OpenStack Hardware Requirements and Capacity Planning: Servers, CPU and RAM, Part 1

by [Ronen Kofman](http://www.stratoscale.com/author/ronen-kofman/) | Apr 6, 2016 | [OpenStack](http://www.stratoscale.com/blog/openstack/)

Determining hardware requirements for your [OpenStack cloud](http://www.stratoscale.com/solutions/the-cloud-and-openstack/) is not a trivial task; this complex process is **driven by workloads and cloud availability requirements**. At the same time, the variety of hardware available on the market makes it even more challenging to match your cloud’s requirements to specific options.

In the first part of this article we will define several basic principles to **enable prioritization and determine the CPU, number of servers and memory requirements** for your OpenStack cloud. In the second part we will focus on hardware storage and network requirements.

Compute nodes

In OpenStack, users’ VM’s run on the compute nodes, which are hypervisor hosts managed by OpenStack. To adequately choose the number and the configuration of such nodes, in general, translate the workload requirements to capacity measures, in terms of CPU, memory network and storage.

To learn more about deploying OpenStack, [click here](http://www.stratoscale.com/blog/openstack/openstack-deployment-tools-questions-and-options).

CPU requirements

Most modern CPUs support hardware virtualization (AMD-V technology provided by AMD, VT-x technology provided by Intel). Ensure that the CPU you choose and the hypervisor both support hardware virtualization. The most common CPU architecture for OpenStack cloud is x86-64.

Intel’s [Hyper-Threading](http://www.intel.com/content/www/us/en/architecture-and-technology/hyper-threading/hyper-threading-technology.html) feature improves parallelization, so having,  for example, a 12 core CPU with Hyper-Threading has about the same performance as 15-24 core CPU, depending on the workload.

To identify the number of servers, number of sockets and cores per socket you need to have the requirements for your workload. Usually you need a number of VMs,  a number of VCPU per VM and average expected performance of each VCPU. For example, you can calculate the total number of CPU cores you need:

**Cores = VMs / HT / CO**

Where:

* Cores – the total number of CPU cores you need
* VMs –  the total number of VMs you need to run
* HT – the Hyper-Threading coefficient

HT equals 1.3 if the CPU supports Hyper-Threading, and equals 1 if it does not.

* CO – the CPU oversubscription coefficient (1 for no oversubscription)

The number of CPU sockets:

**Sockets = Cores / Cores\_Per\_Socket**

Where:

* Sockets – the number of CPU sockets you need
* Cores – the total number of CPU cores you need
* Cores\_Per\_Socket – the number of cores per socket

And finally, the number of servers you need:

**Servers = Sockets / Sockets\_Per\_Server**

Where Sockets\_Per\_Server is the number of sockets per server.

Example:

* You need to run 100 VMs:

VMs = 100

* You will not use Hyper-Threading

HT = 1

* We will use CPU oversubscription equals to 2, which means that in average we want to use a single CPU core (for example, 2.4GHz) for 2 VCPUs.

CO = 2

* The total number of CPU cores you need:

Cores = 100 / 1 / 2 = 50

* You prefer to use 8 core CPU:

Cores\_Per\_Socket = 8

* The number of CPU sockets you need (rounded to the next integer number):

Sockets = 50 / 8 = 7

* You prefer to use 2 CPU sockets per server:

Sockets\_Per\_Server = 2

* The number of servers you need (rounded to the next integer number):

Servers = 7 / 2 = 4

Therefore, to run 100 VMs with 2 virtual CPUs each, without Hyper-Threading and CPU oversubscription, you will need 4 servers for the compute nodes. Each server will contain 2 CPU sockets and 8 2.4GHz CPU cores per socket.

Notes:

* If you have several types of VMs, calculate the total number of CPU cores per type, then sum them up.
* For the servers with NUMA you should be careful about CPU and memory pinning to ensure that each VM consumes CPUs and memory from the same NUMA node, otherwise the VM’s performance will be degraded. If a VM requires many virtual CPUs, using the NUMA node pinning results in a bigger number of physical servers required to run such workload.

Memory requirements

You can use the following formulas to calculate memory required for the compute node:

**VMs\_Per\_Server = VMs / Servers**

**RAM = RAM\_Per\_VM \* VMs\_Per\_Server / MO + OS\_RAM**

Where:

* VMs –  the total number of VMs you need to run
* Servers – the number of servers you need
* VMs\_Per\_Server – the calculated number of VMs per server
* RAM\_Per\_VM – RAM per VM
* MO – the RAM oversubscription coefficient (1 for no oversubscription)
* OS\_RAM – RAM required for the operating system and the hypervisor

Note, that the hypervisor can also use system memory for disk caches. The amount of RAM required for the disk caches for all of the VMs is hard to estimate: it depends on the number of VMs on the host, the number of disks for each VM and type of cache for each disk.

**Example:**

* You need to run 100 VMs:

VMs = 100

* Each VM requires 4GB RAM:

RAM\_Per\_VM = 4

* You will use 4 servers from the example above:

Servers = 4

* The number of VMs per server:

VMs\_Per\_Server = 100 / 4 = 25

* No RAM oversubscription:

MO = 1

* You will use 16GB RAM for the operating system

OS\_RAM = 16

* RAM required for the compute node:

RAM = 4 \* 25 / 1 + 16 = 116

Therefore, to run 100 VMs with 4GB RAM each, without RAM oversubscription and disk caches you will need 116GB RAM on each compute node.

To reduce the amount of physical RAM required to run the VMs you can use memory de-deduplication techniques per host. For example, Transparent Page Sharing (TPS) for Xen and has Kernel Samepage Merging (KSM) for KVM (see also[KVM Hypervisor: Memory management of over-commitment](http://www.stratoscale.com/blog/compute/kvm-hypervisor-memory-management-of-over-commitment/)). If all of the VMs on the host are based on the same image, then they have a lot of common pages, which can be de-deduplicated.

Other nodes

Once the requirements for the compute nodes are determined, it is time to think about servers for the [OpenStack](http://www.stratoscale.com/blog/openstack/openstack-tutorial-upgrading-pitfalls-and-solutions/)services and potentially about other nodes you may need for your OpenStack cloud, such as separate storage nodes. In general, there are two parameters for the OpenStack services that are relevant to the hardware planning:

* How many of the service’s instances will be deployed (service redundancy)
* Where are the service instances that will be deployed (deployment mode)

Services redundancy

For a large cloud footprint you will need multiple service instances and also to set load balancing for requests for the services. Also you will need more than one instance of each service to support high availability.

Deployment mode

* **Segregated services mode:** assumes that there are dedicated hardware nodes for the OpenStack services and that each node contains all the required services. This mode is the **simplest to implement and manage**. However such dedicated nodes can be not fully utilized.
* **Converged services mode:**assumes that OpenStack services are deployed on the existing nodes. Potentially, as an extreme case, each node in the cloud can contain all of the required OpenStack services in addition to the running workloads. This mode does not necessary require dedicated servers, but the cloud nodes should have additional CPU and RAM resources to run OpenStack services. **Resources for the workloads and for the controlling services should be properly isolated** to avoid performance degradation. Needless to say, this mode is harder to implement and manage than the segregated mode.

Learn about [High Availability](http://www.stratoscale.com/blog/openstack/openstack-liberty-high-availability-overview-and-guidelines-part-2/) in OpenStack Liberty

The required hardware resources for the individual services depend upon many parameters, including:

* The number of service instances running in the cloud and on the specific node
* Which other services are running on the same node
* The number of compute nodes
* How the cloud will be used

Here we will provide very common recommendations for individual services.

Operating system

The amount of memory, disk space and required resources depends on the operating system and system services that are running on the node. In general, we recommend using at least 2 CPU cores and 16GB of RAM for the operating system, system services and file caches.

RabbitMQ

[RabbitMQ](https://www.rabbitmq.com/) requires at least 128MB of RAM and by default will use up to 40% of available RAM. With the segregated services mode you need to decide how much memory you will reserve for RabbitMQ. By default, RabbitMQ requires 50MB of free disk space at all times and we recommend at least 2GB of free disk space. The more queues and consumers you need to support the more memory and disk space you need to reserve for RabbitMQ.

MySQL with Galera

Each[Galera](http://galeracluster.com/) node requires at least 1GHz CPU core and 512MB of RAM. For the production grade clouds you will need at least 2 CPU cores and 4-8GB RAM.

OpenStack services

Depending on the service, you need to have 0.5 – 2 CPU cores and 2-4GB RAM for each service.

For simplicity of calculation, let’s use the segregated services mode. In this case we would recommend using the following configuration:

* An odd number (at least 3) of dedicated hardware servers
* Each server should contain at least 2 CPU sockets with 6 CPU cores (x86-64) per socket
* Each server should contain at least 64GB RAM

This configuration permits an OpenStack cloud with 1000+ VMs. Note, that you can add new compute nodes as your cloud grows. Also, with HA architecture you can start with one small controller node (2 CPU cores, 2-4 GB RAM), which is not highly available by itself, but all of the services it contains are “HA-ready”, allowing you to add controller nodes with no downtime.

Additional nodes

Take into consideration that the following additional nodes are required in some cases for OpenStack clouds:

* Optional OpenStack director/master node to store the master configuration, orchestrate OpenStack installation and manage the installed cloud (such node is required by some deployment tools from OpenStack vendors)
* Optional separate nodes for database (3 for HA configuration)
* Optional separate nodes for monitoring the cloud (3 for HA configuration)
* Optional separate nodes for centralized logs storage (3 for HA configuration)

If you are planning to use a staging/testing cloud to verify all of the updates you are going to apply to the production cloud, ideally the number of the nodes in the production cloud should be doubled.

You can optimize the number of nodes for the staging/testing cloud, but the optimization should be smart, because you may want to have the same number of regions and availability zones that you have for the production cloud. Also if you are planning to **test the staging cloud’s performance and do its benchmarking, then the configuration of the staging cloud should be similar to the production one, including the number of nodes**.

In the second part we will focus on hardware planning for OpenStack cloud’s storage and network