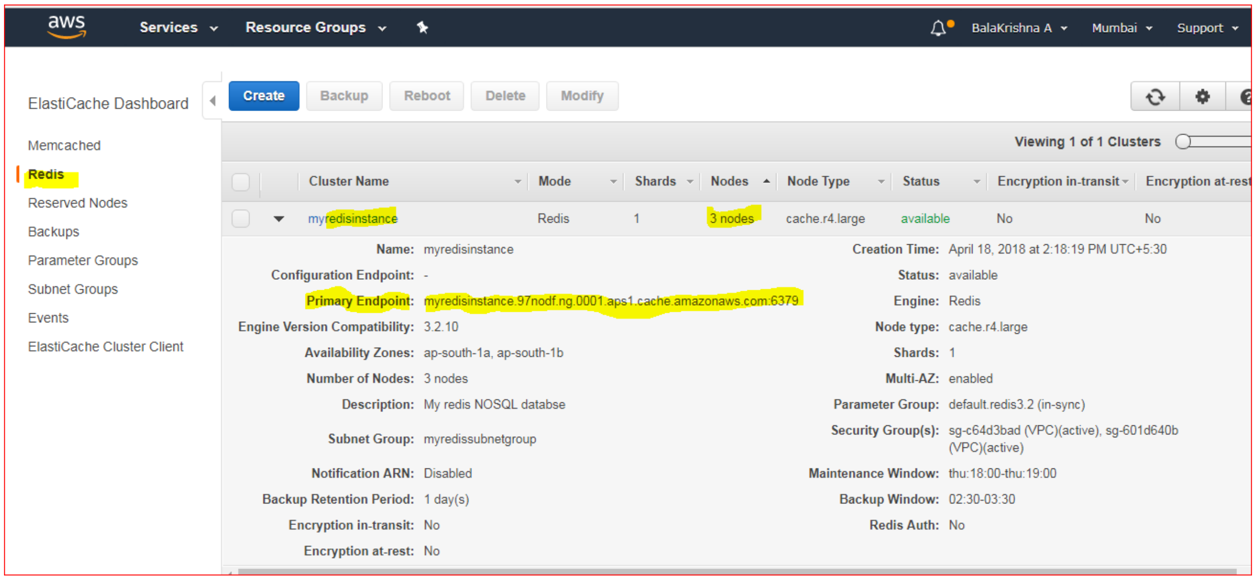
```
root@ip-172-31-24-245:~/redis-stable
[root@ip-172-31-24-245 redis-stable]# ls -ltr
total 304
drwxr-xr-x  8 1000 1000   4096 Mar 26 16:04 utils
drwxr-xr-x 10 1000 1000   4096 Mar 26 16:04 tests
-rw-r--r--  1 1000 1000   7606 Mar 26 16:04 sentinel.conf
-rwxr-xr-x  1 1000 1000    281 Mar 26 16:04 runtest-sentinel
-rwxr-xr-x  1 1000 1000    280 Mar 26 16:04 runtest-cluster
-rwxr-xr-x  1 1000 1000    271 Mar 26 16:04 runtest
-rw-r--r--  1 1000 1000  58766 Mar 26 16:04 redis.conf
-rw-r--r--  1 1000 1000  20543 Mar 26 16:04 README.md
-rw-r--r--  1 1000 1000   4223 Mar 26 16:04 MANIFESTO
-rw-r--r--  1 1000 1000    151 Mar 26 16:04 Makefile
-rw-r--r--  1 1000 1000     11 Mar 26 16:04 INSTALL
-rw-r--r--  1 1000 1000   1487 Mar 26 16:04 COPYING
-rw-r--r--  1 1000 1000   1815 Mar 26 16:04 CONTRIBUTING
-rw-r--r--  1 1000 1000     53 Mar 26 16:04 BUGS
-rw-r--r--  1 1000 1000 157632 Mar 26 16:04 00-RELEASENOTES
drwxr-xr-x  6 1000 1000   4096 Apr 18 09:18 deps
drwxr-xr-x  3 1000 1000   4096 Apr 18 09:18 src
[root@ip-172-31-24-245 redis-stable]# src/redis-cli -h myredisinstance.97nodf.ng.0001.aps1.cache.amazo
aws.com -p 6379
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379>
```

5. Run Redis commands.

You are now connected to the cluster and can run Redis commands like the following.

```
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
get b                  // Get value for key "b" results in miss
(nil)
set b "Good-bye" EX 5  // Set key "b" with a string value and a 5 second expiration
"Good-bye"
```

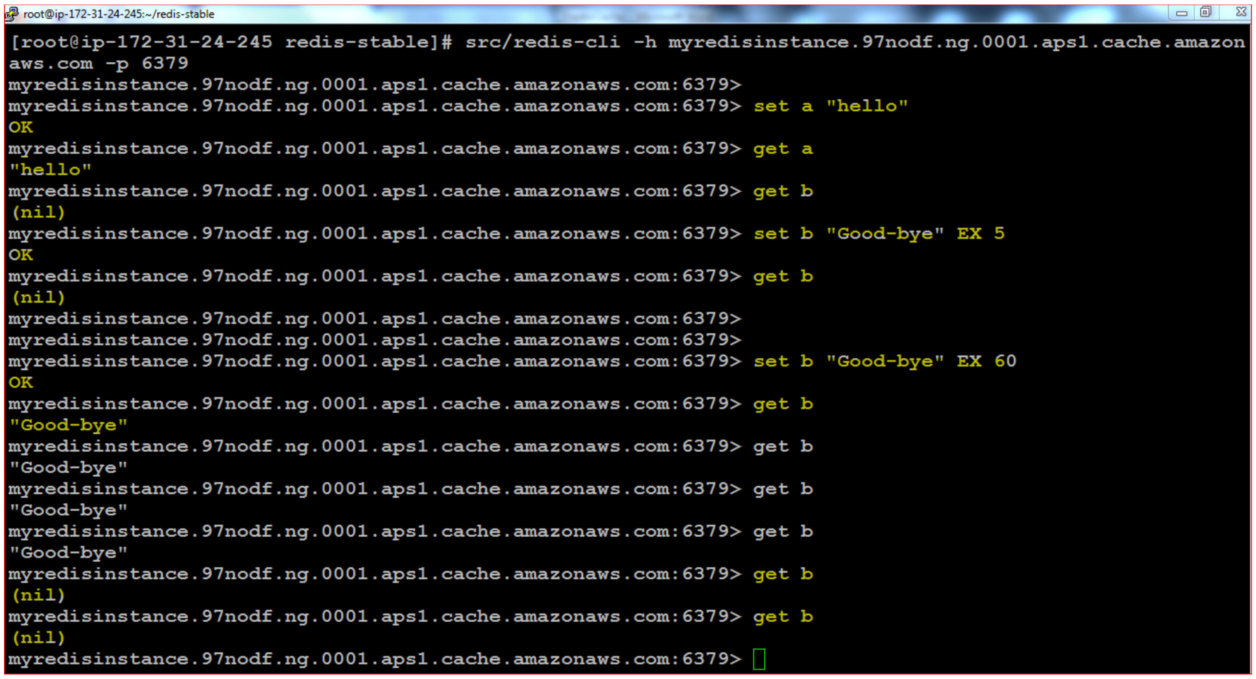
```
get b                        // Get value for key "b"
"Good-bye"
                             // wait >= 5 seconds
get b
(nil)                        // key has expired, nothing returned
quit                         // Exit from redis-cli
```



Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encryption at-rest
myredisinstance	Redis	1	3 nodes	cache.r4.large	available	No	No

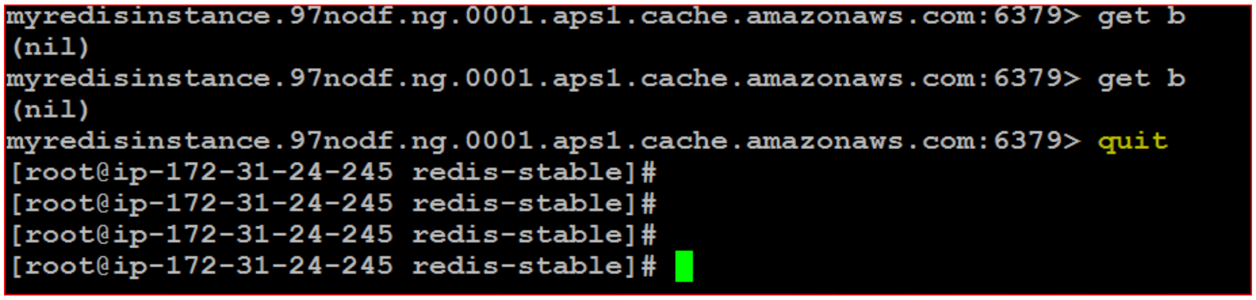
Name:	myredisinstance	Creation Time:	April 18, 2018 at 2:18:19 PM UTC+5:30
Configuration Endpoint:	-	Status:	available
Primary Endpoint:	myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379	Engine:	Redis
Engine Version Compatibility:	3.2.10	Node type:	cache.r4.large
Availability Zones:	ap-south-1a, ap-south-1b	Shards:	1
Number of Nodes:	3 nodes	Multi-AZ:	enabled
Description:	My redis NOSQL databse	Parameter Group:	default.redis3.2 (in-sync)
Subnet Group:	myredissubnetgroup	Security Group(s):	sg-c64d3bad (VPC)(active), sg-601d640b (VPC)(active)
Notification ARN:	Disabled	Maintenance Window:	thu:18:00-thu:19:00
Backup Retention Period:	1 day(s)	Backup Window:	02:30-03:30
Encryption in-transit:	No	Redis Auth:	No
Encryption at-rest:	No		

Code to Primary Node



```
[root@ip-172-31-24-245 redis-stable]# src/redis-cli -h myredisinstance.97nodf.ng.0001.aps1.cache.amazon
aws.com -p 6379
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379>
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> set a "hello"
OK
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get a
"hello"
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> set b "Good-bye" EX 5
OK
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379>
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379>
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> set b "Good-bye" EX 60
OK
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
"Good-bye"
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
"Good-bye"
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
"Good-bye"
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
"Good-bye"
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379>
```

Quit from redis-cli



```
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> get b
(nil)
myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379> quit
[root@ip-172-31-24-245 redis-stable]#
[root@ip-172-31-24-245 redis-stable]#
[root@ip-172-31-24-245 redis-stable]#
[root@ip-172-31-24-245 redis-stable]#
```

Step 5.2.a: Connect to an Encrypted Redis Cluster or Replication Group

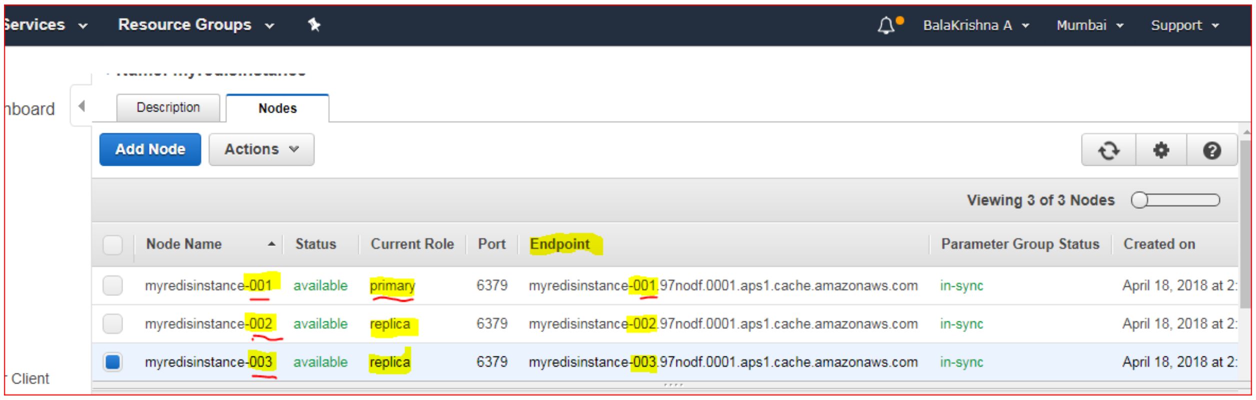
Amazon ElastiCache introduced Encryption in In-Transit, At-Rest and Authentication. To connect to an Encryption in transit enabled cluster we need to use a client that supports SSL. Unfortunately, redis-cli does not support SSL.

The redis-cli does not support SSL/TLS connections.

```
redis-cli -h master.ssltest.xxxxxx.use1.cache.amazonaws.com -p 6379
master.ssltest.xxxxxx.use1.cache.amazonaws.com:6379>set key1 value
```

Output from the preceding command:

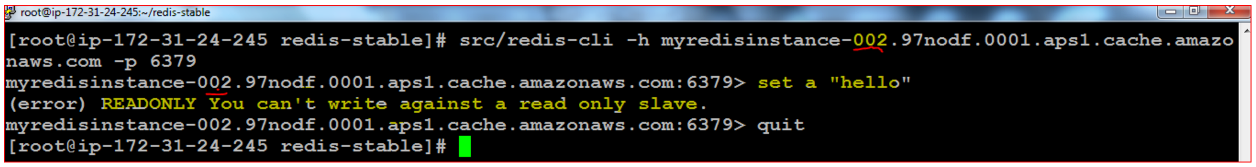
```
Error: Connection reset by peer
```



Viewing 3 of 3 Nodes

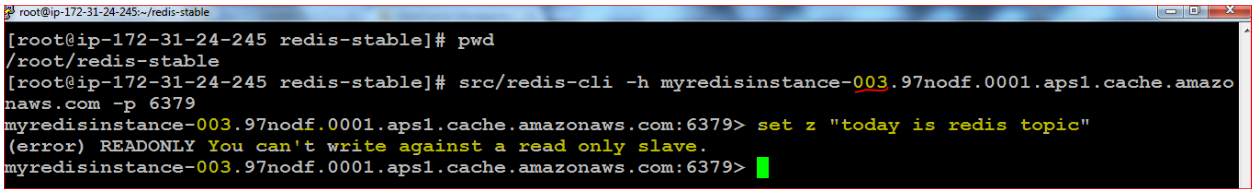
	Node Name	Status	Current Role	Port	Endpoint	Parameter Group Status	Created on
	myredisinstance-001	available	primary	6379	myredisinstance-001.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:
	myredisinstance-002	available	replica	6379	myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:
	myredisinstance-003	available	replica	6379	myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:

Connecting to node2(slave2)



```
[root@ip-172-31-24-245 redis-stable]# src/redis-cli -h myredisinstance-002.97nodf.0001.aps1.cache.amazo
naws.com -p 6379
myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com:6379> set a "hello"
(error) READONLY You can't write against a read only slave.
myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com:6379> quit
[root@ip-172-31-24-245 redis-stable]#
```

Connecting to node3 (slave3)



```
[root@ip-172-31-24-245 redis-stable]# pwd
/root/redis-stable
[root@ip-172-31-24-245 redis-stable]# src/redis-cli -h myredisinstance-003.97nodf.0001.aps1.cache.amazo
naws.com -p 6379
myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com:6379> set z "today is redis topic"
(error) READONLY You can't write against a read only slave.
myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com:6379>
```

Because tools like redis-cli and telnet are useful for running ad-hoc commands, this section will show you how to create and use an SSL tunnel to your Redis cluster and then use redis-cli to run commands.

We can create SSL tunnel using `stunnel` and use redis-cli over it to connect to encrypted Redis. It is very easy to setup `stunnel` as most of the configuration setup are already done at the ElastiCache layer.

1. Install `stunnel`.

   ```
   sudo yum install stunnel
   ```

2. Configure `stunnel`. You can set up as many connections as are needed.

```
3.  cat /etc/stunnel/redis-cli.conf
4.  fips=no
5.  setuid=root
6.  setgid=root
7.  pid=/var/run//stunnel.pid
8.  debug=7
9.  options=NO_SSLv2
10. options=NO_SSLv3
11. [redis-cli]
12.   client = yes
13.   accept = 127.0.0.1:6379
14.   connect = master.ssltest.xxxxxx.use1.cache.amazonaws.com:6379
15. [redis-cli-slave]
16.   client = yes
17.   accept = 127.0.0.1:6380
18.   connect = ssltest-002.ssltest.xxxxxx.use1.cache.amazonaws.com:6379
```

19. Start `stunnel`.

   ```
   sudo stunnel /etc/stunnel/redis-cli.conf
   ```

Output from the preceding command:

```
# netstat -tulnp | grep -i stunnel
tcp        0      0 127.0.0.1:6379              0.0.0.0:*
LISTEN      3189/stunnel
tcp        0      0 127.0.0.1:6380              0.0.0.0:*
LISTEN      3189/stunnel
```

20. Use redis-cli to connect to the encrypted redis node using the local endpoint of the tunnel.

    Using redis-cli:

    ```
    redis-cli -h localhost -p 6379 -a MySecretPassword
    ```

    Run redis-cli commands.

    ```
    set key1 value
    get key1
    "value"
    ```

    Using telnet:

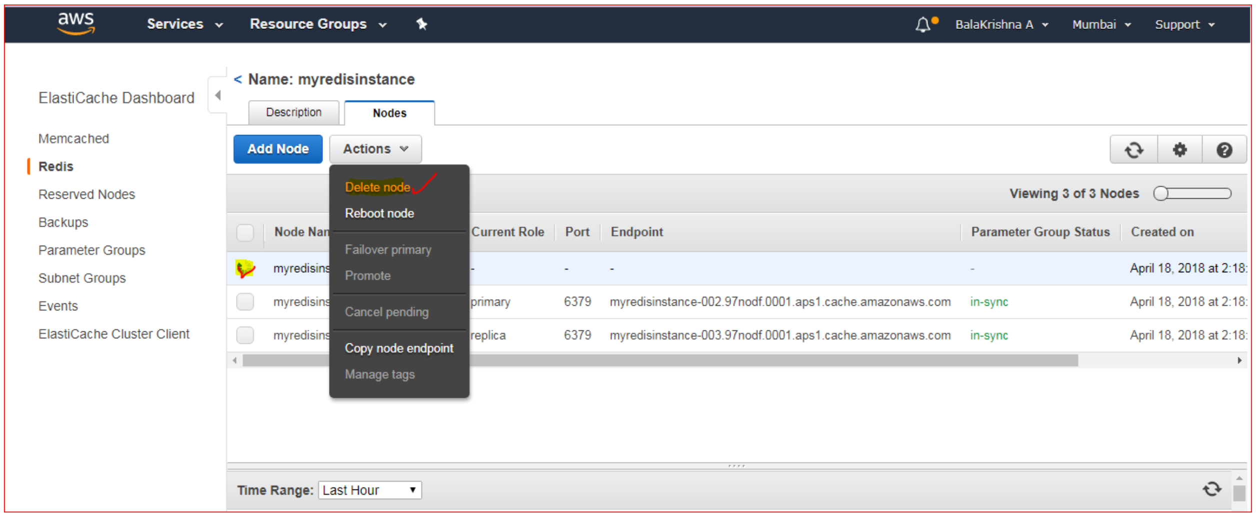
    ```
    telnet localhost 6379
    Trying 127.0.0.1...
    Connected to localhost.
    Escape character is '^]'.
    auth MySecretPassword
    +OK
    get key1
    $5
    value
    ```

21. Stop and close the SSL tunnel by killing the `stunnel` process.

    ```
    sudo pkill stunnel
    ```

Deleting Node:-

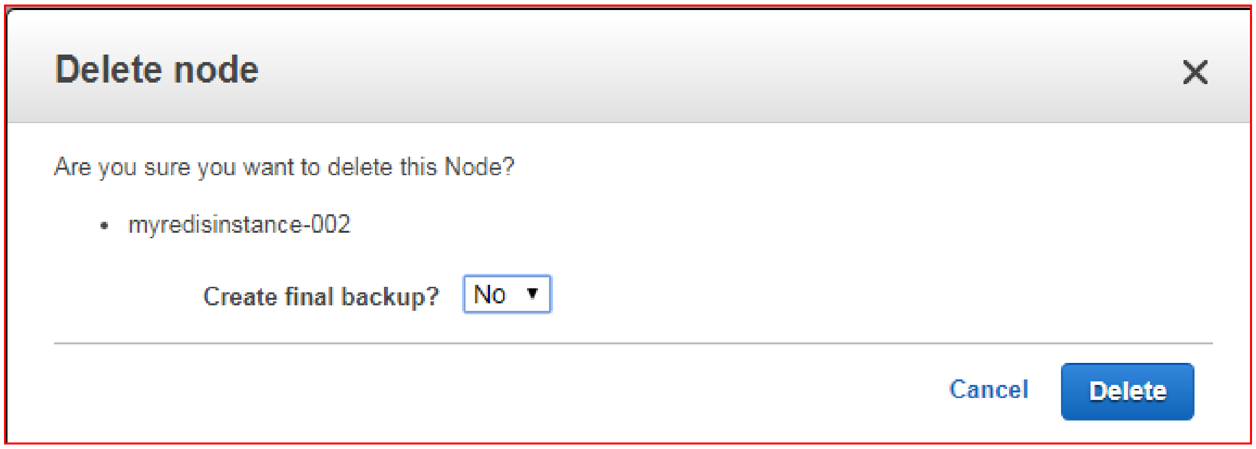
If you delete Primary Node, then the rest available slave node becomes primary Node, below is the example



< Name: myredisinstance

Viewing 3 of 3 Nodes

Node Nam	Current Role	Port	Endpoint	Parameter Group Status	Created on
myredisins	-	-	-	-	April 18, 2018 at 2:18
myredisins	primary	6379	myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:18
myredisins	replica	6379	myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 2:18

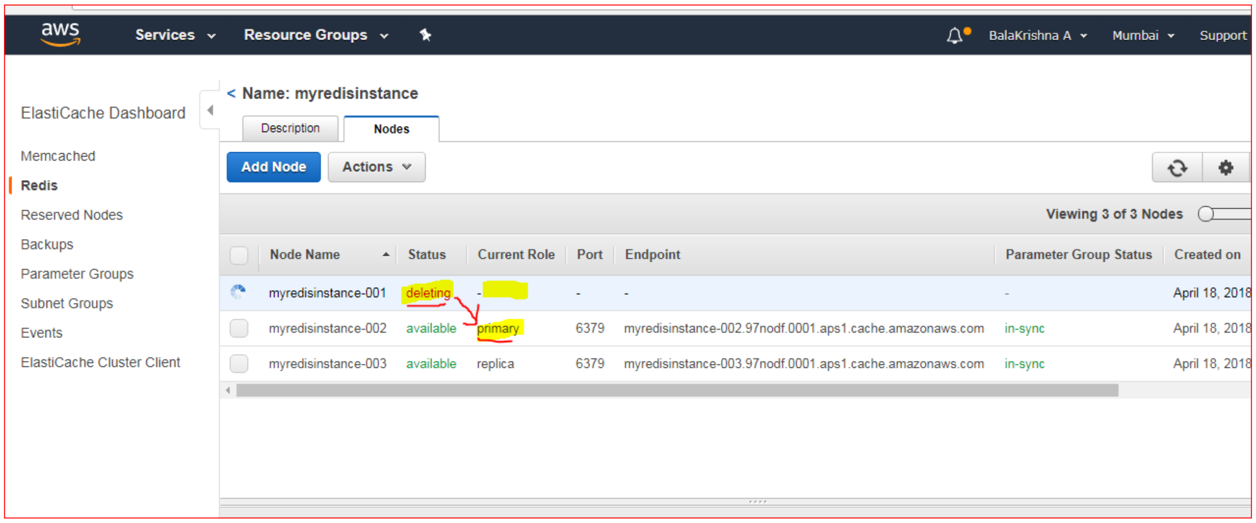


Delete node

Are you sure you want to delete this Node?

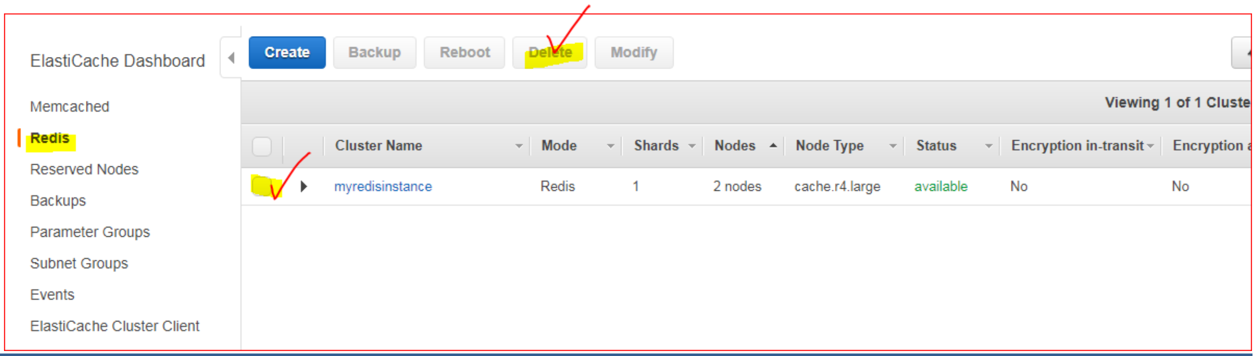
- myredisinstance-002

Create final backup? No

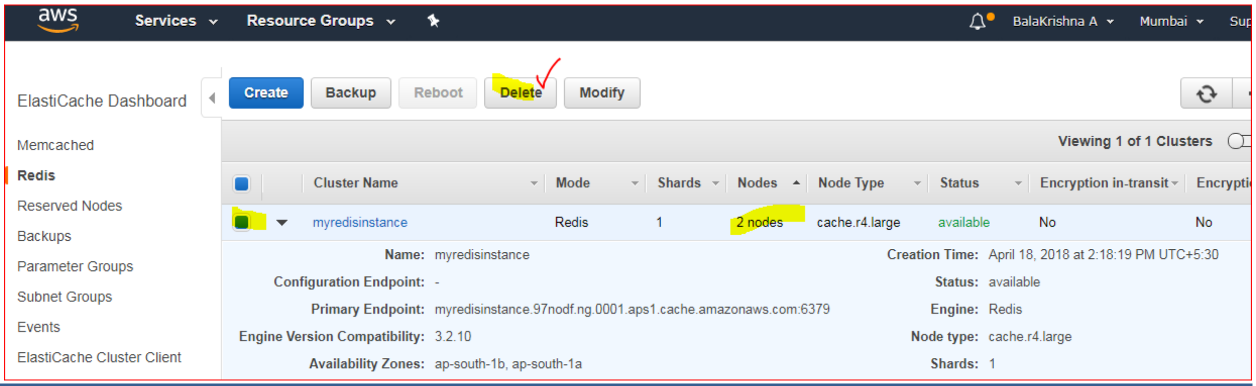


Node Name	Status	Current Role	Port	Endpoint	Parameter Group Status	Created on
myredisinstance-001	deleting	-	-	-	-	April 18, 2018
myredisinstance-002	available	primary	6379	myredisinstance-002.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018
myredisinstance-003	available	replica	6379	myredisinstance-003.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018

Delete a Cluster:-

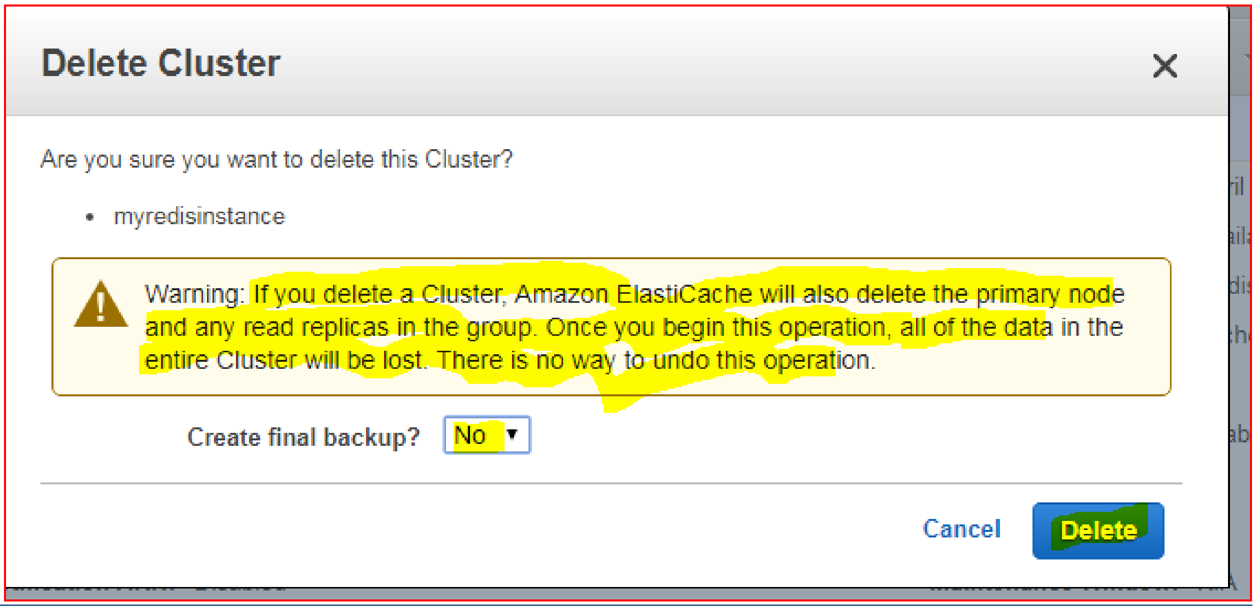


Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encryption a
myredisinstance	Redis	1	2 nodes	cache.r4.large	available	No	No



Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encrypti
myredisinstance	Redis	1	2 nodes	cache.r4.large	available	No	No

Name: myredisinstance
Configuration Endpoint: -
Primary Endpoint: myredisinstance.97nodf.ng.0001.aps1.cache.amazonaws.com:6379
Engine Version Compatibility: 3.2.10
Availability Zones: ap-south-1b, ap-south-1a
Creation Time: April 18, 2018 at 2:18:19 PM UTC+5:30
Status: available
Engine: Redis
Node type: cache.r4.large
Shards: 1

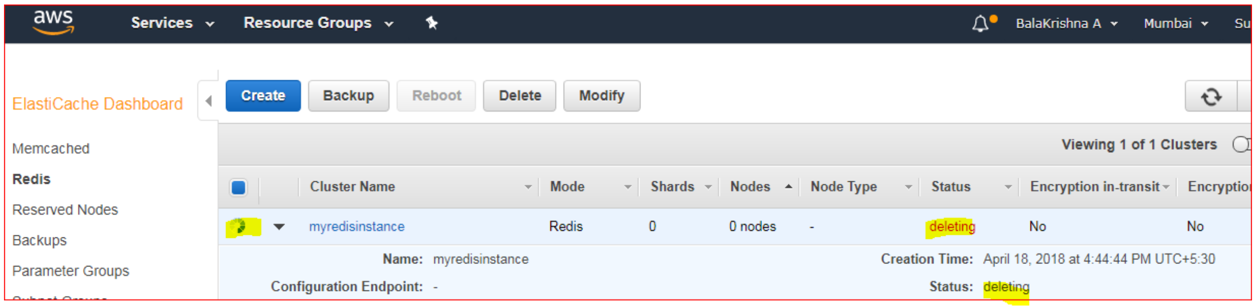

Delete Cluster

Are you sure you want to delete this Cluster?

- myredisinstance

Warning: If you delete a Cluster, Amazon ElastiCache will also delete the primary node and any read replicas in the group. Once you begin this operation, all of the data in the entire Cluster will be lost. There is no way to undo this operation.

Create final backup? No

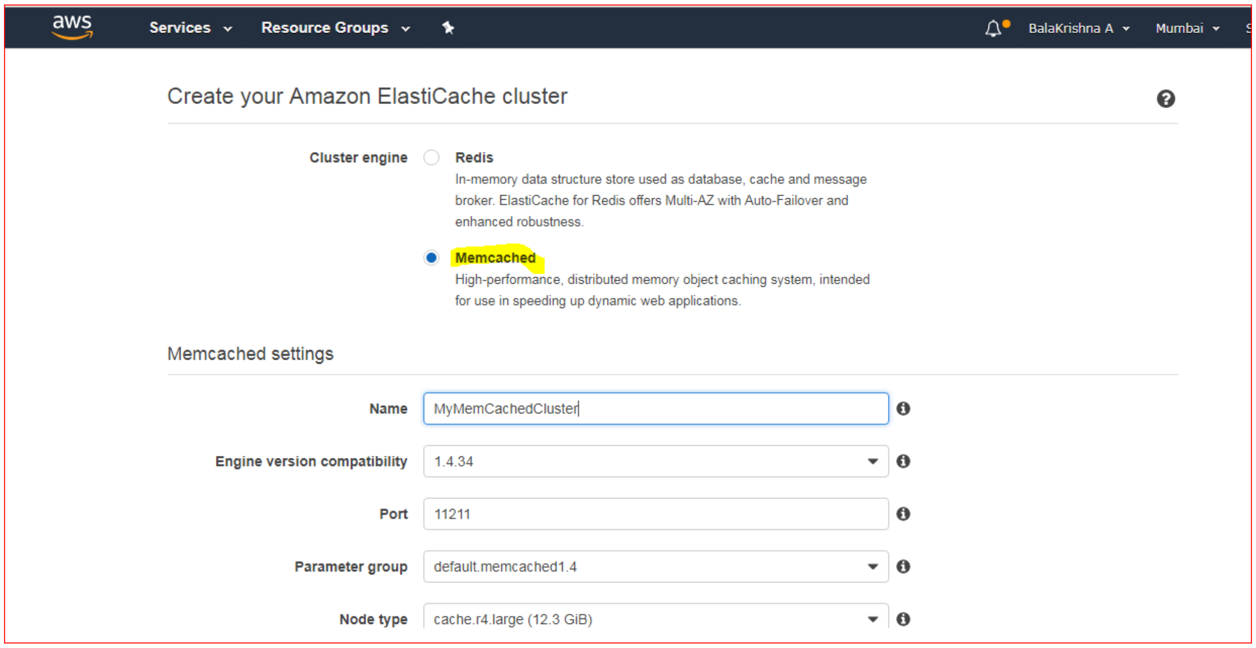


Viewing 1 of 1 Clusters

Cluster Name	Mode	Shards	Nodes	Node Type	Status	Encryption in-transit	Encryptio
myredisinstance	Redis	0	0 nodes	-	deleting	No	No

Name: myredisinstance
Creation Time: April 18, 2018 at 4:44:44 PM UTC+5:30
Configuration Endpoint: -
Status: deleting

ElastiCache :- Memcached



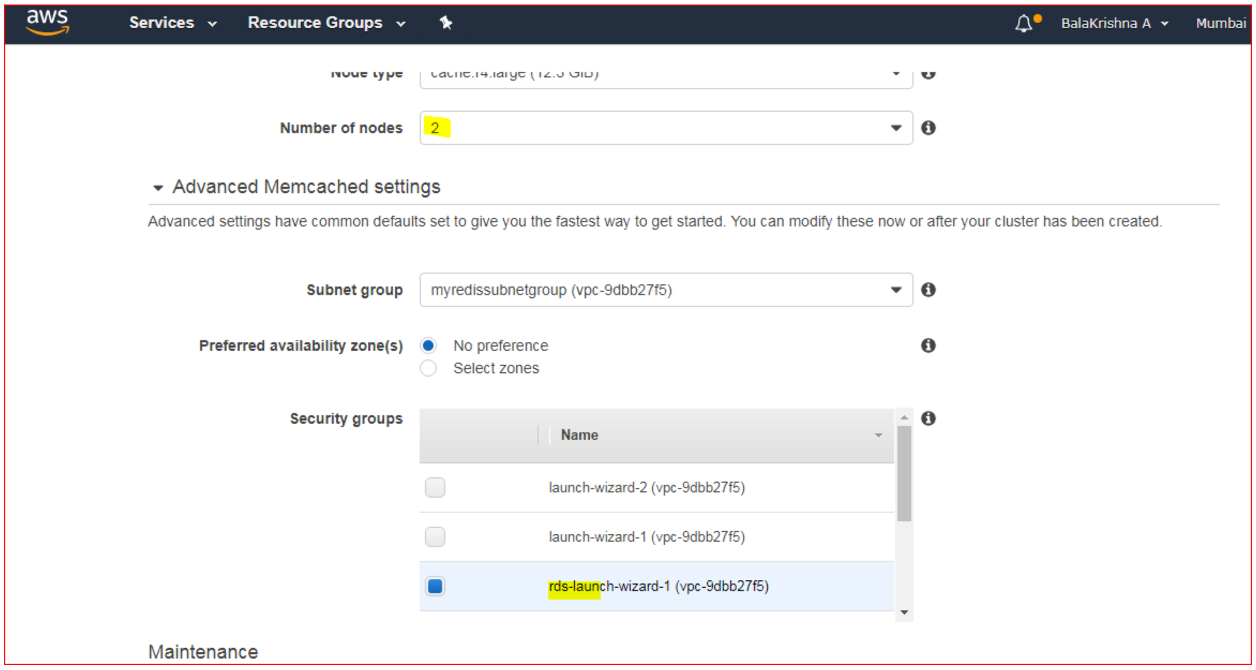
Create your Amazon ElastiCache cluster

Cluster engine

- ○ **Redis** In-memory data structure store used as database, cache and message broker. ElastiCache for Redis offers Multi-AZ with Auto-Failover and enhanced robustness.
- ● **Memcached** High-performance, distributed memory object caching system, intended for use in speeding up dynamic web applications.

Memcached settings

Name	MyMemCachedCluster
Engine version compatibility	1.4.34
Port	11211
Parameter group	default.memcached1.4
Node type	cache.r4.large (12.3 GiB)



Number of nodes	2

Advanced Memcached settings

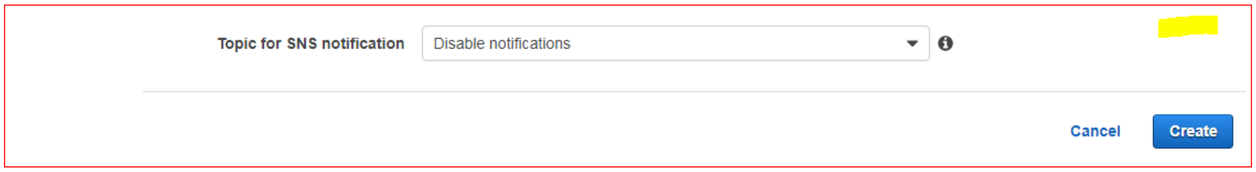
Advanced settings have common defaults set to give you the fastest way to get started. You can modify these now or after your cluster has been created.

Subnet group	myredissubnetgroup (vpc-9dbb27f5)
Preferred availability zone(s)	● No preference ○ Select zones

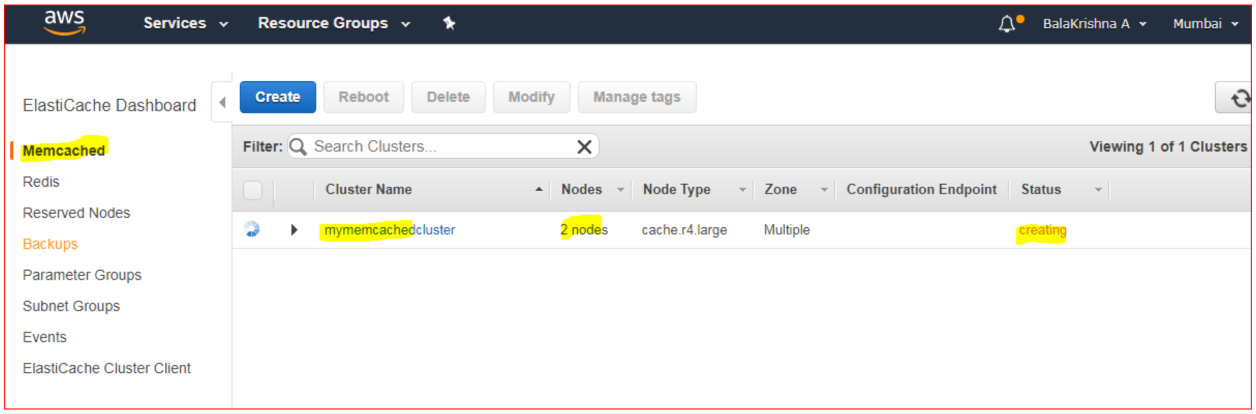
Security groups

	Name
☐	launch-wizard-2 (vpc-9dbb27f5)
☐	launch-wizard-1 (vpc-9dbb27f5)
☑	rds-launch-wizard-1 (vpc-9dbb27f5)

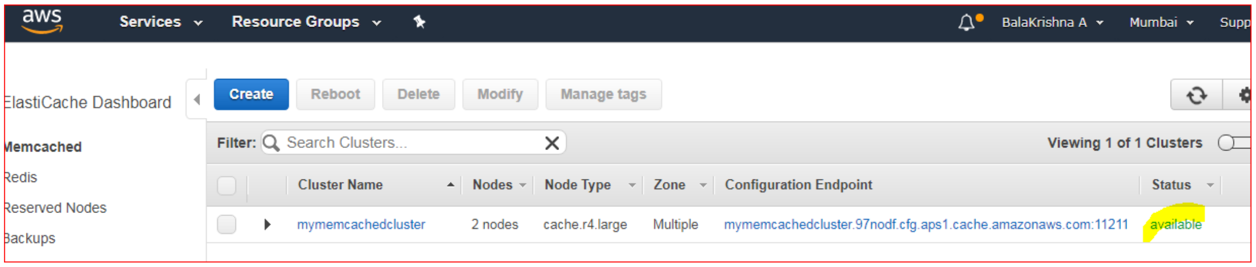
Maintenance



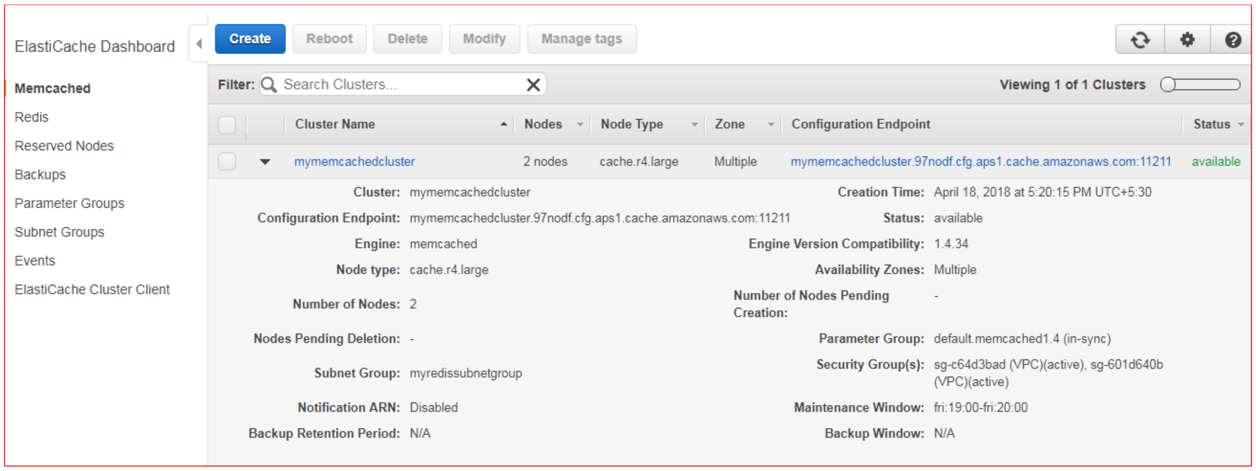
Topic for SNS notification Disable notifications



Cluster Name	Nodes	Node Type	Zone	Configuration Endpoint	Status
mymemcachedcluster	2 nodes	cache.r4.large	Multiple		creating



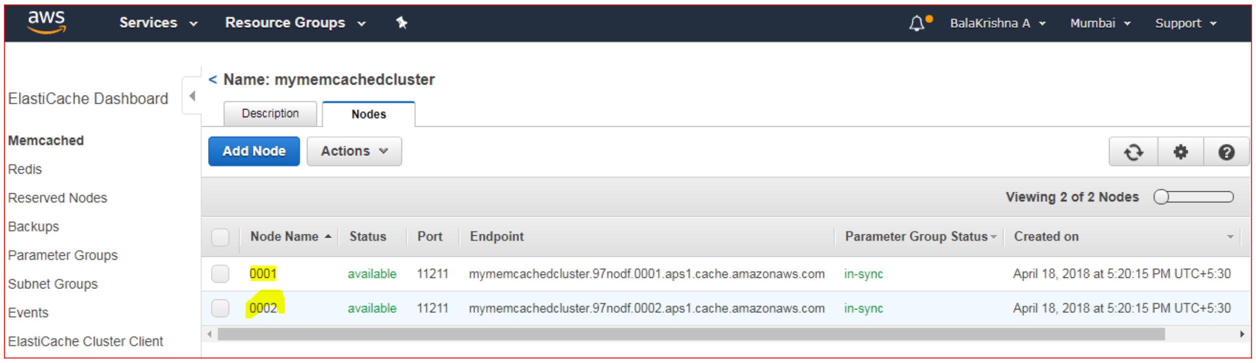
Cluster Name	Nodes	Node Type	Zone	Configuration Endpoint	Status
mymemcachedcluster	2 nodes	cache.r4.large	Multiple	mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws.com:11211	available



Cluster Name	Nodes	Node Type	Zone	Configuration Endpoint	Status
mymemcachedcluster	2 nodes	cache.r4.large	Multiple	mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws.com:11211	available

Cluster: mymemcachedcluster
Configuration Endpoint: mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws.com:11211
Engine: memcached
Node type: cache.r4.large
Number of Nodes: 2
Nodes Pending Deletion: -
Subnet Group: myredissubnetgroup
Notification ARN: Disabled
Backup Retention Period: N/A

Creation Time: April 18, 2018 at 5:20:15 PM UTC+5:30
Status: available
Engine Version Compatibility: 1.4.34
Availability Zones: Multiple
Number of Nodes Pending Creation: -
Parameter Group: default.memcached1.4 (in-sync)
Security Group(s): sg-c64d3bad (VPC)(active), sg-601d640b (VPC)(active)
Maintenance Window: fri:19:00-fri:20:00
Backup Window: N/A



< Name: mymemcachedcluster

Node Name	Status	Port	Endpoint	Parameter Group Status	Created on
0001	available	11211	mymemcachedcluster.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 5:20:15 PM UTC+5:30
0002	available	11211	mymemcachedcluster.97nodf.0002.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 5:20:15 PM UTC+5:30

Accessing Amazon ElastiCache

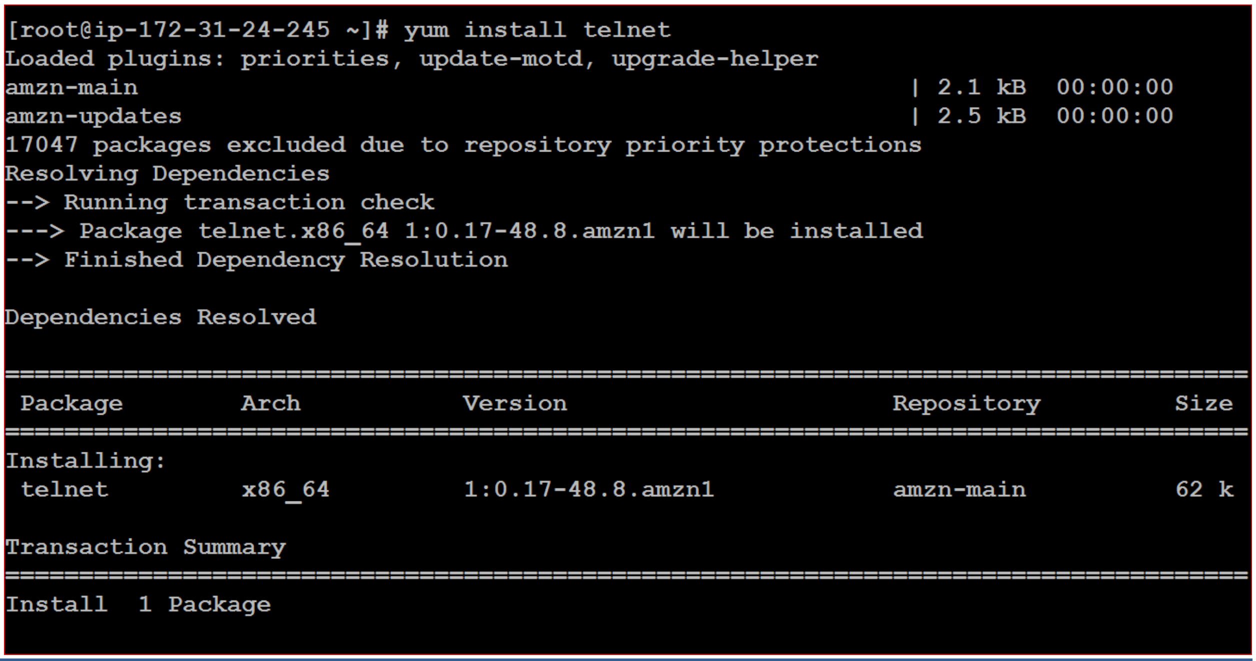
Your Amazon ElastiCache instances can only be accessed through an Amazon EC2 instance.

If you launched your ElastiCache instance in an Amazon Virtual Private Cloud (Amazon VPC), you can access your ElastiCache instance from an Amazon EC2 instance in the same Amazon VPC. Or, by using VPC peering, you can access your ElastiCache instance from an Amazon EC2 in a different Amazon VPC.

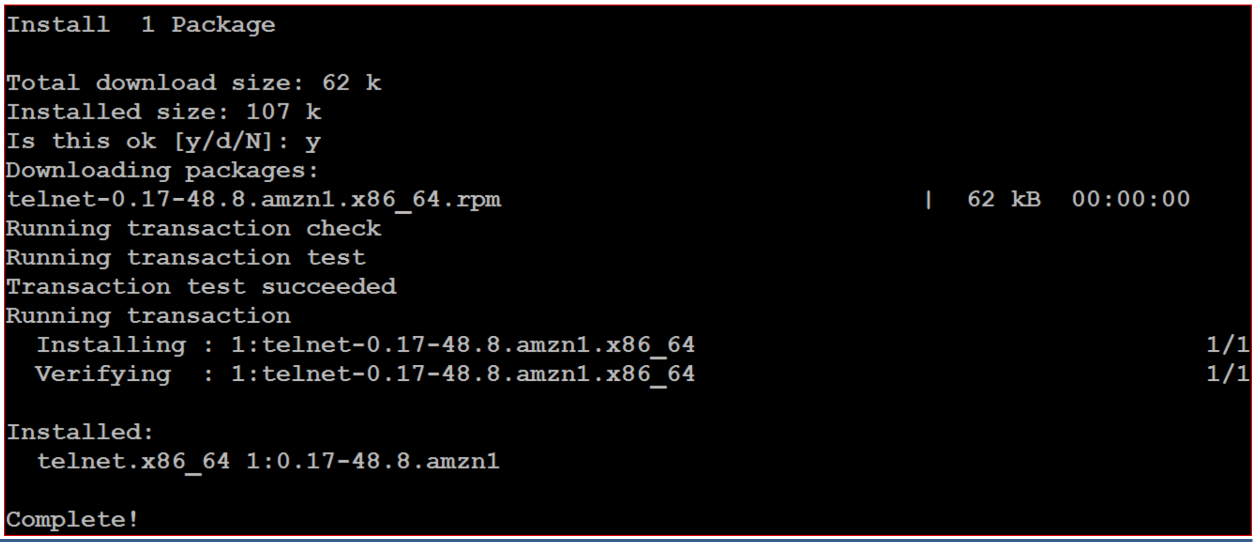
If you launched your ElastiCache instance in EC2 Classic, you allow the EC2 instance to access your cluster by granting the Amazon EC2 security group associated with the instance access to your cache security group. By default, access to a cluster is restricted to the account that launched the cluster.

Managing ElastiCache

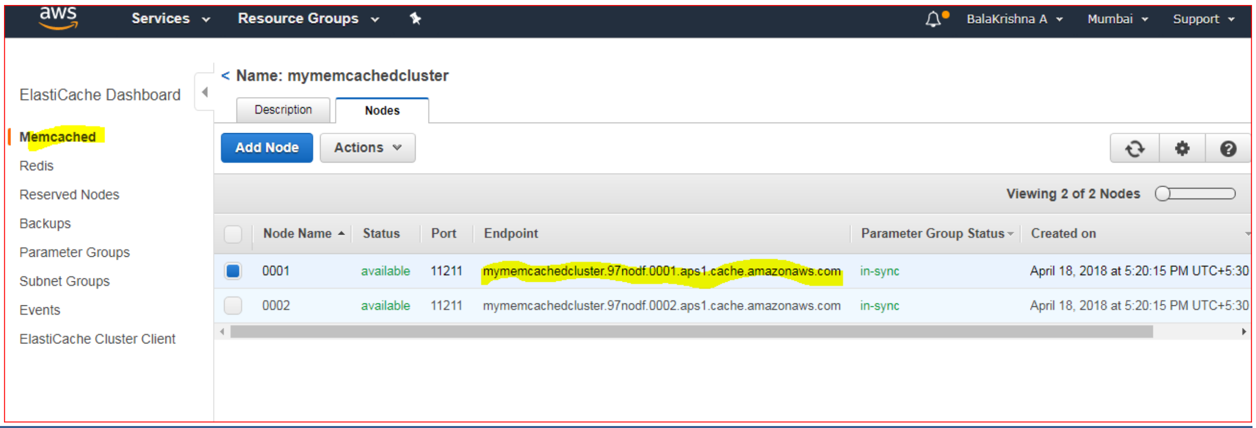
Once you have granted your Amazon EC2 instance access to your ElastiCache cluster, you have four means by which you can manage your ElastiCache cluster: the AWS Management Console, the AWS CLI for ElastiCache, the AWS SDK for ElastiCache, and the ElastiCache API.



```
[root@ip-172-31-24-245 ~]# yum install telnet
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-main                                                 | 2.1 kB  00:00:00
amzn-updates                                              | 2.5 kB  00:00:00
17047 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package telnet.x86_64 1:0.17-48.8.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package         Arch           Version                     Repository      Size
================================================================================
Installing:
 telnet          x86_64         1:0.17-48.8.amzn1           amzn-main       62 k

Transaction Summary
================================================================================
Install  1 Package
```



```
Install  1 Package

Total download size: 62 k
Installed size: 107 k
Is this ok [y/d/N]: y
Downloading packages:
telnet-0.17-48.8.amzn1.x86_64.rpm                          |  62 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:telnet-0.17-48.8.amzn1.x86_64                              1/1
  Verifying  : 1:telnet-0.17-48.8.amzn1.x86_64                              1/1

Installed:
  telnet.x86_64 1:0.17-48.8.amzn1

Complete!
```

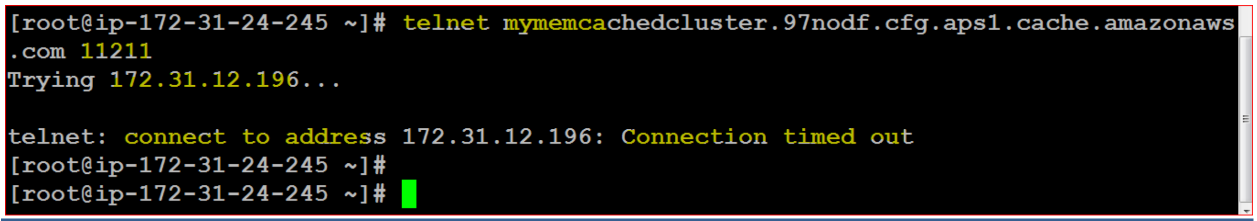


< Name: mymemcachedcluster

Viewing 2 of 2 Nodes

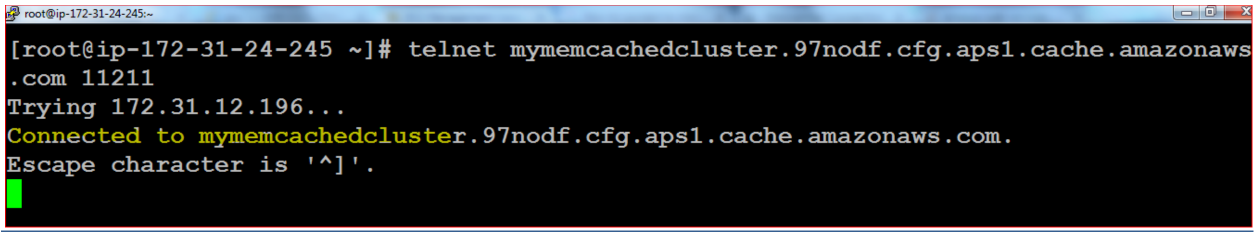
Node Name	Status	Port	Endpoint	Parameter Group Status	Created on
0001	available	11211	mymemcachedcluster.97nodf.0001.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 5:20:15 PM UTC+5:30
0002	available	11211	mymemcachedcluster.97nodf.0002.aps1.cache.amazonaws.com	in-sync	April 18, 2018 at 5:20:15 PM UTC+5:30

When you are trying to connect the EndPoint, here in below senarion getting connect timedout. One may be the reason need to verify the security Group and if you didn't allow the port 11211 that may be the reason for timed out exception.



```
[root@ip-172-31-24-245 ~]# telnet mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws
.com 11211
Trying 172.31.12.196...

telnet: connect to address 172.31.12.196: Connection timed out
[root@ip-172-31-24-245 ~]#
[root@ip-172-31-24-245 ~]#
```

Allow the port : 11211 in Security Group, once done then try to connect, it showing connected



```
[root@ip-172-31-24-245 ~]# telnet mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws
.com 11211
Trying 172.31.12.196...
Connected to mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws.com.
Escape character is '^]'.
```

You can store sample data, as memcached is key value store, you need to give the data in key value pair format

Below is the sytax for storing data for memcached

Memcached **set** command is used to set a new value to a new or existing key.

Syntax

The basic syntax of Memcached **set** command is as shown below –

```
set key flags exptime bytes [noreply]
value
```

The keywords in the syntax are as described below –

- **key** – It is the name of the key by which data is stored and retrieved from Memcached.
- **flags** – It is the 32-bit unsigned integer that the server stores with the data provided by the user, and returns along with the data when the item is retrieved.
- **exptime** – It is the expiration time in seconds. 0 means no delay. If exptime is more than 30 days, Memcached uses it as UNIX timestamp for expiration.
- **bytes** – It is the number of bytes in the data block that needs to be stored. This is the length of the data that needs to be stored in Memcached.
- **noreply (optional)** - It is a parameter that informs the server not to send any reply.
- **value** – It is the data that needs to be stored. The data needs to be passed on the new line after executing the command with the above options.

Output

The output of the command is as shown below –

```
STORED
```

- **STORED** indicates success.
- **ERROR** indicates incorrect syntax or error while saving data.

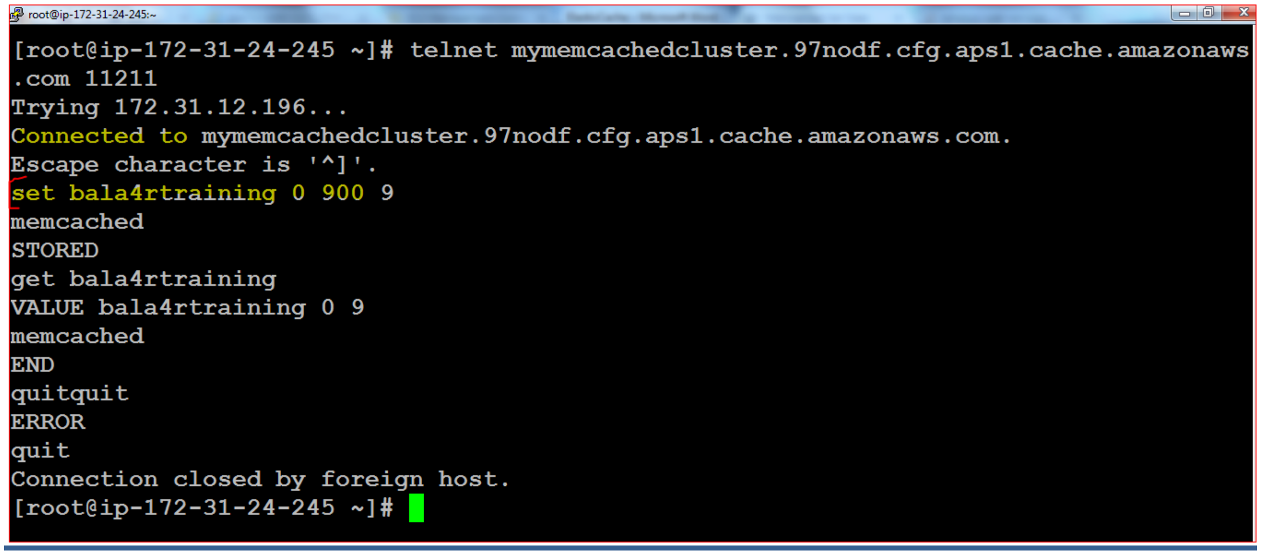
How to get the data

Memcached **get** command is used to get the value stored at key. If the key does not exist in Memcached, then it returns nothing.

Syntax

The basic syntax of Memcached **get** command is as shown below –

```
get key
```



```
[root@ip-172-31-24-245 ~]# telnet mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws
.com 11211
Trying 172.31.12.196...
Connected to mymemcachedcluster.97nodf.cfg.aps1.cache.amazonaws.com.
Escape character is '^]'.
set bala4rtraining 0 900 9
memcached
STORED
get bala4rtraining
VALUE bala4rtraining 0 9
memcached
END
quitquit
ERROR
quit
Connection closed by foreign host.
[root@ip-172-31-24-245 ~]#
```