Leveraging Amazon EC2 Spot Instances at Scale

AWS Whitepaper

Leveraging Amazon EC2 Spot Instances at Scale: AWS Whitepaper

Table of Contents

Leveraging Amazon EC2 Spot Instances at Scale 1
Abstract 1
Introduction 1
When to Use Spot Instances 2
How to Request Spot Instances 3
How Spot Instances Work 4
Managing Instance Termination 5
Launch Groups 6
Spot Fleets 7
Spot Request Limits 8
Determining the Status of Your Spot Instances 9
Spot Instance Interruptions 10
Spot Instance Best Practices 11
Spot Integration with Other AWS Services 12
Amazon EMR Integration 12
AWS CloudFormation Integration 12
Auto Scaling Integration 12
Amazon ECS Integration 12
Amazon Batch Integration 12
Conclusion 13
Resources 14
Document Details 15
Contributors 15
..... 15
AWS Glossary 16

Leveraging Amazon EC2 Spot Instances at Scale

Publication date: **March 2018** (*Document Details* (p. 15))

Abstract

This is the fourth in a series of whitepapers designed to support your cloud journey. This paper seeks to empower you to maximize value from your investments, improve forecasting accuracy and cost predictability, create a culture of ownership and cost transparency, and continuously measure your optimization status.

This paper provides an overview of Amazon EC2 Spot Instances, as well as best practices for using them effectively.

Introduction

In addition to On-Demand and Reserved Instances, the third major Amazon Elastic Compute Cloud (Amazon EC2) pricing model is Spot Instances. With Spot Instances, you can use spare Amazon EC2 computing capacity at discounts of up to 90% compared to On-Demand pricing. That means you can significantly reduce the cost of running your applications, or grow your application's compute capacity and throughput for the same budget. The only difference between On-Demand Instances and Spot Instances is that Spot Instances can be interrupted by EC2 with two minutes of notification when EC2 needs the capacity back.

Unlike Reserved Instances, Spot Instances do not require an upfront commitment. However, because Spot Instances can be terminated if the Spot price exceeds your maximum price or if no capacity is available for the instance type you've specified, they are best for flexible workloads.

When to Use Spot Instances

You can use Spot Instances for various fault-tolerant and flexible applications. Examples include web servers, API backends, continuous integration/continuous development, and Hadoop data processing.

Workloads that constantly save data to persistent storage—including Amazon Simple Storage Service (Amazon S3), Amazon Elastic Block Store (Amazon EBS), Amazon Elastic File System (Amazon EFS), Amazon DynamoDB, or Amazon Relational Database Service (Amazon RDS)—can work effectively with Spot Instances.

You can also take advantage of Spot Instances to run and scale applications such as stateless web services, image rendering, big data analytics, and massively parallel computations. Spot Instances are typically used to supplement On-Demand Instances, where appropriate, and are not meant to handle 100% of your workload. However, you can use all Spot Instances for any stateless, non-production application, such as development and test servers, where occasional downtime is acceptable. They are not a good choice for sensitive workloads or databases.

How to Request Spot Instances

To use Spot Instances, you create a Spot Instance request that includes the number of instances, the instance type, the Availability Zone, and the maximum price that you are willing to pay per instance hour. You can create a Spot Instance request using the Launch Instance Wizard from the Amazon EC2 console or Amazon EC2 API.

For details on how to create a Spot Instance request using the console, see Creating a Spot Instance Request. For details on how to request Spot Instances through the Amazon EC2 API, see RequestSpotInstances in the Amazon EC2 API Reference.

You can also launch Spot Instances through other AWS services such as Amazon EMR, AWS Data Pipeline, AWS CloudFormation, and Amazon Elastic Container Service (Amazon ECS), as well as through third-party tools.

To learn more about Spot Instance requests, see Spot Instance Requests.

How Spot Instances Work

The Spot price is determined by long-term trends in supply and demand for EC2 spare capacity. You pay the Spot price that's in effect at the beginning of each instance-hour for your running instance, billed to the nearest second.

With Spot Instances, you never pay more than the maximum price you specify. If the Spot price exceeds your maximum price for a given instance or if capacity is no longer available, your instance will automatically be terminated (or be stopped/hibernated, if you opt for this behavior on persistent request).

The Spot price may change anytime, but in general, it will change once per hour and in many cases less frequently. AWS publishes the current Spot price and historical prices for Spot Instances through the describe-spot-price-history command as well as the AWS Management Console. This can help you assess the levels and timing of fluctuations in the Spot price over time.

Spot Instances perform exactly like other EC2 instances while running and can be terminated when you no longer need them. If you terminate your instance, you pay for any partial hour used (as you do for On-Demand or Reserved Instances). However, you are not charged for any partial hour of usage if the Spot price goes above your maximum price and Amazon EC2 interrupts your Spot Instance.

Managing Instance Termination

Spot offers three features to help you better track and control when Spot Instances run and terminate (or stop/hibernate).

- **Termination notices** – If you need to save state, upload final log files, or remove Spot Instances from Elastic Load Balancing before interruption, you can use termination notices, which are issued two minutes prior to interruption. To learn more about managing interruptions, see Spot Instance Interruptions.
- **Persistent requests** – You can opt to set your request to remain open so that a new instance will be launched in its place when the instance is interrupted. You can also have your Amazon EBS-backed instance stopped upon interruption and restarted when Spot has capacity at your preferred price. To learn more about persistent and one-time requests, see Spot Instance Request States.
- **Block durations** – If you need to execute workloads continuously for 1–6 hours, you can also specify a duration requirement when requesting Spot Instances. To learn more about block durations for Spot Instances, see Specifying a Duration for Your Spot Instances.

Launch Groups

You can launch a set of Spot Instances at once in a launch group or in an Availability Zone group. With a launch group, if the Spot service must terminate one of the instances in a launch group, it must terminate them all. With an Availability Zone group, the Spot service launches a set of Spot Instances in the same Availability Zone.

When launch groups are required, try to minimize the group size. Larger groups have a lower chance of being fulfilled. Also, be aware that specifying a specific Availability Zone can increase your chances of successfully launching. To learn more about launch groups and Availability Zone groups, see How Spot Instances Work.

Spot Fleets

With a Spot Fleet, you can automatically request Spot Instances with the lowest price per unit of capacity. To use a Spot Fleet, create a Spot Fleet request that includes the target capacity based on your application needs (in any unit, including instances, vCPUs, memory, storage, or network throughput), one or more launch specifications for the instances, and the maximum price that you are willing to pay. To learn more about Spot Fleets, see How Spot Fleet Works.

Spot Request Limits

By default, there is an account limit of 20 Spot Instances per AWS Region. If you terminate your Spot Instance but do not cancel the request, the request counts against this limit until Amazon EC2 detects the termination and closes the request.

Spot Instance limits are dynamic. When your account is new, your limit might be lower than 20 to start, but then increase over time. In addition, your account might have limits on specific Spot Instance types. If you submit a Spot Instance request and you receive the error **Max spot instance count exceeded**, you can go to the AWS Support Center and request a limit increase. To learn more about default limits and how to request a limit increase, see AWS Service Limits.

Determining the Status of Your Spot Instances

By reviewing Spot status, you can see why your Spot requests state has or has not changed, and you can learn how to optimize your Spot requests to get them fulfilled. To find the Spot status, you can use the DescribeSpotInstanceRequests API action or the describe-spot-instance-requests using the AWS Command Line Interface (CLI).

The AWS Management Console makes a detailed billing report available, which shows Spot Instance start and termination times for all instances. You can check the billing report against historical Spot prices via the API to verify that the Spot price billed was correct.

Spot Instance Interruptions

You can choose to have the Spot service stop instead of terminate your Amazon EBS-backed Spot Instances when they are interrupted. Spot can then fulfill your request by restarting instances from a stopped state when capacity again becomes available within your price and time requirements.

To use this new feature, choose **stop** instead of **terminate** as the interruption behavior when submitting a persistent Spot request. When you choose **stop**, Spot will shut down your instance upon interruption. The EBS root device and attached EBS volumes are saved and their data persists. When capacity is available again within your price and time requirements, Spot will restart your instance. Upon restart, the EBS root device is restored from its prior state, previously attached data volumes are reattached, and the instance retains its instance ID.

This feature is available for persistent Spot requests and Spot Fleets with the maintain fleet option enabled. You will not be charged for instance usage while your instance is stopped. EBS volume storage is charged at standard rates.

Spot Instance Best Practices

Your instance type requirements, budget requirements, and application design will determine how to apply the following best practices for your application:

- **Be flexible about instance types.** Test your application on different instance types when possible. Because prices fluctuate independently for each instance type in an Availability Zone, you can often get more compute capacity for the same price when you have instance type flexibility. Request all instance types that meet your requirements to further reduce costs and improve application performance. Spot Fleets enable you to request multiple instance types simultaneously.
- **Choose pools where prices haven't changed much.** Because prices adjust based on long-term demand, popular instance types (such as recently launched instance families), tend to have more price adjustments. Therefore, picking older-generation instance types that are less popular tends to result in lower costs and fewer interruptions. Similarly, the same instance type can have different prices in different Availability Zones.
- **Minimize the impact of interruptions.** Because the Spot service can terminate Spot Instances without warning, it is important to build your applications in a way that allows you to make progress even if your application is interrupted. There are many ways to accomplish this, two of which are:

- **Add checkpoints.** Add checkpoints that save your work periodically, for example, to an Amazon EBS volume. Another approach is to launch your instances from Amazon EBS–backed Amazon Machine Images.
- **Split up the work.** By using Amazon Simple Queue Service (Amazon SQS), you can queue up work increments and keep track of work that has already been done.

Spot Integration with Other AWS Services

Amazon EC2 Spot Instances integrate with several AWS services.

Amazon EMR Integration

You can run Amazon EMR clusters on Spot Instances and significantly reduce the cost of processing vast amounts of data on managed Hadoop clusters. You can run your EMR clusters by easily mixing Spot Instances with On-Demand and Reserved Instances using the instance fleet feature. To learn more about setting up an EMR cluster with Spot, see the EMR Developer Guide.

AWS CloudFormation Integration

AWS CloudFormation makes it easy to organize and deploy a collection of AWS resources, including EC2 Spot, and lets you describe any dependencies or special parameters to pass in at runtime. For a sample high-performance computing framework using AWS CloudFormation that can use Spot Instances, see the cfncluster demo. To learn more about setting up AWS CloudFormation with Spot, see the Amazon EC2 User Guide.

Auto Scaling Integration

You can use Auto Scaling groups to launch and manage Spot Instances, maintain application availability, and scale your Amazon EC2 Spot capacity up or down automatically according to the conditions and maximum prices you define. To learn more about using Auto Scaling with Spot Instances, see the Auto Scaling Developer Guide.

Amazon ECS Integration

You can run Amazon ECS clusters on Spot Instances to reduce the operational cost of running containerized applications on Amazon ECS. The Amazon ECS console is also tightly integrated with Amazon EC2 Spot, and you can use the Create Cluster Wizard to easily set up an ECS cluster with Spot Instances.

Amazon Batch Integration

AWS Batch plans, schedules, and executes your batch computing workloads on AWS. AWS Batch dynamically requests for Spot Instances on your behalf, reducing the cost of running your batch jobs.

Conclusion

Whether you have flexible compute needs or want to augment capacity without growing your budget, Spot Instances can be a great way to optimize your AWS costs. By properly architecting your workloads, you can take advantage of Spot pricing for a wide range of needs. For more information about Spot Instances, visit the Spot Instances overview.

Resources

- AWS Architecture Center
- AWS Whitepapers
- AWS Architecture Monthly
- AWS Architecture Blog
- This Is My Architecture videos
- AWS Answers
- AWS Documentation

Document Details

Contributors

The following individuals and organizations contributed to this document:

- Amilcar Alfaro, Sr. Product Marketing Manager, AWS
- Erin Carlson, Marketing Manager, AWS
- Keith Jarrett, WW BD Lead - Cost Optimization, AWS Business Development

Date	Description
March 2018	First publication

AWS Glossary

For the latest AWS terminology, see the AWS Glossary in the *AWS General Reference*.