CLOUD COMPUTING APPLICATIONS

AWS ElastiCache for Memcached

Prof. Reza Farivar

MemCached

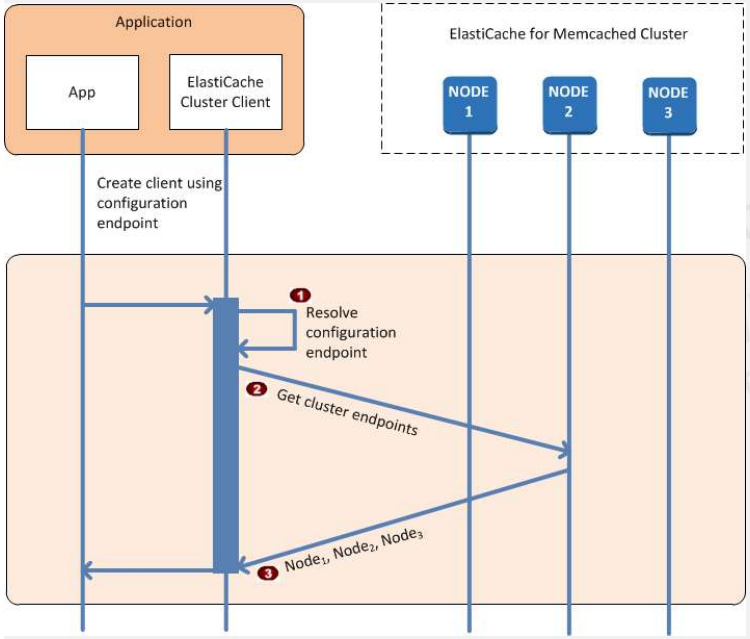
- Simple key/value storage model
  - E.g. in a Lambda Python function, just import pymemcache and set / get data
  - Other libraries: python-Memcached, pylibmc, twisted-Memcached, etc.
- Simple data model:
  - Strings, Objects
  - From the results of database calls, API calls, even HTML page renderings (e.g. PHP to HTML)
- No data storage
  - If a node goes down, you lose all the data in that node's memory
- Multi-threaded engine
- Multi-cluster using consistent hashing
  - Default limit 20 nodes
- Auto Discovery

ElastiCache Memcached Auto Discovery

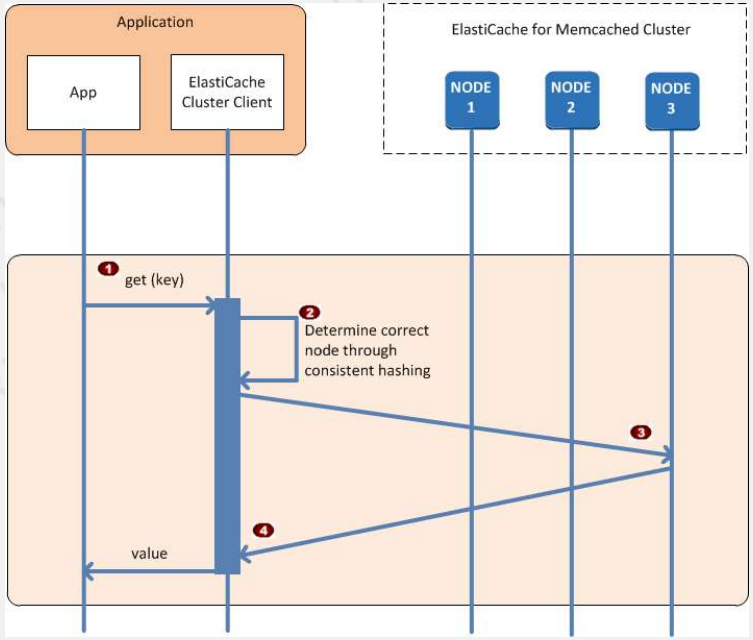
- Configuration endpoint
  - When you increase the number of nodes in a cache cluster, the new nodes register themselves with the configuration endpoint and with all of the other nodes
- When you remove nodes from the cache cluster, the departing nodes deregister themselves.
- In both cases, all the other nodes in the cluster are updated with the latest cache node metadata.
- Cache node failures are automatically detected; failed nodes are automatically replaced.
- A client program only needs to connect to the configuration endpoint.
- After that, the Auto Discovery library connects to all of the other nodes in the cluster.
- Client programs poll the cluster once per minute (this interval can be adjusted if necessary). If there are any changes to the cluster configuration, such as new or deleted nodes, the client receives an updated list of metadata. Then the client connects to, or disconnects from, these nodes as needed.

ElastiCache for Memcached Auto Discovery

Connecting to Cache nodes



Normal operation



Cluster Client available for Java, .Net and PHP

Example use of Elasticache for Memcached

```
from __future__ import print_function
import time
import uuid
import sys
import socket
import elasticache_auto_discovery
from pymemcache.client.hash import HashClient

#elasticache settings
elasticache_config_endpoint = "your-elasticache-cluster-endpoint:port"
nodes = elasticache_auto_discovery.discover(elasticache_config_endpoint)
nodes = map(lambda x: (x[1], int(x[2])), nodes)
memcache_client = HashClient(nodes)

def handler(event, context):
    """
    This function puts into memcache and get from it.
    Memcache is hosted using elasticache
    """

    #Create a random UUID... this will be the sample element we add to the cache.
    uuid_inserted = uuid.uuid4().hex
    #Put the UUID to the cache.
    memcache_client.set('uuid', uuid_inserted)
    #Get item (UUID) from the cache.
    uuid_obtained = memcache_client.get('uuid')
    if uuid_obtained.decode("utf-8") == uuid_inserted:
        # this print should go to the CloudWatch Logs and Lambda console.
        print ("Success: Fetched value %s from memcache" %(uuid_inserted))
    else:
        raise Exception("Value is not the same as we put :(. Expected %s got %s" %(uuid_inserted, uuid_obtained))

    return "Fetched value from memcache: " + uuid_obtained.decode("utf-8")
```