Demystifying the Number of vCPUs for Optimal Workload Performance

September 2018

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Abstract

Following industry standard rules of thumb when migrating physical servers or desktops into a virtual environment doesn't ensure optimal CPU performance after consolidation, especially for CPU-intensive workloads. This paper describes a proven scientific methodology for benchmarking CPU performance for different CPU generations, with detailed examples, to achieve optimal performance. Learn how to choose Amazon EC2 instance types based on CPU resources and apply best practices for CPU selection with Amazon EC2.

Introduction

When you migrate physical servers or desktops to a virtual environment using a hypervisor (such as ESX, Hyper-V, KVM, Xen, etc.), you're typically advised to follow industry standard rules of thumb for high-workload consolidation. For example, you might be advised to use 1 CPU core for every 2 virtual machines (VMs). However, this ratio might not provide a realistic estimate for CPUs with high clock speeds, such as those running at 1.6 GHz to 3.3 GHz.

You should use a higher consolidation ratio with faster CPUs. New generation CPUs provide better performance, even when running at the same clock speed or with the same number of CPU cores, compared with prior generation CPUs. The price-performance ratio with new CPUs is better as well.

So how do we benchmark the CPU performance for different CPU generations to get the optimal performance after VM consolidation?

As part of the answer, and to ensure predictable results, we should have a scientific approach to determine the most appropriate CPU sizing. Remember that ***undersizing a CPU resource can cause poor user experience*** and ***oversizing a CPU resource can cause wasted resources and higher Operating Expenses (OPEX), yielding a higher Total Cost Ownership (TCO)***.

This paper examines a proven methodology for choosing the right Amazon Elastic Compute Cloud (EC2) instance types based on CPU resources and includes detailed examples. In addition, some best practices for CPU selection with Amazon EC2 are discussed.

Methodology

Step 1: Normalize the CPU performance index (Pi) for different generation CPUs using the Moore's Law equation[1]:

$$P_i(t) = 2^{0.05556(t)} \qquad (1)$$

Where,

P_i (t) is the CPU performance index at the reference month t = 0.

In other words, if we're trying to migrate a system with a CPU_A being first sold on Jan 2015 to CPU_B being first sold on June 2016, then the performance index for CPU_A is P_i (0) = 1 and CPU_B is P_i (18) = 2.

Step 2: Determine the normalized CPU utilization, in terms of clock speed (GHz), of the current workload utilization by inserting Equation (1) into Equation (2). The normalized CPU utilization (CPU Utilization $_{(Norm.)}$) equation will be explained as shown below:

$$CPU\ Utilization_{(Norm.)} = [\#CPU \times \#Core \times CPU\ Freq. \times CPU\ Utilization \times P_i(t)] \qquad (2)$$

Where,

- #CPU = Current number of CPU sockets per physical server. If it is a VM, it should be equivalent to 1.
- #Core = Current number of CPU cores per physical server. If it is a VM, it should be equivalent to the number of currently deployed vCPUs. (We are assuming that there is no oversubscription in this case.)
  If hyper-threading is enabled, the number of CPU cores or vCPUs should be doubled.

[1] In the mid-1960s Gordon Moore, the co-founder of Intel, made the observation that computer power measured by the number of transistors that could be fit onto a chip, doubled every 18 months. This law has performed extremely well over the preceding years.

- CPU Freq. = Current CPU clock speed, in GHz.
- CPU Utilization = Current CPU utilization, as a percentage.
- $P_i(t)$ = Performance index for vCPUs, per month.

Step 3: Determine the estimated CPU utilization by reserving sufficient buffer for a workload spike. This is calculated by inserting the required headroom, in terms of percentage (%), into Equation (3). This gives a conservative estimate of the CPU sizing to avoid suboptimal performance. The estimated CPU utilization (CPU Utilization $_{(Est.)}$) equation is explained as shown below.

$$CPU\ Utilization_{(Est.)} = CPU\ Utilization_{(Norm.)} \times (1 + Headroom) \quad (3)$$

Where,

$Headroom$ = Percentage of CPU resource reserved as a buffer for a workload spike.

Step 4: Refer to Amazon EC2 Instance Types to find the most appropriate CPU type for particular instance classes by using Equation (4).

$$CPU\ Utilization_{(Est.)} \leq CPU\ Capacity_{(new)} = \left[\frac{\#vCPU_{(new)}}{2} \times CPU\ Freq_{(new)} \times P_{i(new)}(t)\right] \quad (4)$$

Where,

- $\#vCPU_{(new)}$ = Newly selected number of vCPUs for the Amazon EC2 instance. It is divided by 2 since hyper-threading is used on the Amazon EC2 instance.
- $\#CPU\ Freq_{(new)}$ = Newly designated CPU clock speed (GHz) for the Amazon EC2 instance.
- $P_{i(new)}(t)$ = Performance index for new vCPUs per month.

Discussion by Example

Step 1: Table 1 shows the performance index, which is calculated by using Equation (1), for various CPU models. The oldest CPU model, Xeon E5640, is used as the benchmark. Both the Xeon E5640 and E5647 models belong to the current state of usage.

CPU Model	CPU Frequency (GHz)	# Cores	First Sold	Performance Index	Performance Index Per Core
Xeon E5640	2.67	4.0	Mar-10	1.00	0.25
Xeon E5647	2.93	4.0	Feb-11	1.53	0.38

Table 1: CPU Performance index for various CPU models

Step 2: Table 2 shows the total CPU utilization, in GHz, after using Equation (2) for all the physical servers' workloads that will be migrated to Amazon EC2.

Host Name	CPU Model	CPU Freq. (GHz)	# CPU	# Cores	CPU Util. (%)	Performance Index Per Core	CPU Util.$_{(Norm.)}$ (GHz)
Server01	Xeon E5640	2.67	2.0	4.0	25%	0.25	1.34
Server02	Xeon E5640	2.67	2.0	4.0	40%	0.25	2.14
Server03	Xeon E5647	2.93	2.0	4.0	30%	0.38	2.67
Server04	Xeon E5647	2.93	2.0	4.0	60%	0.38	5.34

Table 2: Normalized CPU utilization in GHz

Step 3: Table 3 shows the estimated CPU utilization in GHz after we include the buffer using Equation (3).

Host Name	CPU Model	Headroom (%)	CPU Util.$_{(Est.)}$ (GHz)
Server01	Xeon E5640	20%	1.60
Server02	Xeon E5640	20%	2.56
Server03	Xeon E5647	20%	3.21
Server04	Xeon E5647	20%	6.41

Table 3: Estimated CPU utilization in GHz

Step 4: After reviewing Amazon EC2 Instance Types, we decided to deploy M4 instances. Table 4 shows the performance index that is calculated using Equation (1) by taking the CPU model Xeon E5-2686 v4 as reference t = 0.

CPU Model	CPU Frequency (GHz)	# Cores	First Sold	Performance Index	Performance Index Per Core
Xeon E5-2686 v4	2.30	18.0	Jun-16	17.96	1.00

Table 4: Performance index for M4 class instances

Table 5 illustrates the CPU capacity of M4 instances after normalization.

Model	vCPU*	CPU Freq. (GHz)	Mem (GiB)	SSD Storage (GB)	Perf. Index Per Core	CPU Capacity$_{new}$ (GHz)
m4.large	2/2	2.3	8	EBS-only	1.00	2.30
m4.xlarge	4/2	2.3	16	EBS-only	1.00	4.60
m4.2xlarge	8/2	2.3	32	EBS-only	1.00	9.20
m4.4xlarge	16/2	2.3	64	EBS-only	1.00	18.40
m4.10xlarge	40/2	2.3	160	EBS-only	1.00	46.00
m4.16xlarge	64/2	2.3	256	EBS-only	1.00	73.60

Table 5: M4 class instances' CPU capacity after normalization

* The number of vCPUs is divided by 2 because each vCPU in an Amazon EC2 instance is a hyperthread of an Intel Xeon CPU core.

By comparing the results that you obtain from steps 3 and 4, Table 6 demonstrates the CPU selection mapping against each source machine that is being migrated to Amazon EC2.

Host Name	CPU Model	Recommended AWS Instance Type
Server01	Xeon E5640	m4.large
Server02	Xeon E5640	m4.xlarge
Server03	Xeon E5647	m4.xlarge
Server04	Xeon E5647	m4.2xlarge

Table 6: Recommended instance type

This example didn't take into account memory, storage, or I/O factors. For actual scenarios, we should consider taking a more holistic view to optimally balance performance and TCO saving. Amazon EC2 has many different classes of instance types, such as Compute Optimized, Memory Optimized, Storage Optimized, IO Optimized, and GPU Optimized – see https://aws.amazon.com/ec2/instance-

types for more detailed information. These different classes of instance types are optimized to deliver the best performance and TCO saving depending on your application's behavior and usage characteristics.

Best Practices

1. **Assess the requirements of your applications and select the appropriate Amazon EC2 instance family as a starting point for application performance testing.** Amazon EC2 provides you with a variety of instance types, each with one or more size options, organized into distinct instance families that are optimized for different types of applications. You should start evaluating the performance of your applications by:

    a) Identifying how your application compares to different instance families (for example, is the application compute-bound, memory-bound, or I/O bound?)

    b) Sizing your workload to identify the appropriate instance size. There is no substitute for measuring the performance of your entire application, because application performance can be impacted by the underlying infrastructure or by software and architectural limitations. We recommend application-level testing, including the use of application profiling and load testing tools and services.

2. **Normalize generations of CPUs by using Moore's Law.** Processing performance is usually bound to the number of CPU cores, clock speed, and type of CPU hardware instances that an application runs on. A new CPU model will usually outperform the models it precedes, even with the same number of cores and clock speed. Therefore, you should normalize different generations of CPUs by using Moore's Law, as shown earlier in Methodology, to obtain more realistic comparison results.

3. **Have a data-collection period that is long enough to capture the workload utilization pattern.** Workload changes in accordance with time shifting. For analysis, your data-collection period should be long enough to show you the peak and trough utilization across your business cycle (for example, monthly or quarterly). You should include peak utilization instead of average utilization for the purposes of CPU sizing. This will

ensure that you provide a consistent user experience when workloads are under peak utilization.

4. **Deploy discovery tools.** For large-scale environments (more than a few hundred machines), deploy automated discovery tools such as the AWS Application Discovery Service to perform data collection. It's critical to ensure that the discovery tools include basic inventory capabilities to collect the required CPU inventory and utilization (maximum, average, and minimum) that are specified in Methodology. Determine whether the discovery tool requires specific user permissions or secure/compliant ports to be opened. Also investigate whether the discovery tool requires the source machines to be rebooted to install agents. In many critical production environments, server rebooting is not permissible.

5. **Allocate enough buffer for spikes.** When you perform the CPU sizing and capacity planning, always include a reasonable buffer of 10–15% of total required capacity. This buffer is crucial to avoid any overlap of scheduled and unscheduled processing that may cause unexpected spikes.

6. **Monitor continuously.** Carry out the performance benchmarks before and after migration to investigate user experience acceptance levels. Deploy a cloud monitoring tool, such as Amazon CloudWatch, to monitor CPU performance. The cloud monitoring tool should use monitoring to send alerts if the CPU utilization exceeds the predefined threshold level. The tool also should provide reporting capability that generates relevant reports for short and long-term capacity planning purposes.

7. **Determine the right VM sizing.** A VM is considered undersized or stressed when the amount of CPU demand peaks above 70% for more than 1% of any 1 hour. A VM is considered oversized when the amount of CPU demand is below 30% for more than 1% of the entire range of 30 days. Figure 1 and Figure 2 give a good illustration of determining stress analysis for undersized and oversized conditions.

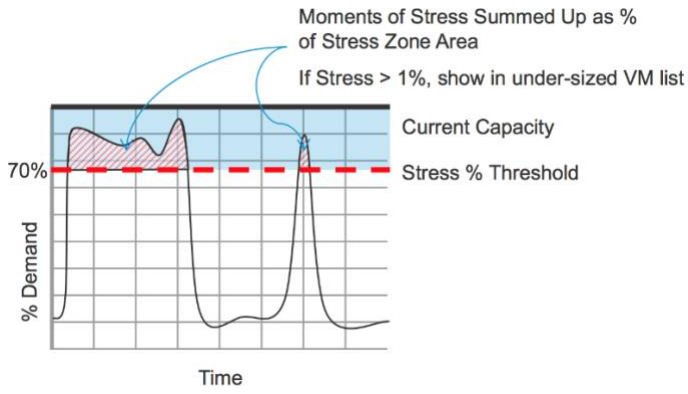

Figure 1: CPU Undersized condition

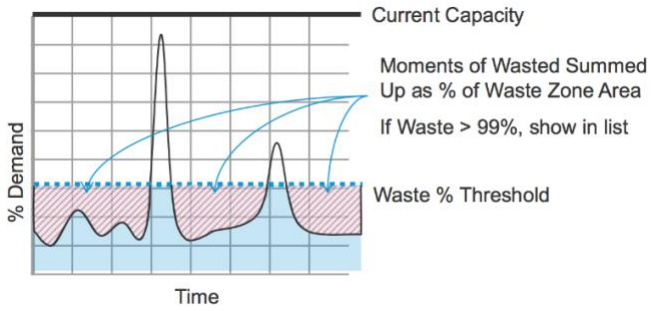

Figure 2: CPU Oversized condition

8. **Deploy single-threaded applications on uniprocessor virtual machines, instead of on SMP virtual machines, for the best performance and resource use.** Single-threaded applications can take advantage of a single CPU. Deploying such applications on dual-processor virtual machines does not speed up the application. Instead, it causes the second virtual CPU to unnecessarily hold physical resources that other VMs could otherwise use.

   The uniprocessor operating system versions are for single-core machines. If used on a multi-core machine, a uniprocessor operating system will recognize and use only one of the cores. The SMP versions, while required to fully utilize multi-core machines, can also be used on single-core machines. However, due to their extra synchronization code, SMP operating systems used on single-core machines run slightly slower than a uniprocessor operating system on the same machine.

9. **Consider using Amazon EC2 Dedicated Instances and Dedicated Hosts if you have compliance requirements.** Dedicated instances and hosts don't share hardware with other AWS accounts. To learn more about the differences between them, see aws.amazon.com/ec2/dedicated-hosts.

Conclusion

The methodology and best practices discussed in this paper give a pragmatic result for optimal performance regarding selected CPU resources. This methodology has been applied to many enterprises' cloud transformation projects and delivered more ***predictable performance*** with ***significant TCO saving***. Additionally, this methodology can be adopted for ***capacity planning*** and helps enterprises establish strong business justifications for platform expansion.

Actual performance sizing in a cloud environment should include memory, storage, I/O, and network traffic performance metrics to give a holistic performance sizing overview.

Contributors

The following individuals and organizations contributed to this document: Tan, Chin Khoon, Enterprise Migration Architect – APAC. For a more comprehensive and holistic example and discussion of cloud environment consolidation, please contact Tan Chin Khoon.

Document Revisions

Date	Description
September 2018	Updated formulas and instructions
August 2016	First publication