



# Builders Online Series

## 9 ways to optimize your costs in the cloud

Gabe Hollombe  
Senior Developer Advocate,  
Amazon Web Services



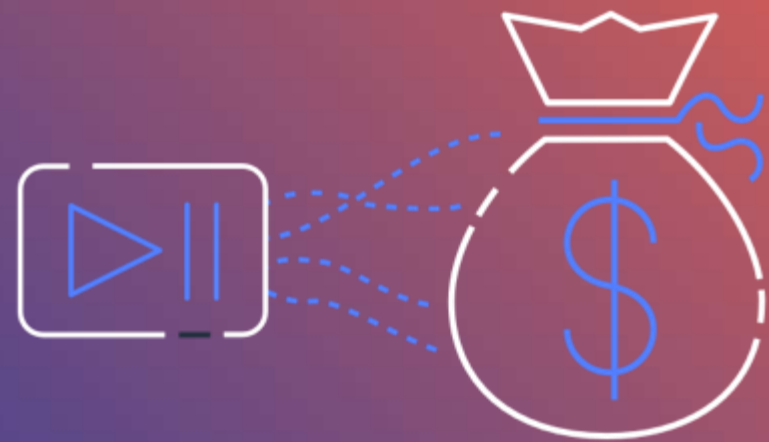
@gabehollombe

Today's focus

Tools and approaches  
that you can use to  
optimize AWS costs



# Before we start...



# 9

## the nine ways

1. Stop paying for idle Amazon EC2 and Amazon RDS instances
2. Use Arm-based Amazon EC2 instances
3. Choose Amazon EC2 Spot Instances
4. Stop paying for under-utilized Amazon EC2 instances
5. Use an Amazon S3 lifecycle policy to transition objects to S3 IA or S3 Glacier
6. Use the Amazon S3 Intelligent Tiering storage class
7. Use an Amazon S3 Gateway endpoint from your VPC private subnets
8. Use on-demand capacity mode for Amazon DynamoDB
9. Use Compute Savings Plans



Implementation time: *Minutes to hours*

Stop paying for idle Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Relational Database Service (Amazon RDS) instances

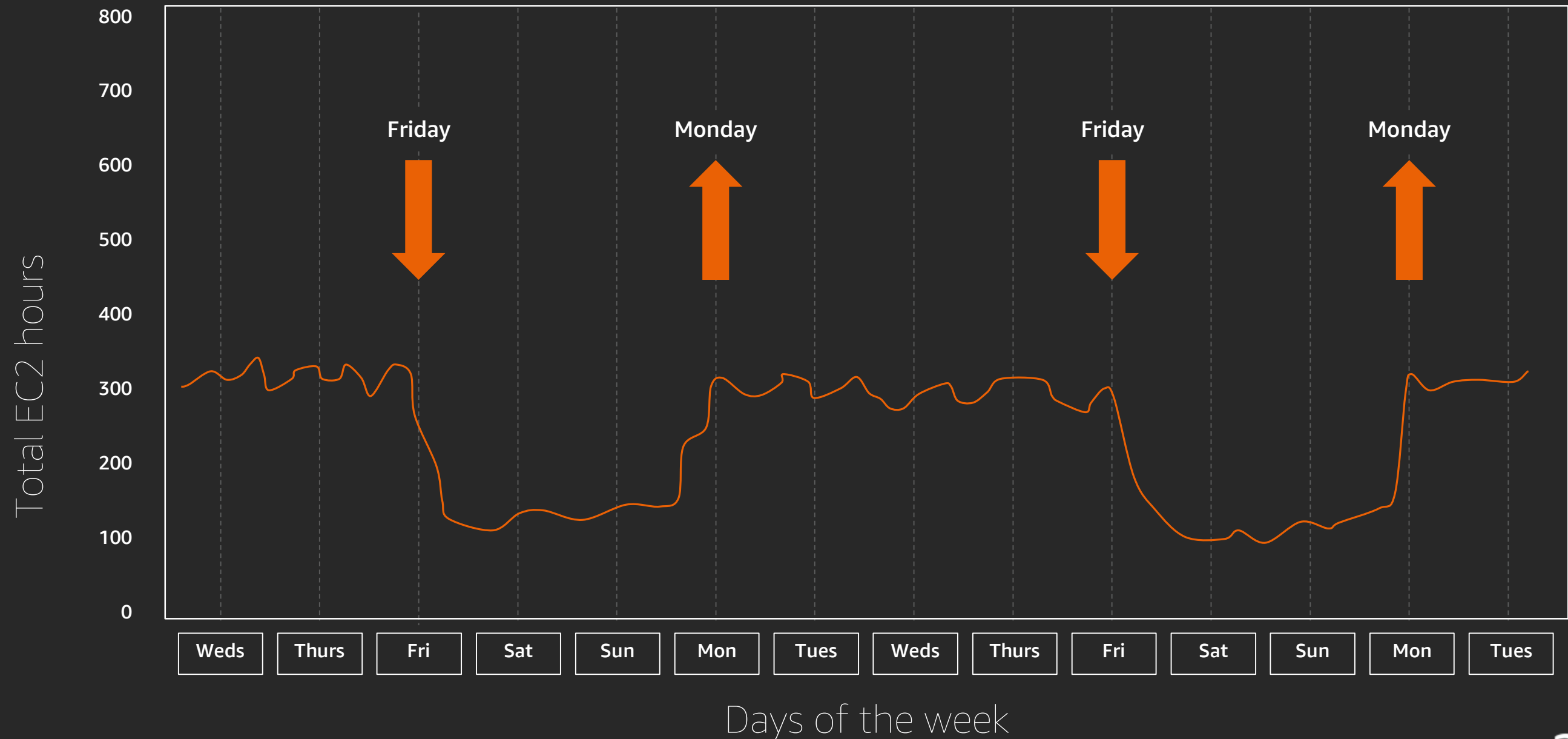
# Scenario

- You leave instances running during evenings, weekends, and holidays
- You might be paying for Amazon EC2 and Amazon RDS instances even when they are idle



Solution: AWS Instance Scheduler

# Pay for what you need



# Schedule Amazon EC2 and Amazon RDS instances in non-production environments

Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes to hours	Reduce on-demand costs by up to 35%*	Minutes	None

*\* For an example instance schedule that stops them on Friday at 6pm and starts them again on Monday at 6am*



# Get started

1

Install the AWS Instance Scheduler

2

Create schedule based on business requirements

3

Tag non-production Amazon EC2 and Amazon RDS instances to be scheduled

<https://aws.amazon.com/solutions/instance-scheduler/>

## AWS Instance Scheduler

Version 1.3.1

Last updated: 03/2020

Author: AWS

Estimated deployment time: 5 min

[Source code](#)

[CloudFormation template](#)

[View deployment guide](#)

[Launch solution in the AWS Console](#)

[Deploy with an AWS IQ expert](#)

## Deployment resources

[Download deployment guide](#)

[AWS Solution resources »](#)

[Contact us »](#)





Implementation time: *Minutes to hours*

# Use Arm-based Amazon EC2 instances

# Scenario

- You are running typical open-source application stacks deployed on Intel x86-64 architectures
- You might be able to run the same software on instances using AWS Graviton2 Arm-based chip



Solution: Migrate to AWS Graviton2-based instances like the Amazon EC2 M6g, C6g, and R6g

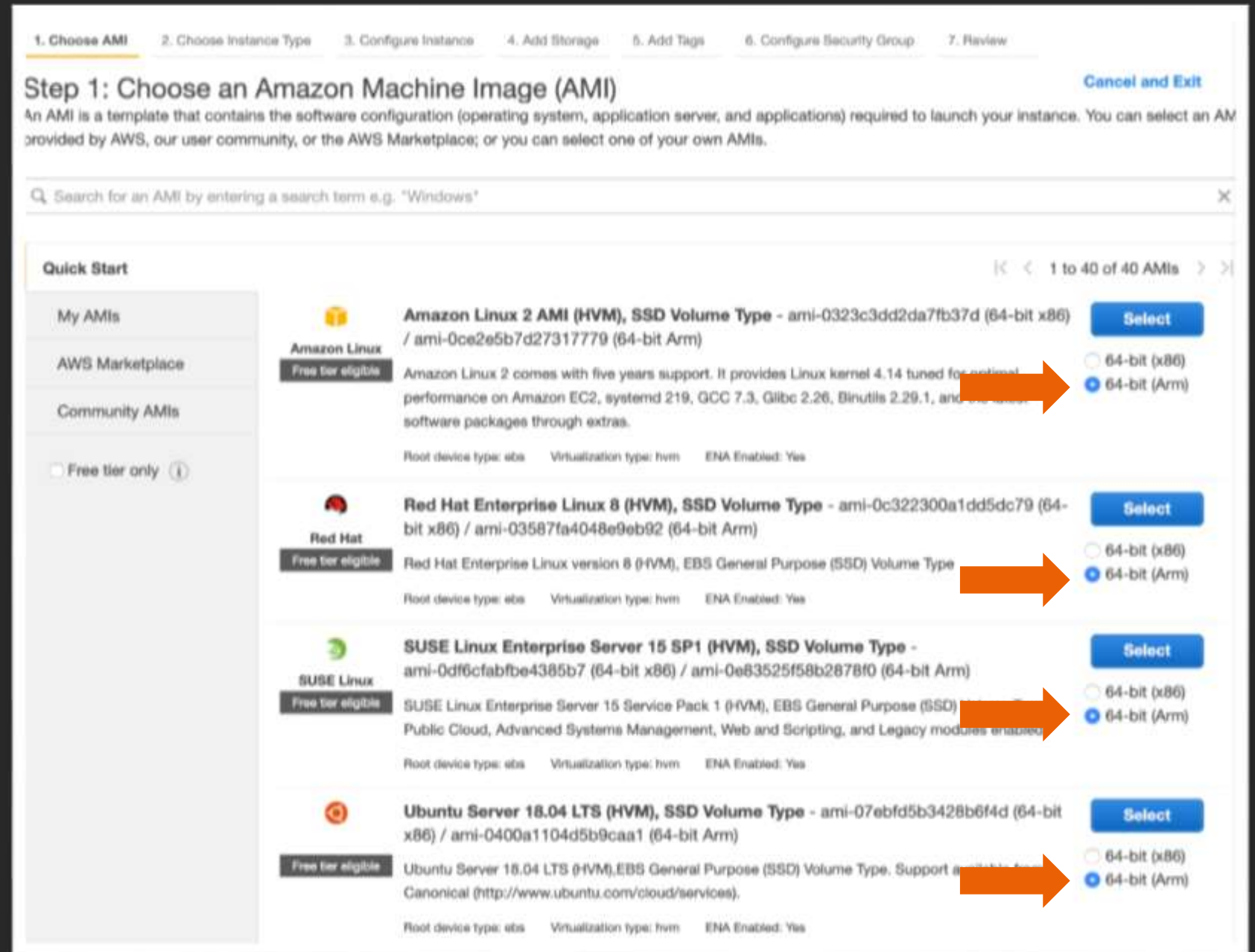
# Resize, pause, and resume Amazon Redshift clusters

Implementation time	Savings potential	Time to realize savings	Commitment required
Hours to days	Up to 40% better price performance over comparable x86-based instances for many workloads	Hours	None

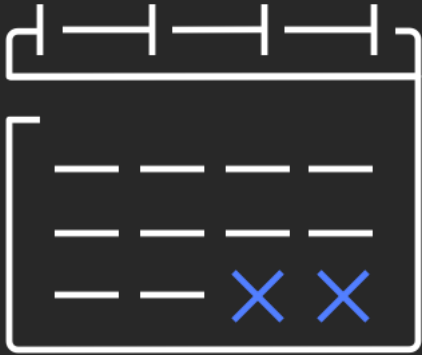
# Get started

Select the Amazon Machine Image (AMI) corresponding to the Arm version of your favorite distribution when launching an instance in the AWS Management Console.

Be sure to select the 64-bit (Arm) button on the right part of the screen.



For info on building for AWS Graviton2 see <https://github.com/aws/aws-graviton-getting-started>



Implementation time: *Hours/days to weeks*

# Choose Amazon EC2 Spot instances

# Scenario

- You are running big data, containerized workloads, CI/CD, web servers, high-performance computing, or other test & development workloads that are fault-tolerant
- You are paying the default on-demand pricing



Solution: Amazon EC2 Spot

# Choose Amazon EC2 Spot for workloads that are stateless, fault-tolerant, and loosely-coupled

Implementation time	Savings potential	Time to realize savings	Commitment required
Hours/days to weeks	Up to 90% less than on-demand pricing	Hours	None



# Getting started

## Self-service Amazon EC2 Spot references



Amazon EC2 Spot workshops

<https://ec2spotworkshops.com/>



Cost Optimize Big Data Workloads

<https://aws.amazon.com/ec2/spot/use-case/emr/>



Amazon ECS, Amazon EKS, and AWS Fargate Spot

<https://aws.amazon.com/ec2/spot/containers-for-less/get-started/>

<https://aws.amazon.com/blogs/compute/run-your-kubernetes-workloads-on-amazon-ec2-spot-instances-with-amazon-eks/>

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/fargate-capacity-providers.html>



Implementation time: *Minutes/hours to days*

# Stop paying for underutilized Amazon EC2 instances

# Scenario

- You are running Amazon EC2 workloads that are compute-intensive but CPU usage stays under 40%
- You may be paying for overprovisioned instances



Solution: Amazon EC2 Resource Optimization Recommendations in AWS Cost Explorer

# Right size or terminate underutilized Amazon EC2 instances identified by AWS Cost Explorer

Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes/hours to days	\$100s to \$1000s	Hours	None

# Get started

AWS Cost Explorer will provide you with Amazon EC2 rightsizing recommendations

1

Enable rightsizing recommendations in the AWS Cost Explorer

2

Review individual rightsizing recommendations

3

Perform the Amazon EC2 instance size modifications

<https://aws.amazon.com/blogs/aws-cost-management/launch-resource-optimization-recommendations/>

The screenshot displays the AWS Cost Explorer interface. At the top right, there is a 'Settings' button with a gear icon. Below it, the 'Recommendations' section is visible, featuring a checkbox labeled 'Receive Amazon EC2 resource recommendations' which is checked. A description below the checkbox states: 'Enable Amazon EC2 Rightsizing Recommendations which examine your historical usage pattern recommendations.'

An orange arrow points from the text 'AWS Cost Explorer will provide you with Amazon EC2 rightsizing recommendations' to this 'Recommendations' section.

Below the 'Recommendations' section, there is a detailed view titled 'Underutilized Amazon EC2 instance details'. This view includes a table with columns: 'Account name', 'Account ID', 'Instance ID', 'Instance type', 'Region', and 'Tag(s)'. The instance type is 'm5.xlarge' and the region is 'US East (Ohio)'. Below the table, there is a 'Utilization' section showing metrics: CPU (1%), Disk (-), Memory (2%), Network capacity (Up to 10 Gigabit), RI hours (0), On-Demand hours (333), and Total running hours (333). The 'Recommended action' section states: 'We recommend that you modify your underutilized m5.xlarge instance to a m5.large instance in order to save an estimated \$828 annually. You can improve these recommendations by installing the CloudWatch Agent. [Learn more](#)'. Below this, the 'AWS recommendation' section shows the recommended instance type 'm5.large' with a projected utilization of 'CPU: 2% | Disk: - | Memory: 3%' and an estimated savings of '\$69 per month'. At the bottom right, there are 'Cancel' and 'Go to the Amazon EC2 console' buttons.

An orange arrow points from the text 'Perform the Amazon EC2 instance size modifications' to the 'Recommended action' section of the 'Underutilized Amazon EC2 instance details' view.





Implementation time: *Minutes/hours to days*

Use an Amazon Simple Storage Service (Amazon S3) lifecycle configuration to transition objects to cost-optimized storage classes

# Scenario

- You keep all of your data in Amazon S3 buckets using the default Standard storage class
- You are paying more than you need to for data that you don't frequently access



Solution: Amazon S3 Lifecycle configuration rules

# Use Amazon S3's Infrequent Access (IA) storage class to save on costs for appropriate data

Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes	Close to 50% less than the <i>Standard</i> storage class	30 days	30 days minimum storage period



# Get started

## Add a lifecycle rule to your Amazon S3 bucket

1

Select a bucket and optionally provide a prefix or tag to select objects for transitioning

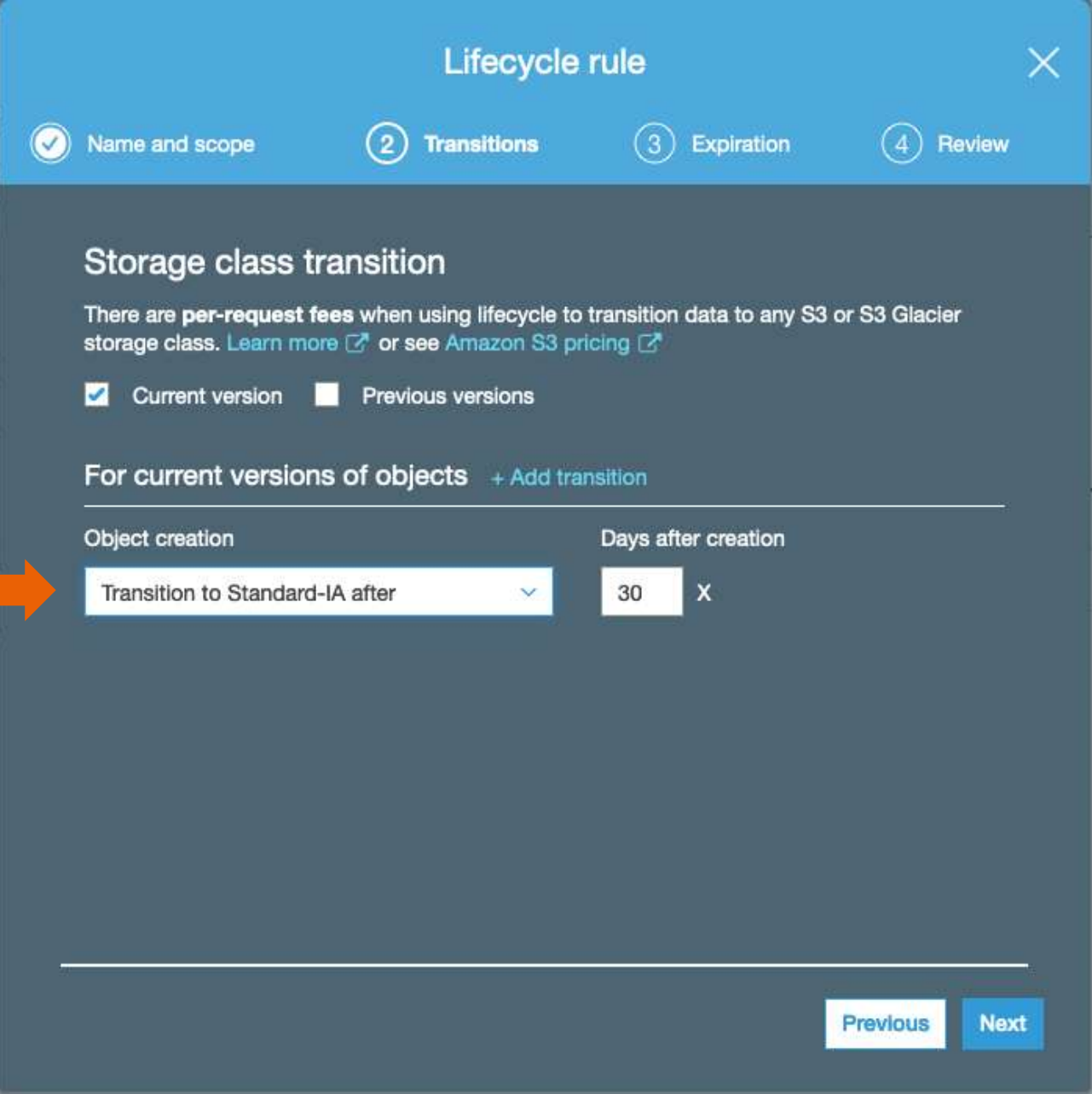
2

Select a storage class to transition to and specify the number of days to wait before transitioning

3

Review and save your configuration

<https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html>



The screenshot shows the 'Lifecycle rule' configuration window in the AWS console, specifically the 'Transitions' step. The window has a blue header with a close button (X) and four steps: 1. Name and scope, 2. Transitions (active), 3. Expiration, and 4. Review. The main content area is titled 'Storage class transition' and includes a warning about per-request fees. Below this, there are checkboxes for 'Current version' (checked) and 'Previous versions'. A section titled 'For current versions of objects' has a '+ Add transition' link. Below this, there are two columns: 'Object creation' and 'Days after creation'. In the 'Object creation' column, a dropdown menu is open, showing 'Transition to Standard-IA after'. An orange arrow points to this dropdown. In the 'Days after creation' column, there is a text input field containing '30' and a checkbox labeled 'X'.

Lifecycle rule

1 Name and scope 2 Transitions 3 Expiration 4 Review

**Storage class transition**

There are **per-request fees** when using lifecycle to transition data to any S3 or S3 Glacier storage class. [Learn more](#) or see [Amazon S3 pricing](#)

☒ Current version ☐ Previous versions

For current versions of objects [+ Add transition](#)

Object creation	Days after creation
Transition to Standard-IA after	30 X

[Previous](#) [Next](#)





Implementation time: *Minutes*

# Enable Amazon S3 Intelligent-Tiering

# Scenario

- You are using Amazon S3 the *Standard* storage class and you have data with changing access patterns
- You might benefit from monitoring and transitioning data to a lower cost/frequency storage class



Solution: Amazon S3 Intelligent-Tiering

# Enable Amazon S3 Intelligent-Tiering for objects with changing access patterns

Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes	20% – 30% (for S3 Standard objects transitioned to S3 INT)	30 days	30 days minimum storage period

# Get started



Upload objects directly into  
S3 Intelligent-Tier (API)

The screenshot shows the 'Lifecycle rule' configuration window in the AWS console. The window has a blue header with the title 'Lifecycle rule' and a close button. Below the header is a progress bar with four steps: 1. Name and scope (checked), 2. Transitions (active), 3. Expiration, and 4. Review. The main content area is titled 'Storage class transition' and includes a description: 'You can add rules in a lifecycle configuration to tell Amazon S3 to transition objects to another storage class. [Learn more](#)'. There are two checkboxes: 'Current version' (checked) and 'Previous versions' (unchecked). Below this is a section titled 'For current versions of objects' with a link to 'Add transition'. The configuration table shows one transition rule: 'Transition to Intelligent-Tiering af...' with a dropdown arrow, under the 'Object creation' column, and '30' days under the 'Days after creation' column, with an 'X' icon to delete the rule.

Create Lifecycle rules that make use  
of Intelligent-Tiering (UI)

<https://aws.amazon.com/blogs/aws/new-automatic-cost-optimization-for-amazon-s3-via-intelligent-tiering/>



Implementation time: *Minutes*

Use an Amazon S3 Gateway endpoint from your private subnet instances inside a VPC

# Scenario

- You have Amazon EC2 instances in a private subnet that need to communicate with an Amazon S3 bucket in the same region
- You can terminate your NAT Gateway and communicate with Amazon S3 directly from your private subnets



Solution: Amazon S3 Gateway endpoint

# Use an Amazon S3 Gateway endpoint for same-region traffic

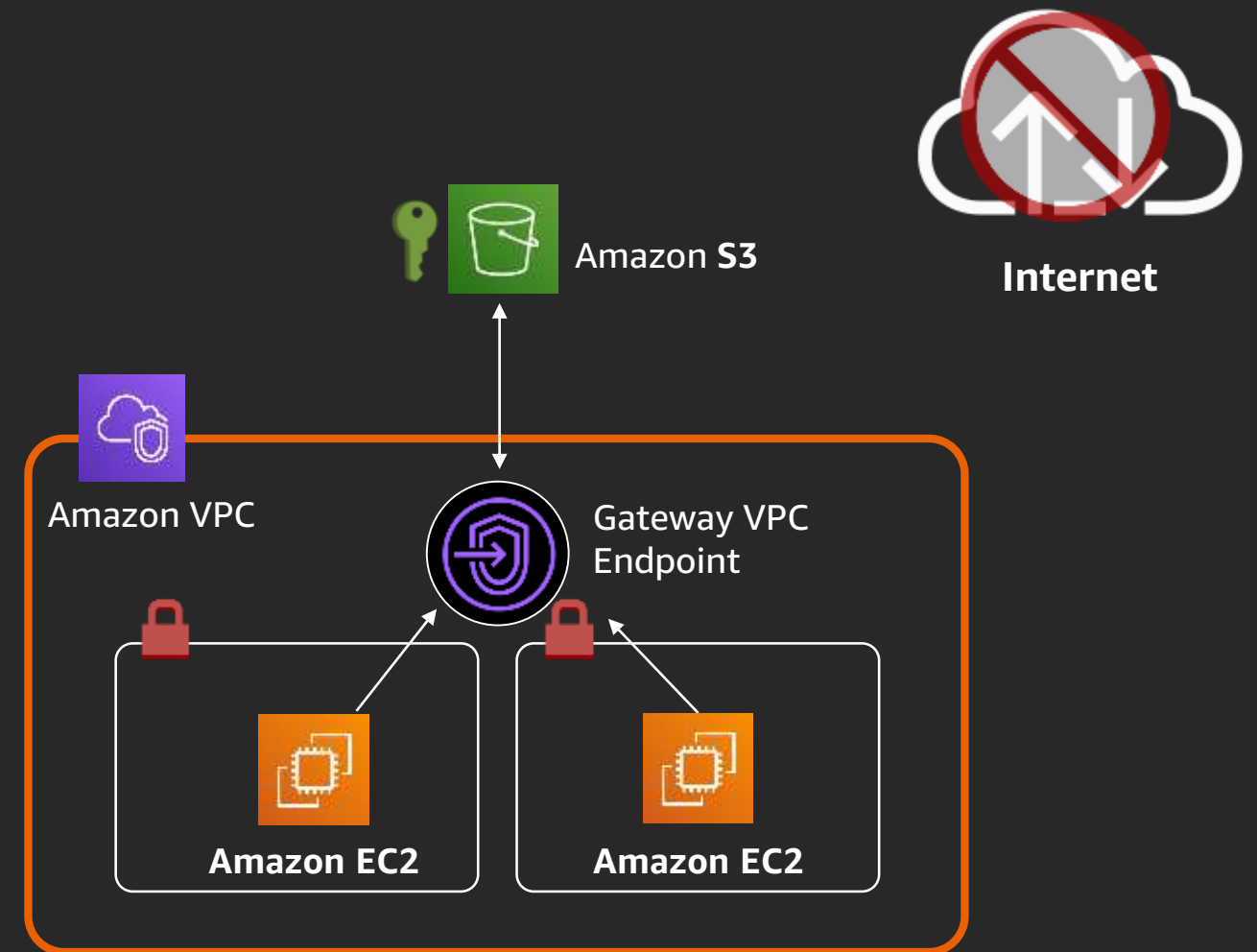
Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes	About \$30 per month per terminated NAT gateway & About \$45 per TB of network data	Hours	None



# Get started

Access Amazon S3 using an Amazon S3 Gateway endpoint without using NAT or Internet gateways

You can also restrict access to your Amazon S3 bucket from traffic originating from outside of your VPC



<https://docs.aws.amazon.com/vpc/latest/userguide/vpce-gateway.html>



Implementation time: *Minutes*

# Use on-demand capacity mode for Amazon DynamoDB

# Scenario

- You are using Amazon DynamoDB Tables configured using provisioned capacity
- You might be paying for provisioned capacity that you aren't using



Solution: Amazon DynamoDB On-Demand

# Configure on-demand capacity mode for Amazon DynamoDB tables that are idle, or for unpredictable workloads

Implementation time	Savings potential	Time to realize savings	Commitment required
Minutes	Requires estimating and/or evaluating	Minutes	24 hours

# Get started



**Read/write capacity mode**

Select on-demand if you want to pay only for the read and writes you perform, with no capacity planning required. Select provisioned to save on throughput costs if you can reliably estimate your application's throughput requirements. See the [DynamoDB pricing page](#) and [DynamoDB Developer Guide](#) to learn more.


Read/write capacity mode can be changed later.

☐ Provisioned (free-tier eligible)

☒ On-demand

**Last change to on-demand mode:** No read/write capacity mode changes have been made.

**Next available change to on-demand mode:** You can update to on-demand mode at any time.

**Provisioned capacity** 

Not applicable because read/write capacity mode is on-demand.

**Auto Scaling**

Not applicable because read/write capacity mode is on-demand.

**Save** **Cancel**

Configure on-demand capacity mode for existing or new Amazon DynamoDB Tables

<https://aws.amazon.com/blogs/aws/amazon-dynamodb-on-demand-no-capacity-planning-and-pay-per-request-pricing/>



Implementation time: *Hours*

# Use Compute Savings Plans

# Scenario

- You have Amazon EC2 or AWS Fargate workloads that are always on
- You are leveraging AWS Lambda in your architecture
- You are paying the default on-demand pricing

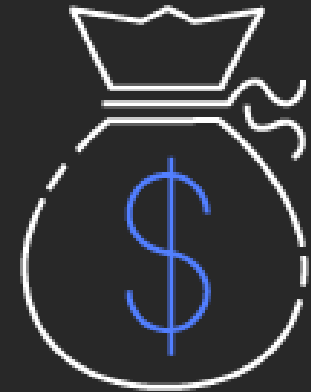


Solution: Compute Savings Plans

# Compute Savings Plans

Provides the most flexibility across...

- Instance family: e.g. Move from C5 to M5
- Region: e.g. change from EU (Ireland) to EU (London)
- OS: e.g. Windows to Linux
- Tenancy: e.g. switch dedicated tenancy to default tenancy
- Compute options: e.g. move from Amazon EC2 to AWS Fargate or AWS Lambda





# Choose 1 year, no upfront Compute Savings Plans

Implementation time	Savings potential	Time to realize savings	Commitment required
Hours	Up to 54% (Amazon EC2), 20% (AWS Fargate), 12% (AWS Lambda) less than on-demand	Hours	1 year, No upfront costs

# Get started

## AWS Cost Explorer will provide you with Savings Plans recommendations

1

Review your Savings Plans recommendations in the AWS Cost Explorer

2

Customize recommendations based on your needs (Term length: 1 Year, payment option: no upfront)

3

Add preferred Savings Plans amount to cart and purchase

savings plans / purchase recommendations

Recommendation options

Savings Plans type: ☒ Compute ☐ EC2 Instance

Savings Plans term: ☒ 1 year ☐ 3 year

Payment option: ☒ All upfront ☐ Partial upfront ☐ No upfront

Based on the past: ☐ 1 day ☒ 30 days ☐ 90 days

Recommendation: Purchase a Compute Savings Plan at a commitment of \$0.30/hour

You could save an estimated \$48 monthly by purchasing the recommended Compute Savings Plan.

Based on your past 30 days of usage, we recommend purchasing a Savings Plan with a commitment of \$0.30/hour for a 1-year term. With this commitment, we project that you could save an average of \$0.07/hour - representing a 14% savings compared to On-Demand. To account for variable usage patterns, this recommendation maximizes your savings by leaving an average \$0.09/hour of On-Demand spend.

Before recommended purchase	After recommended purchase Based on your past 30 days of usage	
Monthly On-Demand spend	Estimated monthly spend	Estimated monthly savings
\$329 (\$0.45/hour)	\$281 (\$0.39/hour)	\$48 (\$0.07/hour)
Your estimated On-Demand spend based on your usage over the past 30 days (including all active Savings Plans)	Your recommended \$0.30/hour Savings Plan commitment + an average \$0.09/hour of On-Demand spend	14% monthly savings over On-Demand \$329 - \$281 = \$48

This recommendation estimates your usage over the past 30 days (including your existing Savings Plans and EC2 Reserved Instances) and calculates what your costs would have been had you purchased the recommended Savings Plans. See applicable rules for Savings Plans [here](#). To generate this recommendation, AWS simulates your bill for different commitment amounts and recommends the commitment amount that provides the greatest estimated savings. [Learn more](#)

Recommended Compute Savings Plans

Term	Payment option	Recommended commitment	Estimated hourly savings
1 year	All upfront	\$0.30/hour	\$0.07 (14%)

[Download CSV](#) [Add selected Savings Plans to cart](#)

# Final thoughts



To understand  
your costs

use



AWS  
Cost Explorer



To control  
your costs

use



AWS Budgets



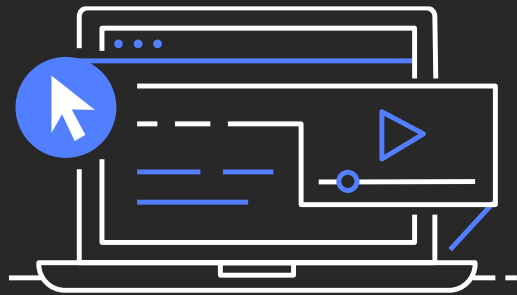
To optimize  
your costs

use



AWS  
Recommendations

# AWS Digital Training



## Flexibility to Learn Your Way

Build cloud skills with 550+ free digital training courses, or dive deep with classroom training

### Featured Courses

- [AWS Cloud Practitioner Essentials \(Second Edition\)](#)  
Learn the fundamentals of the AWS Cloud and prepare for the AWS Certified Cloud Practitioner exam.
- [Amazon DynamoDB for Serverless Architectures](#)  
An introduction to Amazon DynamoDB and how it's leveraged in building a serverless architecture.
- [AWS Security Fundamentals](#)  
Learn fundamental cloud computing and AWS security concepts, including AWS access control and management, governance, logging, and encryption methods.
- [Getting Started with Amazon Simple Storage Service \(Amazon S3\)](#)  
The course provides you with the knowledge to determine when to use Amazon S3 by reviewing typical use cases and understanding how the service provides object storage for your applications.

# Thank you for attending AWS Builders Online Series

We hope you found it interesting! A kind reminder to **complete the survey**.  
Let us know what you thought of today's event and how we can improve the event experience for you in the future.



[aws-apac-marketing@amazon.com](mailto:aws-apac-marketing@amazon.com)



[twitter.com/AWSCloud](https://twitter.com/AWSCloud)



[facebook.com/AmazonWebServices](https://facebook.com/AmazonWebServices)



[youtube.com/user/AmazonWebServices](https://youtube.com/user/AmazonWebServices)



[slideshare.net/AmazonWebServices](https://slideshare.net/AmazonWebServices)



[twitch.tv/aws](https://twitch.tv/aws)



# Builders Online Series

# Thank you

Gabe Hollombe

  @gabehollombe