CumuLogic Elastic Cache Overview Guide

February 2013

Table of Contents

CumuLogic Elastic Cache Overview Guide

Introduction

CumuLogic Elastic Cache provides a fully managed instance of high performance, Memcached-compliant in-memory cache clusters on any private and public IaaS clouds and VMware virtualized environments. CumuLogic Elastic Cache offers the full functionality of Memcached in-memory cache technology without the need to manually manage, operate, monitor and scale the cache clusters in the clouds.

CumuLogic Elastic Cache automates the provisioning, configuration, performance optimization, management, failover, security and access control of the cache servers, thereby eliminating over 90% of the administration tasks required to manage distributed in-memory cache clusters in the cloud and virtualized environments. CumuLogic cache service also applies minor updates and patches automatically eliminating the need to schedule downtime for these management tasks.

CumuLogic cache service is integrated with CumuLogic PaaS, making it simple for developers to deploy applications in conjunction with database services to design scalable architectures.

Detailed Description

Memcached is the most popular object caching technology used to improve read performance of relational databases such as MySQL. It is an in-memory key-value store

for small chunks of arbitrary data (i.e. strings, objects) resulting from database calls, API calls or page rendering. Developers of large scale applications extensively rely on in-memory caching systems such as Memcached to read data from the cache instead of hitting the database for every read operation. If application workloads are read-intensive, Memcached eliminates latencies in application, provides high scalability and improves the user experience. The larger the size of available cache, the more data can be cached to avoid database-read operations or low latency disk-read operations. Memcached supports a distributed in-memory architecture, allowing developers to use system memory from several smaller systems or virtual machines as a cohesive single cache cluster. Distributed in-memory cache clusters provide low-cost options for database acceleration, scaling equally well as any commercial caching systems.

You can find more information about Memcached and a list of customers at Memcached.org.

With CumuLogic Elastic Cache, you can launch one or more instances or nodes of Memcached servers and start using the instances within minutes without the need for any application code changes. CumuLogic cache service manages the health of the cache nodes, monitors them for performance and can help optimize the parameters to fit the workload types. In case failure, the system will detect and re-provision the failed nodes to maintain the level of cache used by the applications.

CumuLogic Elastic Cache can scale out by adding additional cache nodes on-demand as needed. New nodes can be added by a single API call or command line call to CumuLogic PaaS or with a couple of clicks from the Developer Console.

CumuLogic cache service supports runs on any IaaS cloud, as well as VMware vSphere.

Service Features

Features supported by the CumuLogic Elastic Cache include:

Provisioning – Single click or single API call to provision the required number of nodes with the desired configuration and performance parameters optimized for the specific workload.

Pre-optimized nodes – Pre-optimized configuration parameters are used based on the amount of memory available on the node instance. The configuration parameters can be modified using parameter groups.

Failure detection and self healing – Service nodes are monitored in real-time for failures, as well as for performance and utilization metrics. In case of failure, the service controller will identify the failed nodes and will try to restart or re-provision the service. Self healing capabilities guarantee the availability of the appropriate amount of cache for the specific application.

On-demand scaling – Additional nodes can be added to the existing Memcached cluster by launching new nodes from the user interface, or a single API call or simple command line call. Applications can start using new nodes within minutes. The new nodes allow applications to scale during high peak loads without any impact to the performance and user experience.

Automatic minor updates and patches – Based on the user preference, the service controller will apply minor updates and patches to Memcached servers during the maintenance window defined by the user, minimizing downtime and manual operations.

Monitoring – Built in monitoring charts provide visibility into the usage and operation of the caching system and each node. Developers can visualize and optimize the configuration parameters to get the best cache performance.

Service Benefits

Minimal management tasks: CumuLogic Elastic Cache eliminates over 90% of distributed cache management tasks for developers and administrators, while providing the flexibility to control the performance, scalability and performance of the cache nodes.

Single-click deployment: It's very easy to spin up new cache nodes whether for development, QA/Testing or production purposes. Memcached instances can be provisioned using the developer UI or an API call. Deployment and lifecycle control of Memcached instances can be easily automated by using command line tools.

Low cost of operation: Memcached instances are fully managed and monitored, eliminating the need for manual installation, configuration, patching and scaling, thus lowering the cost of application operations by over 90%.

On-demand scalability: CumuLogic Elastic Cache can instantly scale instances by adding new cache nodes for handling all read-only database operations. Cache node sets can be added on-demand based on the current workload requirements and scaled down when not needed, therein eliminating costly over provisioning of resources.

Reliability: You can improve reliability by deploying multiple instances of clustered nodes. Multiple cluster nodes not only provide reliability and failover, but also greatly enhance the scalability of applications. Self healing features of cache nodes allow you to recover lost nodes and eliminate hiccups in application performance.

Security: Cache instances are secured using the firewall settings and security groups of IaaS clouds. Cache nodes can also be configured to only use secured connections.

Compatibility: CumuLogic Elastic Cache is fully compatible with standard open source versions of Memcached, therefore applications require no code changes when using the CumuLogic cache service.

Integrated with PaaS: Applications deployed on the CumuLogic platform can easily be configured to connect to any database instances managed by the CumuLogic database service, allowing cache nodes to be configured to be used in conjunction with the database nodes.

Multi-cloud support: CumuLogic's platform abstracts the underlying APIs of the different Infrastructure-as-a-Service clouds, including Citrix CloudPlatform, Apache CloudStack, OpenStack, Eucalyptus, VMware vCloud and vSphere. This enables users to deploy CumuLogic platform and cloud services on any of the supported private clouds and public clouds from the same platform.

How It Works

The main use case of Memcached is hosted web applications – more specifically *dynamic web applications* by eliminating or reducing the amount of time applications spend in reading data from the databases, most commonly relational databases. Read and write operations in relational databases are slow operations as they require reading data from a physical disk. In well architected applications, read operations can be moved from the database tier to the caching tier which is faster than the database tier. Most of the applications such as Wikipedia, LiveJournal and Twitter are read-intensive applications and use in-memory cache systems such as Memcached to reduce the number of reads from the database directly.

Working with CumuLogic Elastic Cache

With CumuLogic cache service, you can start by launching a cache cluster of the desired memory size and the number of virtual nodes using the User Console, an API call or a command line call. A cache cluster is a logical collection of individual Memcached

nodes. All virtual nodes present the system memory to form a collection of large memory chunk available for cache. To use this logical memory for cache, you can use any Memcached client in your application code.

When you deploy an application on CumuLogic PaaS, the cache nodes will be configured appropriately to allow you access to the cache in your application. You will require to source environment variables CL_CACHE_NODES to obtain IP addresses or DNS names of your cache nodes and start using them right away. All security and access group settings are configured and you need not worry about internal mechanism of the cache nodes. If you prefer to manually launch and configure the cache nodes, you can simply launch the nodes, configure the security and access groups, and you're ready to use the cache nodes in minutes.

Use the following steps to launch a cache cluster for either your existing applications or new applications:

Launching cache node or a cluster: From the CumuLogic User Console or command line tool, launch a cache node instance and select the number and type of instance you need. For example, if you choose a small instance node (most commonly 1.75GB or 2GB memory on public clouds), you will get approximately 1.5GB of memory for the cache. If you need more than 1.5GB, choose the appropriate number of cache node instances. The total memory for cache is equal to the sum of memory in each individual node.

Configuring Access: Configure security and sccess groups on nodes. By default, security groups or firewall settings are configured to close access to the cache nodes. You can use access groups to allow access to your cache nodes from your application servers, web servers or from the IP address of the nodes where you application is deployed.

Configuring Memcached client: In your Memcached client you will need to provide IP address or DNS names and port numbers of your Memcached nodes. The Memcached client will use a policy defined in your configuration to store data in the cache nodes. Refer to Developer Guide on our Documentation Center for details and sample code with Memcached service.

Monitoring and analyzing: Using CumuLogic Console, you can monitor the metrics for cache utilization such as cache hits, misses, etc. to help you optimize the cache configuration for the best performance.

Optimizing cache: You can use Cache Parameter groups to optimize certain performance parameters of cache nodes to further fine tune the cache.

Supported IaaS Clouds

CumuLogic Database Service is integrated and available for use on several common Infrastructure-as-a-Service clouds and virtualized environments including:

- Citrix CloudPlatform
- Apache CloudStack version 2.x and 3.x
- OpenStack Essex, Diablo and Folsom releases
- HP Cloud
- Rackspace Cloud
- Datapipe Stratosphere cloud
- Amazon EC2
- VMware vSphere 4.x
- VMware vCloud 5.1

CumuLogic Elastic Cache FAQs

What is in-memory caching and how can I use caching in my applications?

For applications using relational databases (RDBMS) such as MySQL or Oracle, each access to the database to read or write data involves latency due to disk I/O operation involved. Latencies in database access impact the scalability of the applications and user experience. To overcome such latency issues, the data that is most commonly used can be cached in-memory to avoid reading the same data from database. Applications can be written to read the data once from the database and store it in the cache, and then for subsequent reads, the stored data is read from the cache, which is a must faster operation. Write operations must however always be directed to the database directly.

Memcached is a popular caching technology, which provides cost-effective ways to implement caching on the clouds. Memcached is *distributed caching technology*, which means it can use memory from each virtual machine to scale up the size of the cache.

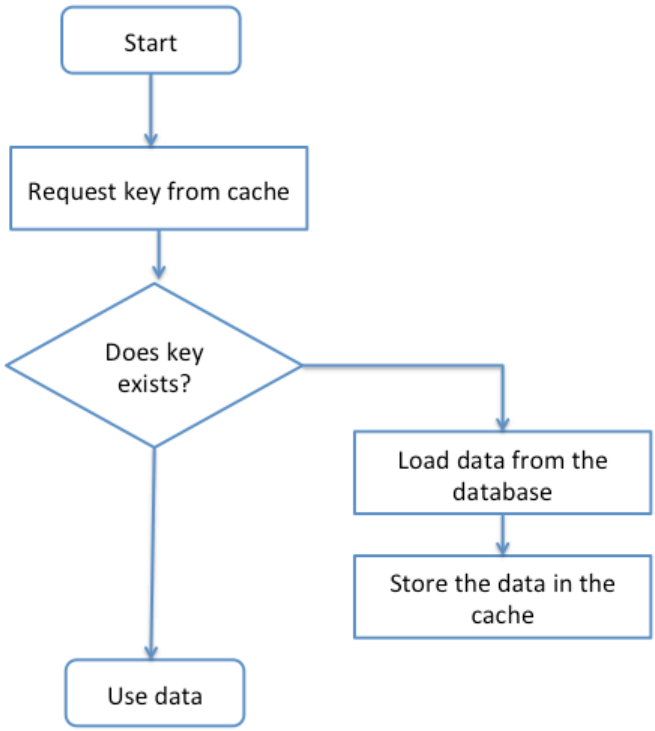

Figure 1: Using Memcached with your database applications

What is CumuLogic Elastic Cache?

CumuLogic Elastic Cache provides users with fully managed instances of high performance, Memcached compliant in-memory cache clusters on any private and public IaaS clouds and VMware vSphere. CumuLogic cache service provides easy access to Memcached in-memory cache technology without the need to manage, operate, monitor and scale the cache clusters in the clouds.

CumuLogic In-memory Cache Service automates the provisioning, configuration, performance optimization, management, failover and security and access control of cache servers, eliminating over 90% of administration task to manage distributed in-memory cache clusters.

What relational database is CumuLogic cache service compatible with?

CumuLogic cache service works independently of the relational database type. You will use cache to store data after you have read from the database using standard SQL queries. CumuLogic Elastic Cache is integrated with CumuLogic DbaaS.

Can I run a cache service on more than one IaaS cloud at the same time?

CumuLogic's platform supports multiple target IaaS clouds and each of the services, including in-memory cache, can be provisioned on more than one IaaS clouds. Please refer to "Supported IaaS Clouds" section for details for types of private clouds and public clouds supported in the latest release.

How is CumuLogic cache service integrated with other CumuLogic services?

CumuLogic platform is tightly integrated with all the services supported making it a breeze to deploy applications with orchestrated services required by the application. In most cases all that is required for deployment is a single-click or single command.

What type of data can I cache in CumuLogic Elastic Cache?

Because Memcached is a key-value pair caching technology, you can store any type of data or objects in the "value" portion of the cache. You can use Memcached for complex structures, images, documents or many other data types.

How does Memcached store data, and does it support namespaces?

Memcached does not support namespaces or allow compartmentalization of data into different sections. Application developers must use interfaces, which provide

automated mechanisms for creating namespaces when storing data. Alternatively, developers can use their own mechanism to tag the data with a unique identifier within the key supplied to store the value. For example, when storing company data, prefix the ID of the company with *company-* or *company#*. Please note that using namespaces does not necessarily provide data security, and it's still possible for multiple clients to access the same data in different namespaces. This mechanism only allows to prevent data corruption of key-value pairs.

How does data expiry work in Memcached?

There are two ways data is "expired" or removed from Memcached instances: time-based data expiration and space-based expiration. Time-based expiration is a user controlled mechanism to delete the data from the cache automatically if it hasn't been refreshed in a specific amount of time. This method ensures that stale data is flushed out of the cache. This is particularly useful when session data is cached in Memcached instances. When a user session has expired, it's recommended to delete the data from the cache as well.

The space-based expiration is the default behavior of Memcached servers. If you try to store new key-value pairs and there is not enough space available in the cache nodes, it removes the least recently used (LRU) objects from the cache.

How does Memcached store data in multiple Memcached nodes?

Memcached clients support several algorithms to choose the node in which to store the data when multiple cache nodes are available. Typically, Memcached clients use a hashing algorithm such as Ketama and Wheel. These are also called *consistent hashing* algorithms, which means they always use a hash of the key to be stored and identify the same server to store the data to. This eliminates inconsistent behavior when a new cache node is added or one cache node is removed from the cache cluster. The only downside to consistent hashing algorithms is when new servers are added. When new servers are

added, keys are redistributed amongst all nodes, and when one node goes down, all the lost keys have to be reallocated to existing nodes, causing a performance degradation during the redistribution of keys. The cache distribution is handled by Memcached clients and developers don't have to code anything in their applications.

How does memory allocation work in Memcached?

Memcached divides the allocated memory into chunks or slabs and each slab is divided up into blocks of suitable size, normally 1MB or more. Memcached will try to find a block to store a given key/value pair and if it finds one in available slab, it will use it. If the size of the data to be stored is larger than the size of the block, then a new slab is created and divided into suitable block sizes to store the data. If an existing key/value pair is updated and is bigger than the size of its existing block, then the updated data will be stored in new slab and a block of suitable size.

To avoid uneven distribution of data in different slabs, Memcached uses a growth factor that is applied to the minimum size of the current block to allocate new size of the blocks. For example, if the growth factor defined is 1.5 and the current size of the block is 100 bytes, then the new block size allocated will be 150 bytes. Having a growth factor too high or too low will cause bigger or smaller blocks of data to be created, and if the data to be stored is smaller than the block size, there will be substantial waste of memory used. You must look at the Memcached statistics on the CumuLogic User Console to fine tune the cache allocation parameters such as growth factor. CumuLogic Elastic Cache provides cache parameter groups to optimize these values for a given cache cluster and nodes.

How can I access Memcached logs?

CumuLogic Memcached service provides three levels of debugging data or log files but you must enable debugging of individual cache nodes. To generate Memcached debugging information, you must use one of three DebugParameterGroups to enable

the desired level of logging information, or you must add one of the debug flags mentioned below to your existing CacheParameterGroup. Three levels of verbosity supported are:

-v - Provides the lowest level of verbosity, as well as information on Errors and Warnings along with the information of client connections and their protocols.

-vv - This flag provides detailed information on client protocol operations, network operations and updates on keys. This level of debugging also provides information on the size of the slabs and block sizes. Use this flag primarily to get information on get and set data operations.

-vvv - This level of verbosity includes additional information including connections and transitions from one state to other. Use this option to debug any client-related issues and identify problems such as client connections dropping.

What Memcached statistics are available and how can I get access to them?

CumuLogic cache service provides several important statistics on Memcached servers, including the number of cache hits, cache misses, slab statics, item statistics and others related to data sizes and client connections. Please refer to CumuLogic Console for the complete list of metrics.

What are the criteria for choosing types and number of nodes in my cluster?

The number of nodes and type of cache nodes depend on the amount of data you wish to cache. Most of the IaaS clouds offer nodes with 2GB, 4GB, 8GB and beyond. If you need to setup a cluster with 16GB memory, you may want to select two 8GB nodes instead of one 16GB node. This provides additional fault-tolerance in case the node crashes or becomes unavailable. In general, you may want to use many smaller nodes rather than fewer larger nodes. This policy also helps when you need to scale out or add

additional nodes in which case, the cache cluster will redistribute the cache data to new nodes, and having smaller nodes will accelerate the redistribution process.

Is there a limit on the number of cache nodes I can run?

CumuLogic cache service does not limit the number of cache nodes you can use. However, some IaaS cloud providers may set a maximum limit on the number of VM instances you are allowed to launch. Please check with your IaaS cloud provider on any such limits on your account.

What protocols does CumuLogic Elastic Cache support?

You can use TCP or UDP to communicate with the cache nodes. CumuLogic cache service provides information on IP addresses and port numbers of cache nodes in environment variables for your application to source and establish a connection with the nodes. The cache node end point information is also provided on the CumuLogic Console under the "Hosts" tab of the cache service. Please note that Memcached provides no mechanism to authenticate the clients, so it's possible to connect by simply using the port numbers. By default, CumuLogic cache service blocks all the ports and you must use AccessGroups to open the firewall ports to allow a connection from your application hosts IP address.

How much memory does Cache Service allocate on different types of nodes?

Memecached on each node will try to allocate as much physical memory available on a given node. Its sage to assume over 90% of physical memory is allocated to cache but the amount of memory actually used depends on the size of the objects stored in the cache.

How can I find out the end point for each cache node in the cluster to use in the client?

The end point of cache nodes is available in environment variable CL_CACHE_NODES in your application, or by checking under the "Hosts" tab in CumuLogic Console. The format of CL_CACHE_NODES is as follows:

CL_CACHE_NODES:IP_ADDRESS:PORT,IP_ADDRESS:PORT

Below is a pseudo code for obtaining cache node hosts details in your application:

```
// Obtain Datasource information
String datasourceName = System.getProperty("GUESTBOOKAPP_CL_DATASOURCE");

//Obtain Cache Nodes information. The nodes is comma separated list of IP address:port number

String cacheHostString = System.getProperty("GUESTBOOKAPP_CACHE_NODES_STRING"); String[]
hosts = cacheHostString.split(","); //this is a array of all ips in cache cluster
```

What happens when a cache node memory is full?

When there is no available memory for cache, Memcached will use the *Least Recently Used* algorithm to delete the objects which have not been used recently to accommodate the new objects.

How does memory allocation work in Memcached?

If you need to cache larger amount of data, you may launch additional nodes and add to the existing cache cluster. Memcached service will automatically redistribute the keys between all the existing and new nodes. The addition of new nodes will require some time for the cache to warm up again.

Does CumuLogic Elastic Cache use large pages?

Yes, CumuLogic cache service supports large pages and can be modified by using CacheParamaterGroups.

Is there a limit on how many concurrent connections the cache nodes support?

CumuLogic cache service sets a default number of simultaneous connections to 1024. You can change this value by using CacheParameterGroup.

What is the maximum size of a object that can be stored in the cache?

The maximum size of a cache object is 1MB. If your object size is larger than 1MB, you may want to use compression, which is supported by most Memcached clients or split the data in more than one keys. If you prefer to change the size of the objects supported in your instance of Memcached nodes, you can change this value by using CacheParameterGroups.

How can I customize cache parameters for my custom workload requirements?

You can customize several parameters for your cache instances by using CacheParameterGroups. CacheParameterGroups are a set of configuration parameters that impact the performance and optimization of the cache service, which are exposed to users to manually define their settings. These parameters can be changed by the *power users* who are familiar with Memcached and its internal workings. Changing the parameters may also negatively impact the cache performance. If you wish to reset the configuration to the default parameters, you can edit a running cache instance and apply the default parameter groups provided for a given instance size.

How can I increase the cache size?

You can increase the cache size by launching additional cache nodes. Please note that cache service does not supports auto-discovery of cache nodes, hence you must configure your client manually to use the new node as part of cache cluster.

Is autoscaling supported with the cache service?

Autoscaling is currently not supported by the cache service. In the future, we will provide support for auto-discovery so Memcached clients will autoscale to launch new nodes when the cache on existing nodes is full.

What do I have to do in my application to manage the scale out of new nodes?

In your application, you must provide the IP address and port number of the cache nodes. If your application requires additional cache, you can launch a new cache node and obtain the IP address and port number of the new nodes by sourcing information from environment variable APPNAME_CL_CACHE_NODES which provides you the list of IP addresses and port numbers of all cache nodes available. APPNAME is the name of your application.

```
//Obtain Cache Nodes information. The nodes is comma separated list of IP address:port number

String cacheHostString = System.getProperty("GUESTBOOKAPP_CL_CACHE_NODES");

String[] hosts = cacheHostString.split(","); //this is a array of all IP addresses  in cache cluster
```

This will provide IP addresses such as:

192.168.1.1: 18080

192.168.5.10: 18080

You can use these IP addresses in your application as follows:

```
public class MyClass {

   protected static MemCachedClient mcc = new MemCachedClient();

   static {
              String[] servers =
         {
            "192.168.1.1:18080",
              '192.168.5.10:18080'
         };
              Integer[] weights = { 1 };
              SockIOPool pool = SockIOPool.getInstance();
         pool.setServers( servers );
         pool.setWeights( weights );

-.......
```

What happens to my data when one of the cache nodes fails?

If a cache node fails, the data stored in the cache is lost, we recommended you store only transient data in the cache. Persistent data store is always your database. Upon failure of a cache node in a cluster, the keys will be redistributed to the other available nodes, which could cause cache misses for a short duration of time.

Does CumuLogic Elastic Cache re-provisions new nodes to recover from failures?

Yes, CumuLogic cache service will restart Memcached server if possible, or will reprovision a new cache node to replace the failed one. The IP addresses of new node is available in the application's environment variable APPNAME_CL_CACHE_NODES.

Please note that the cache service does not support auto-discovery of the Memcached client to discover the new nodes. Developers receive a notification of a failed cache node and must take appropriate actions to use new cache node.

After the recovery of nodes, do I have to change the end point of the cache node?

Yes, you will be required to manually update the IP address of new node in your Memcached client or application code.

What version of Memcached does CumuLogic Elastic Cache support?

CumuLogic cache service is compatible with version ***1.4.5.***

How do I test my application with the latest version of the cache service engine?

You can launch a new cache instance of the version you wish to use and test your application against it before using the application for production purposes. CumuLogic Elastic Cache allows you to tag the cache instances as "Development" or "Production," which allows you to customize each of those environments using different security and performance parameter configurations.

Best practices of using Memcached

Here are some suggestions on using Memcached in your applications:

Read-heavy data: Use Memcached for applications which have mostly read-heavy workloads

Data type: Memcached is key/value pair cache and is best to use for 2-column tables where the first column is a primary-key which has only one item value. If you wish to use Memcached with multiple column values, choose your data types appropriately to be able to use a separator character between numeric values.

Query types: Use Memcached with simple queries with single WHERE clause with simple operators such as =, > and <.

Cache Size: Choose appropriate size of distributed memory cache for optimal performance. You can start with the amount of data you expect to cache and then add or remove additional cache nodes as needed to scale out your application.

Security: Memcached doesn't provides any security on connections or the data and you must use mechanisms to obscure your data by tagging keys. You must use Cache Parameter Groups to secure the connection ports and allow connections from the IP addresses where you run your applications or application nodes in CumuLogic platform.

Performance: You must use Cache Service Performance analytics regularly to review the cache performance and optimize the configuration parameters or your data size or type of data as needed.