**Service Definition Template**

1. **Service Name**

SimpliCache (Memcached)

1. **Service Options**
2. Distributed Memcached
3. Replicated Memcached
4. **Service Infra Roles**
5. **Define the Roles**
6. Master memcached (Distributed Memcached)
7. Master – Master memcached replication (Replicated Memcached)
8. **Define the Role specific Hardware Config for VM’s**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Plan Category** | **Role** | **Plans** | **vCores** | **vRAM** | **Disk1(GB)** |
| 1 | mcache.micro | Master | mcache.micro01 | 1 | 1 | 100 |
| 2 | Master | mcache.micro02 | 1 | 2 | 100 |
| 3 | mcache.small | Master | mcache.small01 | 2 | 8 | 100 |
| 4 | Master | mcache.small02 | 2 | 16 | 100 |
| 6 | mcache.medium | Master | mcache.medium01 | 4 | 16 | 100 |
| 7 | Master | mcache.medium02 | 4 | 24 | 100 |
| 8 | Master | mcache.medium03 | 4 | 32 | 100 |
| 9 | mcache.large | Master | mcache.large01 | 8 | 32 | 100 |
| 10 | Master | mcache.large02 | 8 | 48 | 100 |
| 11 | Master | mcache.large03 | 8 | 64 | 100 |
| 12 | mcache.xlarge | Master | mcache.xlarge01 | 12 | 48 | 100 |
| 13 | Master | mcache.xlarge02 | 12 | 72 | 100 |
| 14 | Master | mcache.xlarge03 | 12 | 96 | 100 |
| 16 | mcache.xxlarge | Master | mcache.xxlarge01 | 16 | 64 | 100 |
| 17 | Master | mcache.xxlarge02 | 16 | 96 | 100 |
| 18 | Master | mcache.xxlarge03 | 16 | 128 | 100 |

1. **Define the additional Infra requests like Disk, Disk type, Perf details etc.**

Not Applicable

1. **Service Specifications**
2. **Functional Specs**

Netmagic will setup online memcached servers in distributed and replication mode. The customer will connect directly to this setup using available memcached host & port.

Memcached is an open-source, high-performance, distributed memory object caching system which can be used to significantly improve latency and throughput for dynamic web applications with heavy workload.

It stores small chunk of random data in-memory in the form of key-value. It is extensively used for database applications, API calls or page rendering like e-commerce websites, social networking, media sharing and search engine.

Memcached intercepts data requests and satisfies a high percentage of them out of its logical cache (i.e. system memory), thereby avoiding trips to read data from back-end disk storage. Memcached also helps reduce compute-intensive tasks by retrieving pre-computed values from the logical cache, thereby avoiding the need for repetitive computations of the same value. In both cases, Memcached can reduce the overall time for responding to requests with cached data.

Key features of having memcached are:

1. Significantly reduces database work load as the information is stored in RAM. The access speed is faster than loading the information each time from the disk.
2. Memcached server is a large hash table because the value portion of the key-value pair does not have any data type restriction.
3. It can cache data such as complex structures, documents or images.

It is a read-only cache and the data information is also available in the database, the failure of any memcached server is not critical.

1. **Backup Specs**

Not required as it is an in-memory caching system

1. **Management Specs**

Consistent Hashing and Distribution

Tune Hashing algorithm to match keys.

Local replication using Replicache.

Deleting data from the server.

Determine if Key exists.

[Store, replace, add, or atomically add data to Memcache server](http://docs.libmemcached.org/memcached_set.html).

1. **Monitoring Specs**

**Monitoring Server Health**  
Memcache is generally low on CPU usage, will take as much memory as you give it, and network usage will vary from mild to moderate, depending on the average size of items. Avoid swapping as the performance will be very poor.

Monitor memcache performance by issuing sets/gets/deletes to the servers occasionally, timing the response of each. If it's slow to connect you might have a connection or network problem.    
**Monitor following stats**  
Connections, Memory Utilization, CPU Utilization, Stats items, Stats Slabs

1. **Service Parameters**
2. **Formula Based – Netmagic Suggested Best Practices**
3. **User defined – UI**

- ServiceType : {ServiceType:"Memcache"}

- ServiceVersion : {ServiceVersion:"1.4.30"}

- OSVersion : {OS\_Version:"Centos"} # Support Centos/RHEL/Ubuntu/Debain

- PORT : {PORT:"11211"}

- MAXCONN : {MAXCONN:"4000"} # [4000 \* no. of cores]

- THREADS : {THREADS :"1"} # [Based on no. of cores]

- CACHESIZE : {CACHESIZE:"800"} # [80% of RAM]

- USER : {USER:"memcached"}

- CHUNKSIZE : {CHUNKSIZE:"48"}

1. **Default Parameters**

- BINDINGPROTOCOL : {BINDINGPROTOCOL:"auto"}

- BACKLOGQUEUELIMIT : {BACKLOGQUEUELIMIT: "1024"}

- CHUNKSIZEGROWTHFACTOR : {CHUNKSIZEGROWTHFACTOR:"1.25"}

- HASHPOWER : {HASHPOWER:"hashpower=16"}

- MAXITEMSIZE : {MAXITEMSIZE:"1048576"}

- MAXCONNSFAST : {MAXCONNSFAST :"maxconns\_fast=yes"}

- ERRORONMEMORYEXHAUSTED : {ERRORONMEMORYEXHAUSTED:"-M"}

- REQUESTPEREVENT : {REQUESTPEREVENT:"20"}

- SLABAUTOMOVE : {SLABAUTOMOVE:"slab\_automove=0"}

1. **Service Administrative Parameters – Username, Password etc.**

Not Applicable

1. **Define the Various Rundeck Jobs with**
2. Commission of Virtual Machine
3. Tasks API From multiple task id (Device Status)
4. Start Virtual Machine
5. Get IP Details of VM
6. Memcache Basic Config
7. Memcache Cluster Config
8. Create Master to Slave Mapping
9. Create Service Component Mapping (Service to node mapping)
10. Memcache Testing
11. Memcache Commission Mail
12. **Parameters required**

**User Defined – UI**

- ServiceType : {ServiceType:"Memcache"}

- ServiceVersion : {ServiceVersion:"1.4.30"}

- OSVersion : {OS\_Version:"Centos"} # Support Centos/RHEL/Ubuntu/Debain

- PORT : {PORT:"11211"}

- MAXCONN : {MAXCONN:"4000"} # [4000 \* no. of cores]

- THREADS : {THREADS :"1"} # [Based on no. of cores]

- CACHESIZE : {CACHESIZE:"800"} # [80% of RAM]

- USER : {USER:"memcached"}

- CHUNKSIZE : {CHUNKSIZE:"48"}

**Fixed Parameters**

- BINDINGPROTOCOL : {BINDINGPROTOCOL:"auto"}  
- BACKLOGQUEUELIMIT : {BACKLOGQUEUELIMIT: "1024"}

- CHUNKSIZEGROWTHFACTOR : {CHUNKSIZEGROWTHFACTOR:"1.25"}

- HASHPOWER : {HASHPOWER:"hashpower=16"}

- MAXITEMSIZE : {MAXITEMSIZE:"1048576"}

- MAXCONNSFAST : {MAXCONNSFAST :"maxconns\_fast=yes"}

- ERRORONMEMORYEXHAUSTED : {ERRORONMEMORYEXHAUSTED:"-M"}

- REQUESTPEREVENT : {REQUESTPEREVENT:"20"}

- SLABAUTOMOVE : {SLABAUTOMOVE:"slab\_automove=0"}

1. **Source of parameters**
2. **User Interface**
3. **Service Database**
4. **Configuration file details**PORT="11211"

USER="memcached"

MAXCONN="4000"

CACHESIZE="800"

BACKLOGQUEUELIMIT="1024"

BINDINGPROTOCOL="auto"

CHUNKSIZE="64"

CHUNKSIZEGROWTHFACTOR="1.25"

REQUESTPEREVENT="20"

ERRORONMEMORYEXHAUSTED="-M"

MAXITEMSIZE="1048576"

MAXCONNSFAST="maxconns\_fast=yes"

SLABAUTOMOVE="slab\_automove=0"

HASHPOWER="hashpower=16"

THREADS="1"

OPTIONS="-vv >> /var/log/memcached 2>&1"

1. **Expected Error Conditions and Error codes**

00 = MEMCACHED\_SUCCESS  
01 = MEMCACHED\_FAILURE   
02 = MEMCACHED\_HOST\_LOOKUP\_FAILURE   
03 = MEMCACHED\_CONNECTION\_FAILURE  
04 = MEMCACHED\_CONNECTION\_BIND\_FAILURE   
05 = MEMCACHED\_WRITE\_FAILURE  
06 = MEMCACHED\_READ\_FAILURE  
17 = MEMCACHED\_MEMORY\_ALLOCATION\_FAILURE  
20 = MEMCACHED\_NO\_SERVERS  
31 = MEMCACHED\_TIMEOUT

1. **Expected Error condition handling process**

**# Memcache successfully installed and running**

00 = MEMCACHED\_SUCCESS

**# Memcache installed with errors. Telnet on default port**

01 = MEMCACHED\_FAILURE

**# Fail to resolve hostname. Ping hostname and check IP Address**   
02 = MEMCACHED\_HOST\_LOOKUP\_FAILURE

**# Fail to establish connection on default port. Telnet on default port**  
03 = MEMCACHED\_CONNECTION\_FAILURE

**# Fail to establish connection on bind ip address. . Telnet on default port**  
04 = MEMCACHED\_CONNECTION\_BIND\_FAILURE

**# Fail to write in memcache. Restart the service. Telnet on default port and try to write it again. In case it still fails reinstall memcache.**

05 = MEMCACHED\_WRITE\_FAILURE

**# Fail to read from memcache. Restart the service. Telnet on default port and try to write and read it again. In case it still fails reinstall memcache.**

06 = MEMCACHED\_READ\_FAILURE

**# Check if memory is not full and memcache is trying to write on swap. Need to allocate more memory for the client and temporarily restart memcache service.**  
17 = MEMCACHED\_MEMORY\_ALLOCATION\_FAILURE  
**# Check if memcache server IP address is reachable. Ping Ip address and telnet on default port.**

20 = MEMCACHED\_NO\_SERVERS  
**# Check if memcache server IP address is reachable. Ping Ip address and telnet on default port. Restart memcache service.**

31 = MEMCACHED\_TIMEOUT

1. **Define the various API requests for the specific Service**
2. Create Virtual Machine
3. Get Task
4. VM Operations
5. Get Infra details
6. Parameters API – UI
7. Parameters API – Fixed
8. Master to Slave Mapping
9. Service node Mapping
10. Commission Mail