Lesson 4 of 12

Amazon EC2 Instance Families

Amazon EC2 offers over 500 instance types. You can choose the latest processor, storage, networking, and operating system to help you match the needs of your workload. Because there are so many options and customizations to choose from, new users such as John, can feel overwhelmed when deciding on an instance type and configuration for their application workloads.

John is learning that instances are categorized into instance families and then into sub-families. He has determined that each instance name references the underlying function and hardware resources available to that instance. There is a vast amount of information available and so many choices that John can make. He's feeling overwhelmed by all the information and is having trouble arranging it in his head. He reaches out to Sofía Martínez, his technical project lead and mentor, to get clarification and help understand exactly how Amazon EC2 is sorted into instance families and sub-families.

To hear what John has to say, choose each numbered marker in order. After you have selected the first marker, you can use the < > arrow keys to navigate through the conversation.

Instance types

An EC2 instance is a virtual machine (VM) that runs in the AWS Cloud. When you launch an instance, you decide the virtual hardware configuration by choosing an instance type. The instance type that you choose determines the hardware of the host computer used for your instance. Each instance type offers different compute, memory, and storage capabilities, and is grouped into an instance family based on these capabilities.

What to remember?

- An **instance** is a VM.
- An **instance type** is the combination of virtual hardware components, such as CPU and memory, that make up the instance.
- Instance types are grouped together into **instance families**. Each instance family is optimized for specific types of use cases.
- Instance families have **sub-families**, which are grouped according to the combination of processer and storage used.
- A virtual central processing unit (**vCPU**) is a measure of processing ability. For most instance types, a vCPU represents one thread of the underling physical CPU core. For example, if an instance type has two CPU cores and two threads per core, it will have four vCPUs.

The AWS instances are currently categorized into five distinct families. To learn more, expand each of the following five categories.

General purpose

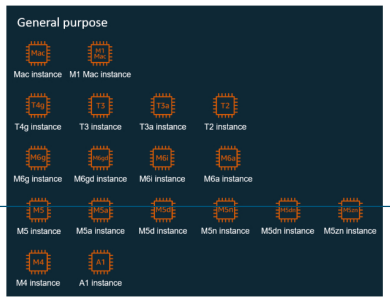
General purpose instances provide a balance of compute, memory, and networking resources and can be used for a wide range of workloads. These instances are ideal for applications that use these resources in equal proportions, such as web servers and code repositories.

Burstable instance options: Many workloads are not busy all the time and do not require sustained CPU performance. Using a large instance for these low-to-moderate workloads leads to waste and unnecessary cost.

For these workloads you can take advantage of the low-cost burstable general purpose instances, which are the T family instances. A **burst** is when the activity on the instance exceeds normal operation for a short period; for example, when the workload temporarily spikes. The T instance family provides a baseline CPU performance with the ability to burst above the baseline at any time for as long as required. The T instances offer a balance of compute, memory, and network resources. They provide you with the most cost-effective way to run a broad spectrum of general purpose applications that have a low-to-moderate CPU usage.

For additional information, see Burstable performance instances.

The following diagram shows the icons for the general purpose family and sub-families as of this publication.

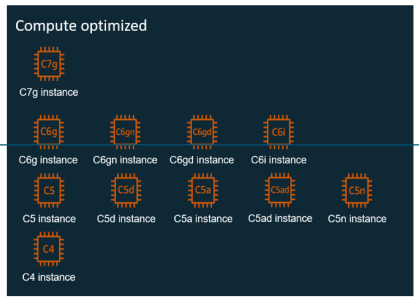


Compute optimized

Compute optimized instances are ideal for compute-bound applications that benefit from high-performance processors. Instances belonging to this family are well suited for compute-intensive operations, such as the following:

- Batch processing workloads
- Media transcoding
- High performance web servers
- High performance computing (HPC)
- Scientific modeling
- Dedicated gaming servers and ad server engines
- Machine learning (ML) inference

The following diagram shows the icons for the compute-optimized family and sub-families as of this publication.



Memory optimized

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

The following diagram shows the icons for the memory optimized family and sub-families as of this publication.

Storage optimized –

Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random input/output (I/O) operations per second (IOPS) to applications.

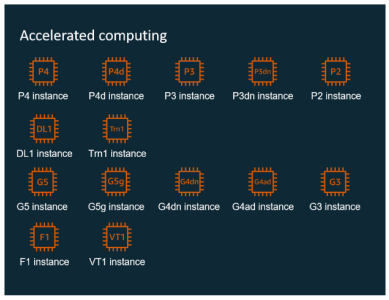
The following diagram shows the icons for the storage optimized family and sub-families as of this publication.

Accelerated computing –

Accelerated computing instances use hardware accelerators, or co-processors, to perform some functions more efficiently than is possible in software running on CPUs. Examples of such functions include floating point number calculations, graphics processing, and data pattern matching. Accelerated computing instances facilitate more parallelism for higher throughput on compute-intensive workloads.

If you require high processing capability, you will benefit from using accelerated computing instances, which provide access to hardware-based compute accelerators such as graphics processing units (GPUs), field programmable gate arrays (FPGAs), or AWS Inferentia.

The following diagram shows the icons for the accelerated computing family and sub families as of this publication.



John has questions

After his meeting with Sofía in the conference room, John spent time reading through the instance families to learn about their specific family-based optimizations. He's grasped the distinction between the families and sub-families, but the instance names are very confusing to him. He goes to Sofía's office to get clarification.

To hear what John has to say, choose each numbered marker in order. After you have selected the first marker, you can use the < > arrow keys to navigate through the conversation.

Decoding instance names

Instance names are a combination of the instance family, generation, and size. They can also indicate additional capabilities, such as specific processor type or optimized networking performance.

To learn about what each letter in an instance family means, choose each numbered marker in order. After you have selected the first marker, you can use the < > arrow keys to navigate through the remaining markers.

Instance sizing

EC2 instances are sized based on the combined hardware resources consumed by that instance type. This means the size is the total configured capacity of vCPU, memory, storage, and networking. The sizes range from nano to upwards of 32xlarge, with a nano-sized instance using the least amount of hardware resources and the 32xlarge instance using the most amount of hardware resources (128 vCPU and 1,024 GiB memory). Let's take a quick look at a size comparison chart to help you understand how the allocated hardware corresponds to the instance size. In the case of the general purpose T instance family, the vCPU allocation remains the same but the memory doubles with each, larger size.

Instance Family	Instance Size	vCPU	Memory (GiB)
General purpose	t4g.nano	2	.5
General purpose	t4g.micro	2	1
General purpose	t4g.small	2	2
General purpose	t4g.medium	2	4
General purpose	t4g.large	2	8

A gibibyte (GiB) is a unit of measure that represents size capacity. Sometimes the measurement gigabyte (GB) and GiB are used interchangeably but this is not accurate. One GiB is defined as 1024^3 bytes, whereas one GB is defined as 1000^3 bytes.

Why does it matter if you use GiB or GB? For small amounts of capacity the numbers of GiB vs GB vary only slightly, but as you scale to the massive capacity of a cloud solution, the size variance between the numbers grows exponentially. For additional information, query GB vs GiB in your favorite browser and read more about how the numbers differ.

Instance family growth

Different instance families grow based on the resource for which the family is optimized. The following table shows the compute optimized growth and how it focuses on vCPU resources and the memory optimized growth and how it focuses on the memory resources.

Instance Family	Instance Size	vCPU	Memory (GiB)
Compute optimized	c5.xlarge	4	8
Compute optimized	c5.2xlarge	8	16
Compute optimized	c5.4xlarge	16	32
Memory optimized	r5g.xlarge	4	32
Memory optimized	r6g.2.xlarge	6	64
Memory optimized	r6g.4.xlarge	16	128

Additional characteristics

Instance names can include additional capabilities. This is a list of additional items that you might see in an instance name and what they mean.

- a – AMD processors
- g – AWS Graviton processors
- i – Intel processors
- d – Instance store volumes
- n – Network optimization
- b – Block storage optimization
- e – Extra storage or memory
- z – High frequency

EC2 instance types
For information on each instance family and the individual component breakdown, such as CPU, memory, and storage, choose the Instance Specifics button.

INSTANCE SPECIFICS

Knowledge check

EC2 instance name challenge:

Keyboard navigation for knowledge checks +

Question 1: In the instance name **c6gn.xlarge**: What does the letter 'g' mean?

- ○ Instance family
- ○ Generation
- ○ AWS Graviton processor
- ○ Network Optimized

SUBMIT

Question 2: In the instance name **c6gn.xlarge**: What does the letter 'c' indicate?

- ○ Instance family
- ○ Generation
- ○ AWS Graviton processor
- ○ Network Optimized

SUBMIT

Question 3: In the instance name **c6gn.xlarge**: What does the number 6 represent?

- ○ Instance family

- Generation
- AWS Graviton processor
- Network Optimized

SUBMIT

Choosing the right workload for the job is important. However, if you choose an instance that isn't right for the load or for the cost associated with it, you can try other instance types to find a better fit. Moving from one instance to another is often done to improve the performance and lower the cost. John doesn't know how straightforward it is to change instance types, so he's focused on finding the perfect instance type. Let's check in on John and Sofía.