# Leverage Compute Rightsizing to Maximize Savings on AWS

AWS Online Tech Talk
October, 2019

Arthur Basbaum basbauma@amazon.com
AWS Cloud Financial Management (CFM)

Erin Carlson erincarl@amazon.com
Product Marketing Manager AWS

**Erin Carlson**
AWS Product Marketing Manager

**Arthur Basbaum**
AWS Cloud Economics

## Agenda

- Problem Statement
- Rightsizing Playbook
- AWS Resource Optimization
- Best Practices
- Q&A

AWS Virtual Workshop LEVEL 200

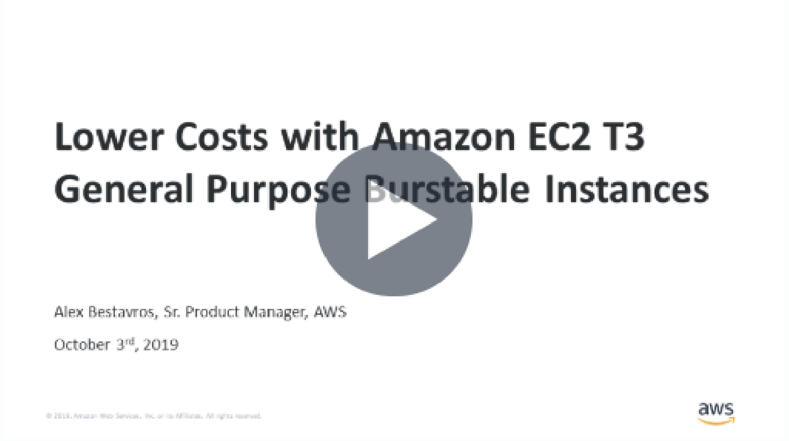
Hands-on Lab: how to monitor and manage your AWS costs

AWS Online Tech Talk
September, 2019

Arthur Basbaum basbauma@amazon.com
AWS Cloud Economics

How to Monitor and Manage Your AWS Costs

AWS Online Tech Talks LEVEL 300


Lower Costs with Amazon EC2 T3 General Purpose Burstable Instances

Alex Bestavros, Sr. Product Manager, AWS
October 3rd, 2019

Lower Costs by Right Sizing Your Instance with Amazon EC2 T3 General Purpose Burstable Instances

https://aws.amazon.com/about-aws/events/monthlywebinarseries/on-demand/

# Cloud Financial Management (CFM) Framework

## Measurement and accountability

Account & tagging strategy

Cost reporting & monitoring processes

Cost show/chargeback

Efficiency and value KPIs

## Cost Optimization

Match capacity with demand

Cost aware architecture, design & service selection

Choose the right pricing model

Identify resource waste

## Planning and forecasting

POC based cost estimation

Budgeting & forecasting variable cloud usage

Business case and value articulation

Strategic fit

## **Cloud financial operations**

Secure executive sponsorship

Partnership between Finance & Technology organizations

Invest in people, governance & tools

Celebrate accomplishments

# Some of the root causes of cloud waste

## Learning curve associated with:

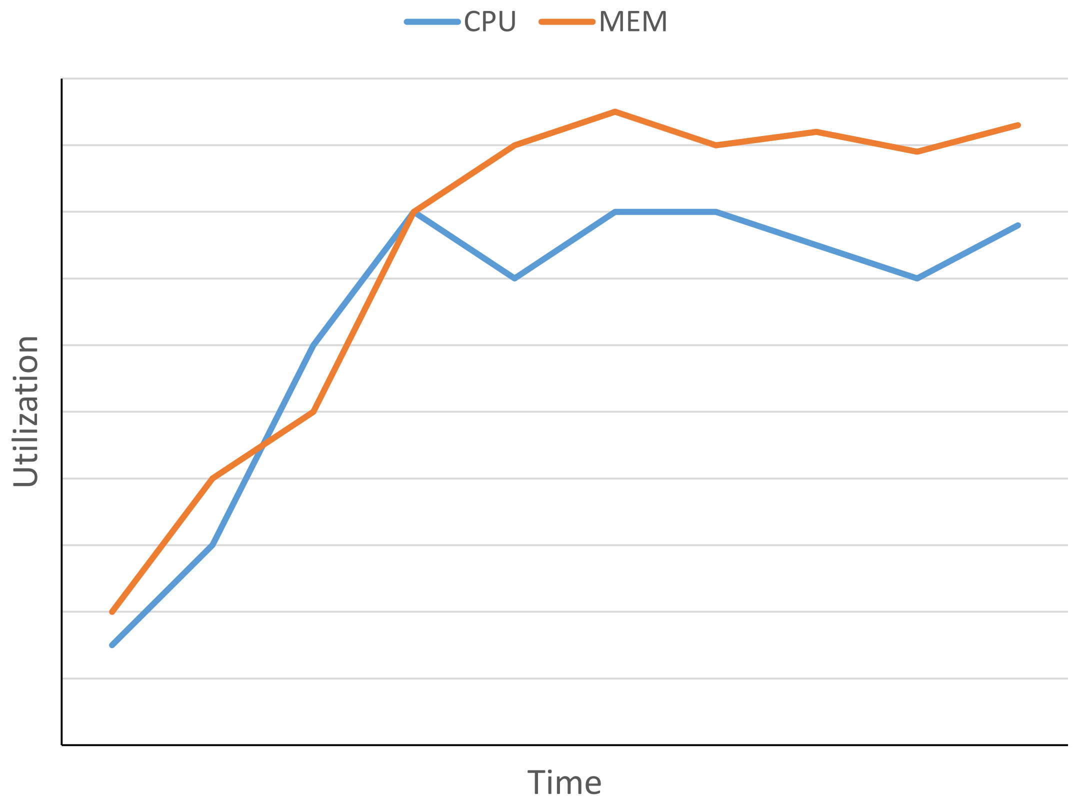
- Managing access to on-demand resources
- Understanding Cloud Pricing options
- Selecting optimal services and resource types/sizes
- Predicting the cost associated with variable usage
- Awareness of resource costs
- Cost governance in a continuous manner

## Amazon EC2 Naming Explained

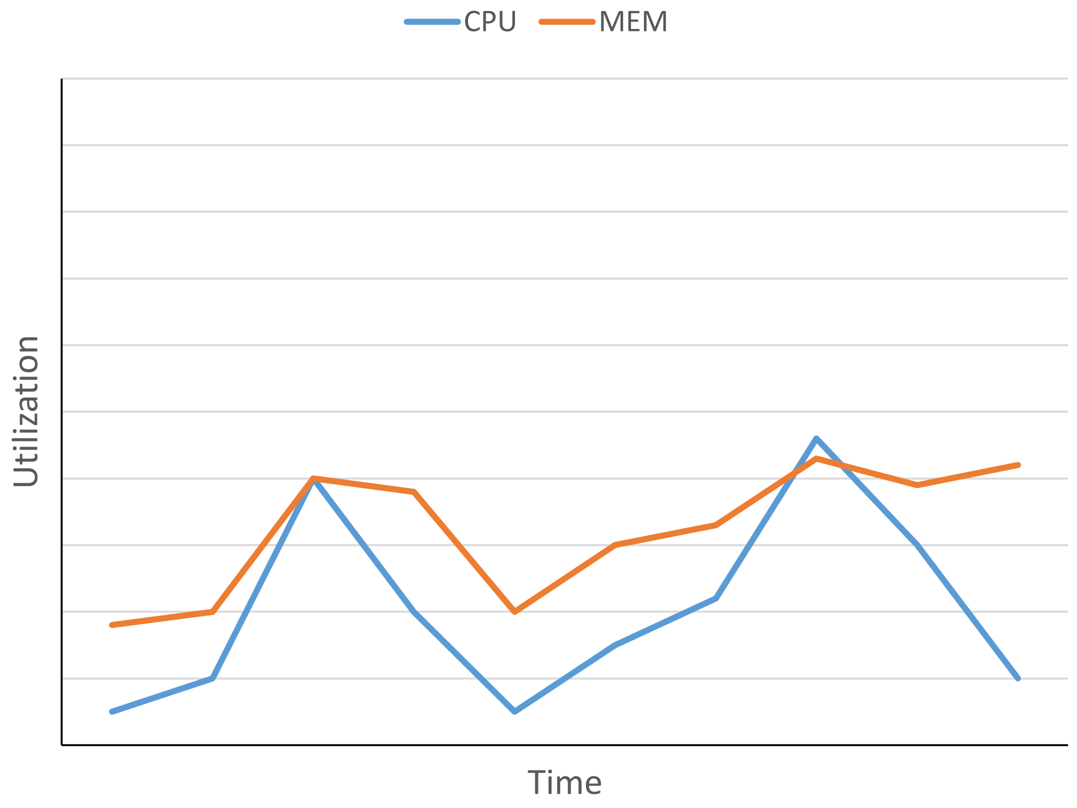
$$\underbrace{\texttt{c}}_{\text{Instance family}}\overbrace{\texttt{5}}^{\text{Instance generation}}\underbrace{\texttt{n}}_{\text{Attribute}}\texttt{.}\underbrace{\texttt{xlarge}}_{\text{Instance size}}$$

# How much performance do you need?

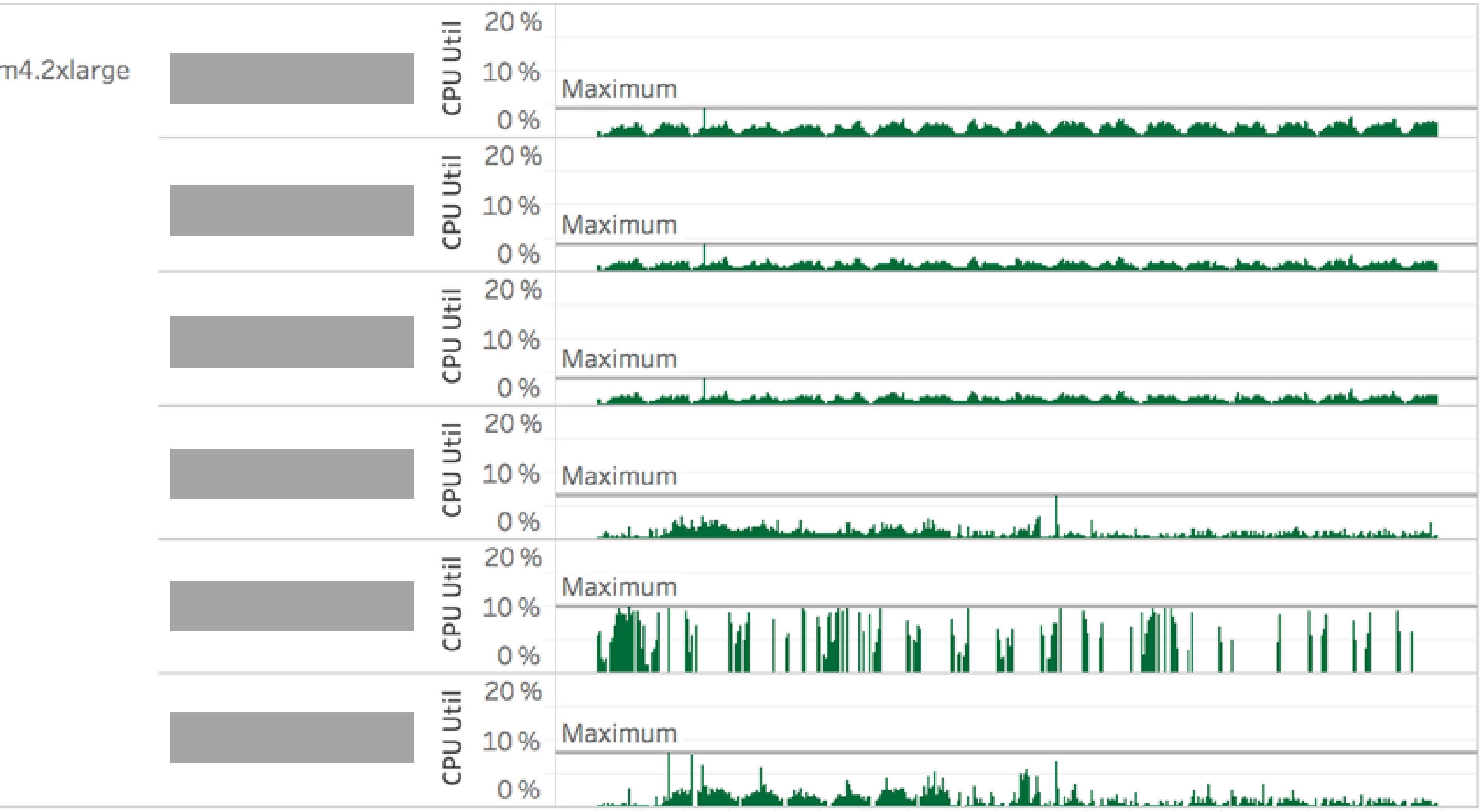
Continuous high MEM/CPU usage



Variable MEM/CPU usage



# Overprovisioning is costly



m4.2xlarge Linux Virginia
8 VCPU 32 Gb RAM
$0.40/hr
$3,504/yr

m4.xlarge Linux Virginia
4 VCPU 16 Gb RAM
$0.20/hr
$1,752/yr

**50% savings**

How much do you think you can save if the compute resources at your company were right sized?

Poll #1

## Broadest and deepest platform choice

### Categories

General purpose

Burstable

Compute intensive

Memory intensive

Storage (High I/O)

Dense storage

GPU compute

Graphics intensive

+

### Capabilities

Choice of processor
(AWS, Intel, AMD)

Fast processors
(up to 4.0 GHz)

High memory footprint
(up to 12 TiB)

Instance storage
(HDD and NVMe)

Accelerated computing
(GPUs and FPGA)

Networking
(up to 100 Gbps)

Bare Metal

Size
(Nano to 32xlarge)

+

### Options

Elastic Block Store

Elastic Graphics

Elastic Inference

=

**200+**
instance types
for virtually
every workload
and business need

# Before looking at the data, two quick tips:

## Newer generations are normally cheaper

- Last generations have a higher performance and normally cost less*

| | Linux | vs new gen | Windows | vs new gen |
|---|---|---|---|---|
| **c3.large** | $0.105/hr | +19% | $0.188/hr | +5% |
| **c4.large** | $0.100/hr | +15% | $0.192/hr | +7% |
| **c5.large** | $0.085/hr | 0% | $0.177/hr | 0% |
| **m3.large** | $0.133/hr | +27% | $0.259/hr | +27% |
| **m4.large** | $0.100/hr | +4% | $0.192/hr | +2% |
| **m5.large** | $0.096/hr | 0% | $0.188/hr | 0% |

* Amazon EC2 Virginia On Demand prices

## Consider testing the "T" family for your workloads

- If your workloads doesn't require a continuous high CPU usage the T family can be very cost effective**

| | VPU | MEM | Linux | vs t3 |
|---|---|---|---|---|
| **t3.large** | 2* | 8 Gb | $0.083/hr | |
| **m5.large** | 2 | 8 Gb | $0.096/hr | +13% |
| **c5.large** | 2 | 4 Gb | $0.085/hr | +2% |
| **r5.large** | 2 | 16 Gb | $0.126/hr | +34% |

* For a 7h 12min burst

** Amazon EC2 Virginia On Demand prices

# Overprovisioning Root Causes

## Timing & Prioritization

*"Let's get it up and running, optimize it later"*

## Missing Telemetry

Lack of historical data: seasonality and peaks.
Telemetry coverage gaps
CPU, Memory, Network, …

## Operational Unknowns

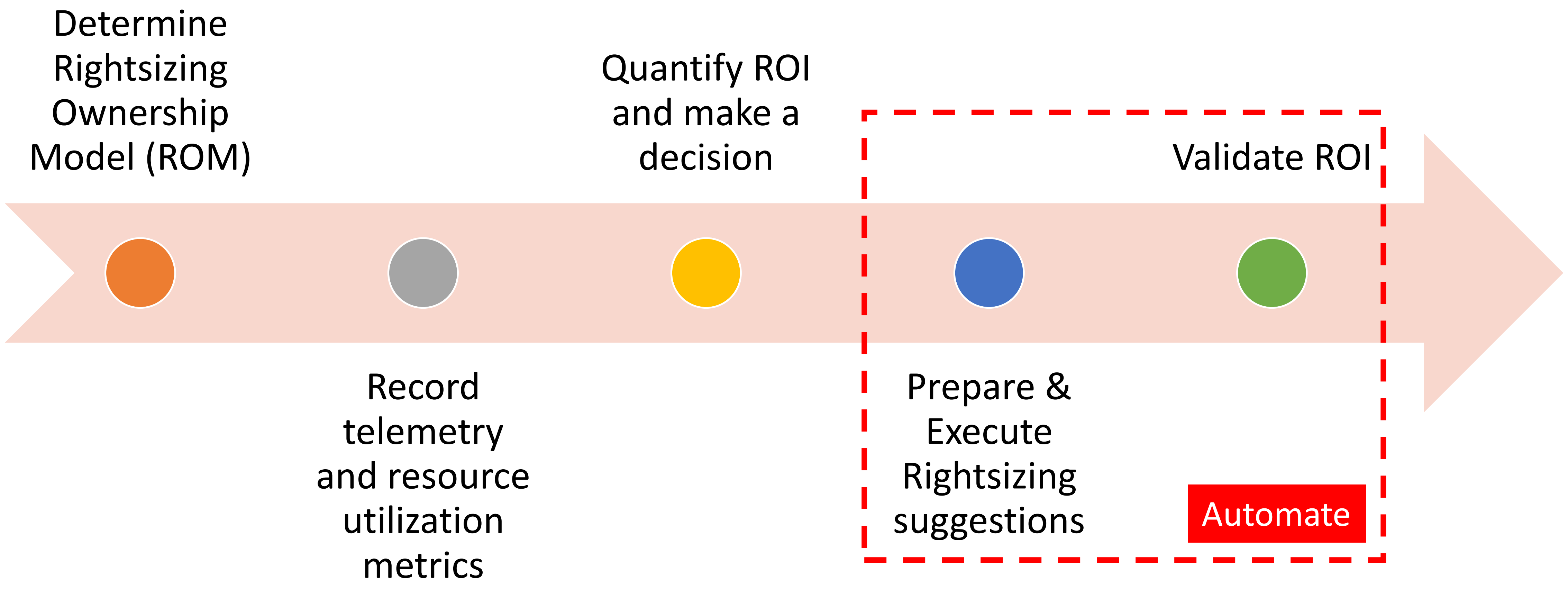
*Who owns rightsizing?*
*How is rightsizing performed? (Ad Hoc, Scheduled, Automated)*

What were the main challenges/blockers you experienced when trying to right size resources?

Poll #2

## How to build a consistent approach around Compute right sizing on AWS?

# Rightsizing Playbook


Determine Rightsizing Ownership Model (ROM)
Record telemetry and resource utilization metrics
Quantify ROI and make a decision
Prepare & Execute Rightsizing suggestions
Validate ROI
Automate

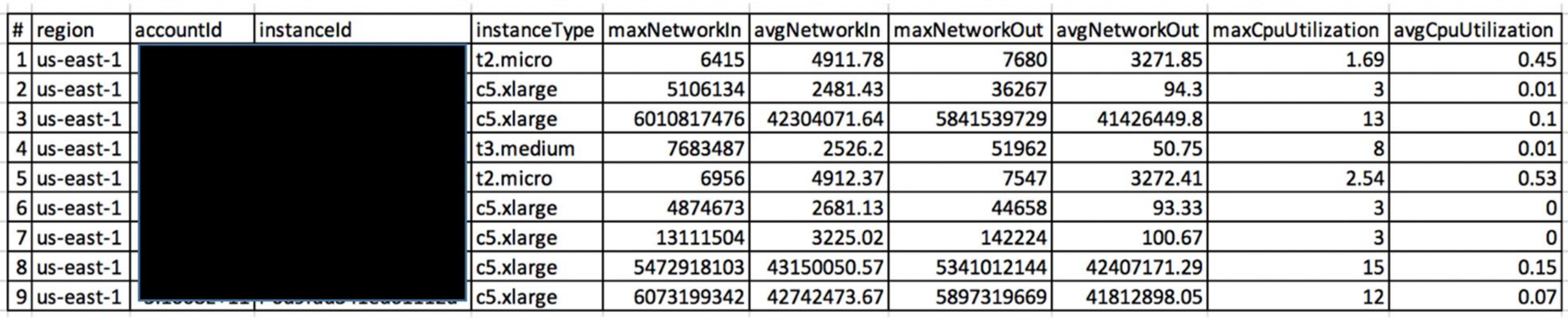
## Step 1: Determine Rightsizing Ownership Model (ROM)

Who will own all the pieces to get to the finish line?

| Example Tasks | Owner |
|---|---|
| Deploying/configuring telemetry | NOC, SRE, Site Ops, DevOps, …? |
| Identifying underutilized resources | Centralized, Decentralized, Hybrid, 3rd Party Tool? |
| Identifying target rightsized instances & quantifying potential benefit $ | Centralized, Decentralized, Hybrid, 3rd Party Tool? |
| Preparing a rightsizing plan/runbook | Centralized, Decentralized, Hybrid? |
| Executing the rightsizing plan/runbook | Centralized, Decentralized, Hybrid? |

## Step 2: Record Telemetry/Resource Utilization Metrics

- Which infrastructure metrics matter?
  - CPU, Memory, Network, Disk (Volume, IOPS), …
  - Max, Min or Avg?
- Does deployed monitoring capture the metrics that matter?
- Can utilization metrics be extracted in a consumable format?
- Can utilization metrics be extracted for relevant historical time windows?
- **What gaps need to be addressed, and how quickly can they be addressed?**



| # | region | accountId | instanceId | instanceType | maxNetworkIn | avgNetworkIn | maxNetworkOut | avgNetworkOut | maxCpuUtilization | avgCpuUtilization |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | us-east-1 | [redacted] | [redacted] | t2.micro | 6415 | 4911.78 | 7680 | 3271.85 | 1.69 | 0.45 |
| 2 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 5106134 | 2481.43 | 36267 | 94.3 | 3 | 0.01 |
| 3 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 6010817476 | 42304071.64 | 5841539729 | 41426449.8 | 13 | 0.1 |
| 4 | us-east-1 | [redacted] | [redacted] | t3.medium | 7683487 | 2526.2 | 51962 | 50.75 | 8 | 0.01 |
| 5 | us-east-1 | [redacted] | [redacted] | t2.micro | 6956 | 4912.37 | 7547 | 3272.41 | 2.54 | 0.53 |
| 6 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 4874673 | 2681.13 | 44658 | 93.33 | 3 | 0 |
| 7 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 13111504 | 3225.02 | 142224 | 100.67 | 3 | 0 |
| 8 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 5472918103 | 43150050.57 | 5341012144 | 42407171.29 | 15 | 0.15 |
| 9 | us-east-1 | [redacted] | [redacted] | c5.xlarge | 6073199342 | 42742473.67 | 5897319669 | 41812898.05 | 12 | 0.07 |

*Example: Default CloudWatch Metrics*

## Step 3: Quantify the ROI and Make a Decision

- Decide on the resources to be rightsized and target resource types
    - Rightsize all or a subset of resources?
    - What resource utilization thresholds don’t jeopardize service quality?
- Calculate estimated savings **(A)**
- Estimate rightsizing costs **(B)**
    - Deploying/configuring/fixing monitoring & telemetry
    - Usage analysis, target type identification
    - Planning
    - Execution
- Decide: Is the difference between (A) and (B) large enough to move forward?
    - Alternatively: are there specific resources where **(A) – (B) = N**, where N is significantly larger than 0?

## Step 4: Prepare and Execute the Rightsizing Plan

- Start small, build muscle, gain confidence, repeat
- Planning
  - What kind of functional and performance testing needs to be executed?
    - Intra family rightsizing: Skip functional tests?
    - Inter family rightsizing: Perform both functional and performance testing?
  - Which environments to start with?
    - Test/Dev Environments vs. Staging vs. Pre Prod vs. Prod vs. [ DR | … ]
  - How to maintain end-user SLAs
  - How to handle auto-launches by ASGs & Clusters
  - Leveraging existing product release cadence, scheduled maintenance windows or change control processes

## Step 4: Prepare and Execute the Rightsizing Plan (cont.)

### Centralized Ownership Model

- Single individual/org performs following:

1 Analyze resource utilization
2 Determine target resource types, ROI
**3 Receive sign off on target types from resource owners**
4 Create resource based runbook
5 Execute rightsizing
6 Validate ROI

## Step 4: Prepare and Execute the Rightsizing Plan (cont.)

### Runbook example

| Target Date | Old Resource ID | Environment | Team | Rightsizing Owner | New Resource ID | Status |
|---|---|---|---|---|---|---|
| 3/8/2019 | i-23nlkj23azv332 | Staging | Data Science | John Doe | i-3959325zaafs | Completed |
| 3/8/2019 | i-98jjhdsaf9325 | Staging | Data Science | John Doe | i-98jjhdsaf9325 | Completed |
| 3/8/2019 | i-53259750235 | Staging | Data Science | John Doe | i-59023bnjko23 | Completed |
| 3/8/2019 | I-957272358952 | Staging | Data Science | John Doe | I-957272358952 | Completed |
| 3/8/2019 | i-95y732892537 | Staging | Data Science | John Doe | i-95mm325jk25 | Completed |
| … | … | | … | … | | … |

## Step 4: Prepare and Execute the Rightsizing Plan (cont.)

### Decentralized Ownership Model

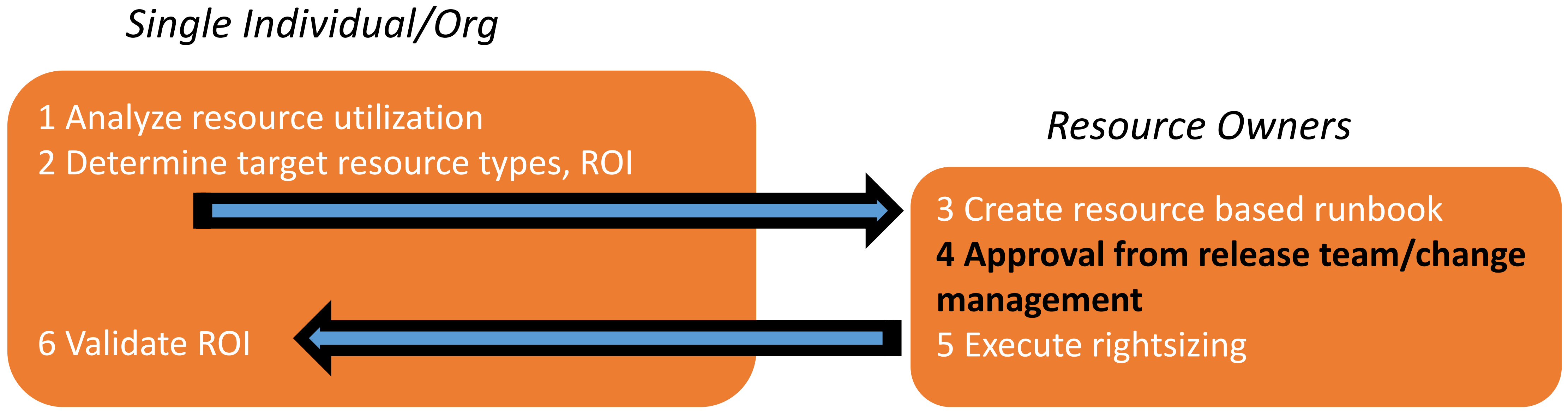
- Resource owners (i.e. product teams) performs following:

1 Analyze resource utilization
2 Determine target resource types, ROI
3 Create resource based runbook
**4 Approval from release team/change management**
5 Execute rightsizing
6 Validate ROI

## Step 4: Prepare and Execute the Rightsizing Plan (cont.)

### Hybrid Ownership Model

- Combined effort:


*Single Individual/Org*
1 Analyze resource utilization
2 Determine target resource types, ROI
6 Validate ROI
*Resource Owners*
3 Create resource based runbook
**4 Approval from release team/change management**
5 Execute rightsizing

## Step 5: Validate the ROI

- **Actual Savings (C)**
  - Compare old vs. new AWS spend run rate for rightsized resources
- **Actual Cost (D)**
  - Calculate actual total rightsizing cost
    - Deploying/configuring/fixing monitoring & telemetry
    - Usage analysis, target type identification
    - Planning
    - Execution
- **Actual ROI = (C)/(D)*100**
- **ROI Variance = (Actual ROI – Estimated ROI)/(Estimated ROI)*100**

## Step 6: Automate

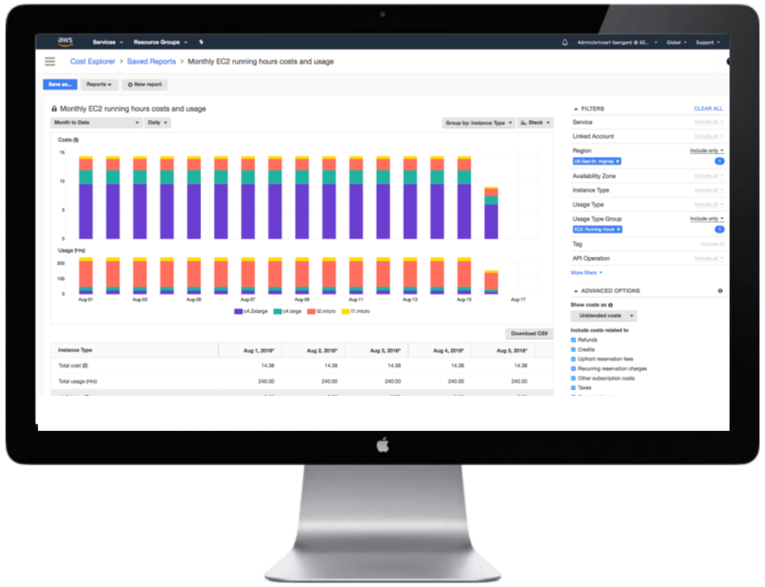
- What portions of Step 1 through 5 can be automated?
- Examples
  - **Step 2:** Ensure telemetry gaps are solved for by complete monitoring, infrastructure bootstrapping or configuration management
  - **Step 3:** Savings calculations via AWS Price List API
  - **Step 4:** CI/CD Pipeline for functional testing, automated performance testing
  - **Step 5:** Cost Explorer API "GetCostAndUsage"

## What tools are natively available on AWS for Compute right sizing?

Do you use any third-party tool to help you identify underutilized resources?

Poll #3

# AWS Cost Management Product Suite



Cost Explorer > Saved Reports > Monthly EC2 running hours costs and usage

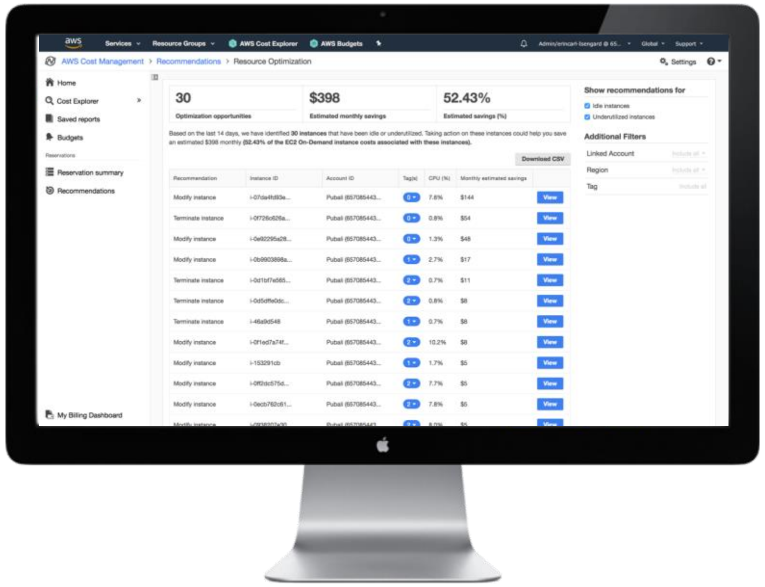
Monthly EC2 running hours costs and usage

| Instance Type | Aug 1, 2018* | Aug 2, 2018* | Aug 3, 2018* | Aug 4, 2018* | Aug 5, 2018* |
|---|---|---|---|---|---|
| Total cost ($) | 14.38 | 14.38 | 14.38 | 14.38 | 14.38 |
| Total usage (Hrs) | 240.00 | 240.00 | 240.00 | 240.00 | 240.00 |

| | |
|---|---|
| Service | Include all |
| Linked Account | Include all |
| Region | Include only: US East (N. Virginia) |
| Availability Zone | Include all |
| Instance Type | Include all |
| Usage Type | Include all |
| Usage Type Group | Include only: EC2: Running Hours |
| Tag | Include All |
| API Operation | Include all |

Show costs as: Unblended costs

Include costs related to
- ☑ Refunds
- ☑ Credits
- ☑ Upfront reservation fees
- ☑ Recurring reservation charges
- ☑ Other subscription costs
- ☑ Taxes



AWS Cost Management > Recommendations > Resource Optimization

30 Optimization opportunities | $398 Estimated monthly savings | 52.43% Estimated savings (%)

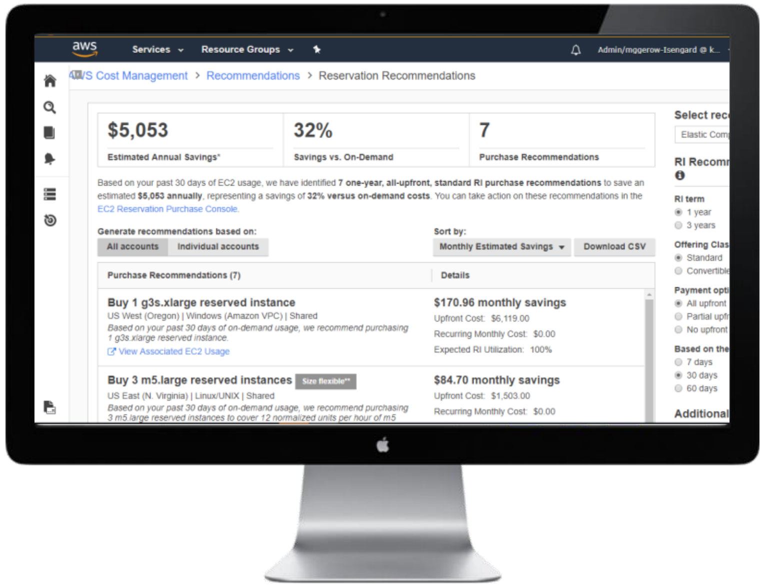
Based on the last 14 days, we have identified **30 instances** that have been idle or underutilized. Taking action on these instances could help you save an estimated $398 monthly **(52.43% of the EC2 On-Demand instance costs associated with these instances).**

| Recommendation | Instance ID | Account ID | Tag(s) | CPU (%) | Monthly estimated savings |
|---|---|---|---|---|---|
| Modify instance | i-07da4fd93e... | Pubali (657085443... | 0 | 7.6% | $144 |
| Terminate instance | i-0f726c626a... | Pubali (657085443... | 0 | 0.8% | $54 |
| Modify instance | i-0e92295a28... | Pubali (657085443... | 0 | 1.3% | $48 |
| Modify instance | i-0b9903898a... | Pubali (657085443... | 1 | 2.7% | $17 |
| Terminate instance | i-0d1bf7e565... | Pubali (657085443... | 2 | 0.7% | $11 |
| Terminate instance | i-0d5dffe0dc... | Pubali (657085443... | 2 | 0.8% | $8 |
| Terminate instance | i-46a9d548 | Pubali (657085443... | 1 | 0.7% | $8 |
| Modify instance | i-0f1ed7a74f... | Pubali (657085443... | 2 | 10.2% | $8 |
| Modify instance | i-153291cb | Pubali (657085443... | 1 | 1.7% | $5 |
| Modify instance | i-0ff2dc575d... | Pubali (657085443... | 2 | 7.7% | $5 |
| Modify instance | i-0ecb762c61... | Pubali (657085443... | 2 | 7.8% | $5 |

Show recommendations for
- ☑ Idle instances
- ☑ Underutilized instances

Additional Filters

| | |
|---|---|
| Linked Account | Include all |
| Region | Include all |
| Tag | Include all |



AWS Cost Management > Recommendations > Reservation Recommendations

$5,053 Estimated Annual Savings* | 32% Savings vs. On-Demand | 7 Purchase Recommendations

Based on your past 30 days of EC2 usage, we have identified **7 one-year, all-upfront, standard RI purchase recommendations** to save an estimated **$5,053 annually**, representing a savings of **32% versus on-demand costs**. You can take action on these recommendations in the EC2 Reservation Purchase Console.

Generate recommendations based on: [✓ All accounts] Individual accounts

Sort by: Monthly Estimated Savings

| Purchase Recommendations (7) | Details |
|---|---|
| **Buy 1 g3s.xlarge reserved instance** US West (Oregon) \| Windows (Amazon VPC) \| Shared *Based on your past 30 days of on-demand usage, we recommend purchasing 1 g3s.xlarge reserved instance.* View Associated EC2 Usage | **$170.96 monthly savings** Upfront Cost: $6,119.00 Recurring Monthly Cost: $0.00 Expected RI Utilization: 100% |
| **Buy 3 m5.large reserved instances** Size flexible** US East (N. Virginia) \| Linux/UNIX \| Shared *Based on your past 30 days of on-demand usage, we recommend purchasing 3 m5.large reserved instances to cover 12 normalized units per hour of m5* | **$84.70 monthly savings** Upfront Cost: $1,503.00 Recurring Monthly Cost: $0.00 |

RI term: [✓ 1 year] 3 years

Offering Clas: [✓ Standard] Convertible

Payment opti: [✓ All upfront] Partial upfr No upfront

Based on the: 7 days [✓ 30 days] 60 days

**AWS Cost Explorer**

**AWS Resource Optimization**

**AWS RI Reports**

# AWS Cost Explorer

## Comprehensive dashboards

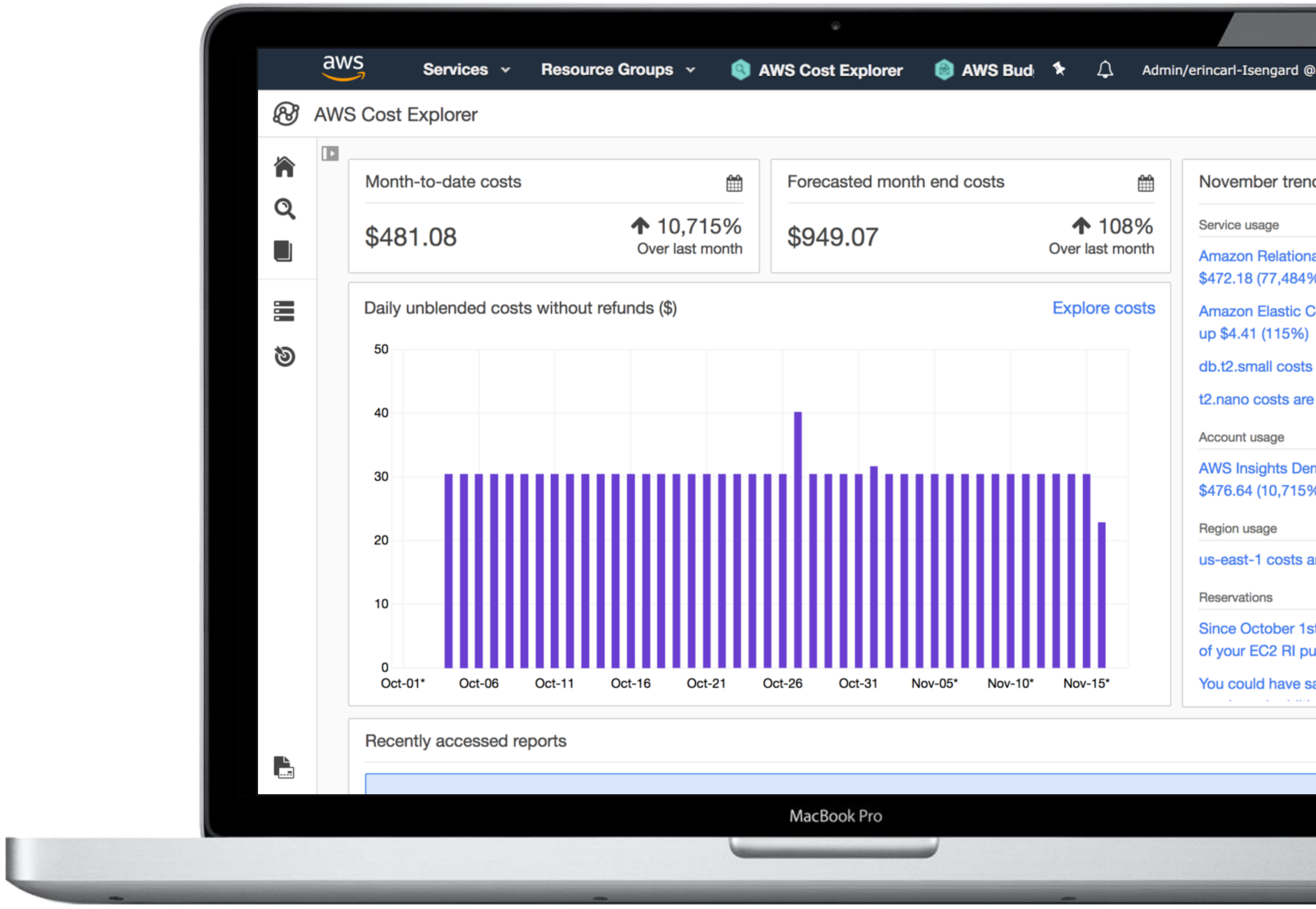
Gain a summary view of key cost details, including month-to-date costs, month-end forecasted costs, and saved reports

## Automated trend analysis

Identifies anomalous cost and usage events, across your account(s), based on historical patterns

## Optimized user experience

Users of all levels of expertise in your organization can quickly onboard and feel confident using Cost Explorer to address their cost management needs



Month-to-date costs
$481.08 ↑ 10,715% Over last month

Forecasted month end costs
$949.07 ↑ 108% Over last month

November tren

Service usage
Amazon Relationa
$472.18 (77,484%
Amazon Elastic C
up $4.41 (115%)
db.t2.small costs
t2.nano costs are

Account usage
AWS Insights Der
$476.64 (10,715%

Region usage
us-east-1 costs a

Reservations
Since October 1s
of your EC2 RI pu
You could have sa

Daily unblended costs without refunds ($)

Recently accessed reports

# AWS Cost Explorer

## Plan for future spending

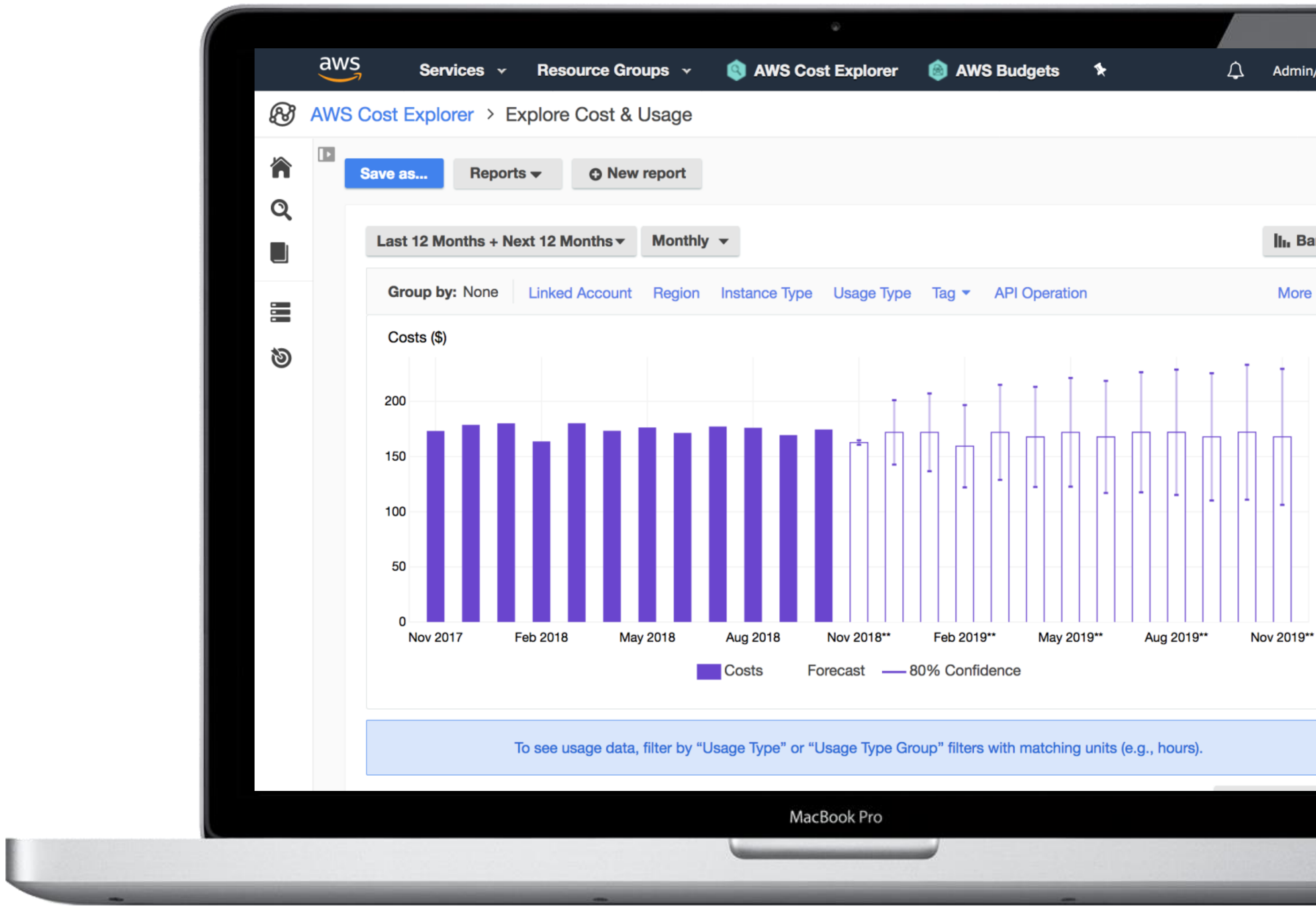
Increased forecasting accuracy due to new machine learning- and rules-based models, allowing you to plan ahead with more confidence

## Customize your forecasts

Use Cost Explorer's filtering capabilities to forecast costs along specific usage dimensions

## Programmatic access

Forecasting functionality is also available via the Cost Explorer API



To see usage data, filter by "Usage Type" or "Usage Type Group" filters with matching units (e.g., hours).

# AWS Cost Explorer Resource Optimization

## Address the challenges of identifying optimization opportunities at scale

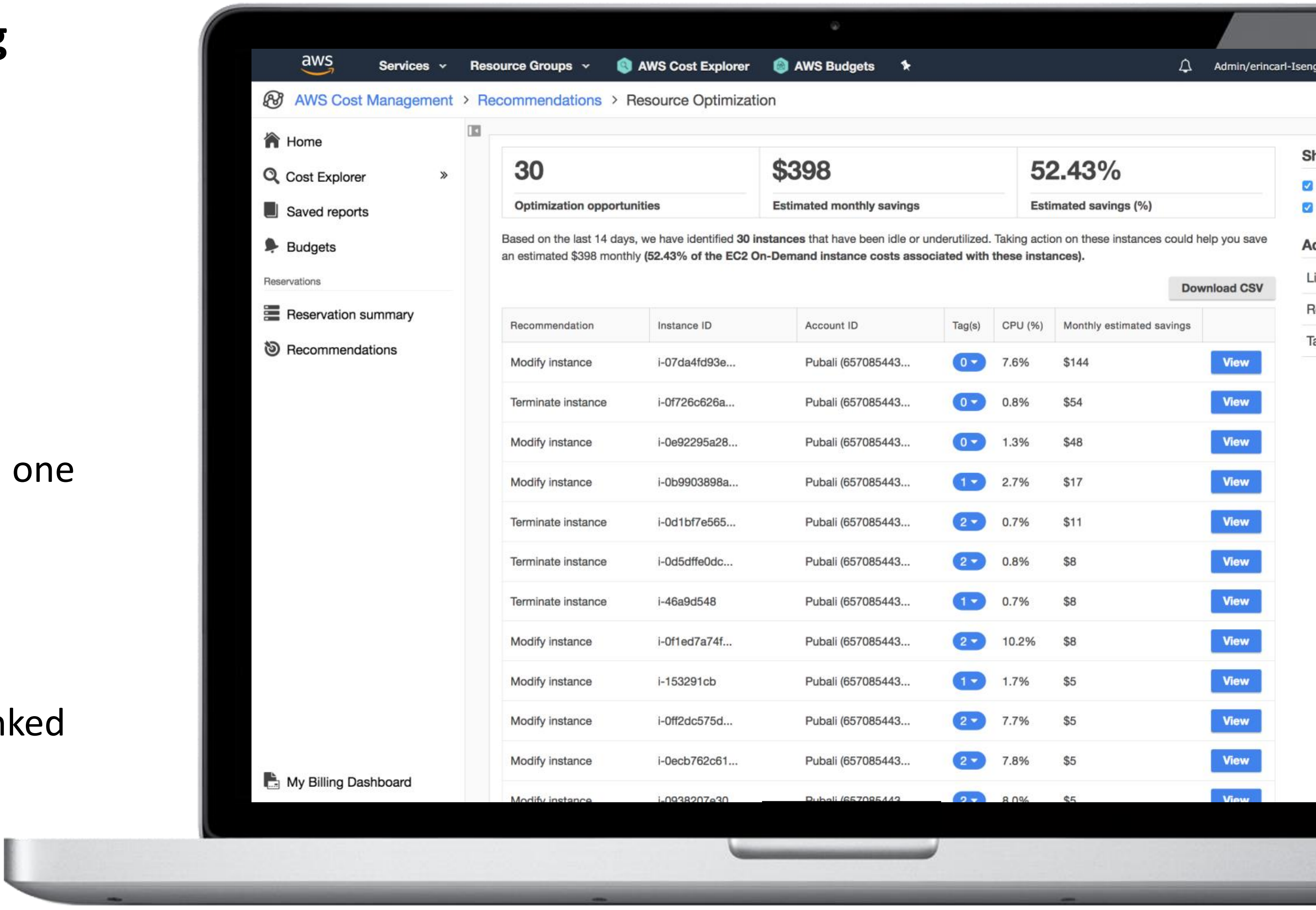
- Get customized Amazon EC2 Rightsizing Recommendations for free

## Single view across your regions and accounts

- Exhaustive view of your opportunities from one place

## Filter your recommendations

- Based on Cost Allocation Tag, Region, or Linked Account.



30 Optimization opportunities | $398 Estimated monthly savings | 52.43% Estimated savings (%)

Based on the last 14 days, we have identified **30 instances** that have been idle or underutilized. Taking action on these instances could help you save an estimated $398 monthly **(52.43% of the EC2 On-Demand instance costs associated with these instances).**

| Recommendation | Instance ID | Account ID | Tag(s) | CPU (%) | Monthly estimated savings | |
|---|---|---|---|---|---|---|
| Modify instance | i-07da4fd93e... | Pubali (657085443... | 0 | 7.6% | $144 | View |
| Terminate instance | i-0f726c626a... | Pubali (657085443... | 0 | 0.8% | $54 | View |
| Modify instance | i-0e92295a28... | Pubali (657085443... | 0 | 1.3% | $48 | View |
| Modify instance | i-0b9903898a... | Pubali (657085443... | 1 | 2.7% | $17 | View |
| Terminate instance | i-0d1bf7e565... | Pubali (657085443... | 2 | 0.7% | $11 | View |
| Terminate instance | i-0d5dffe0dc... | Pubali (657085443... | 2 | 0.8% | $8 | View |
| Terminate instance | i-46a9d548 | Pubali (657085443... | 1 | 0.7% | $8 | View |
| Modify instance | i-0f1ed7a74f... | Pubali (657085443... | 2 | 10.2% | $8 | View |
| Modify instance | i-153291cb | Pubali (657085443... | 1 | 1.7% | $5 | View |
| Modify instance | i-0ff2dc575d... | Pubali (657085443... | 2 | 7.7% | $5 | View |
| Modify instance | i-0ecb762c61... | Pubali (657085443... | 2 | 7.8% | $5 | View |
| Modify instance | i-0938207e30 | Pubali (657085443 | 2 | 8.0% | $5 | View |

# AWS Cost Explorer Resource Optimization

## Identify your idle and underutilized instances

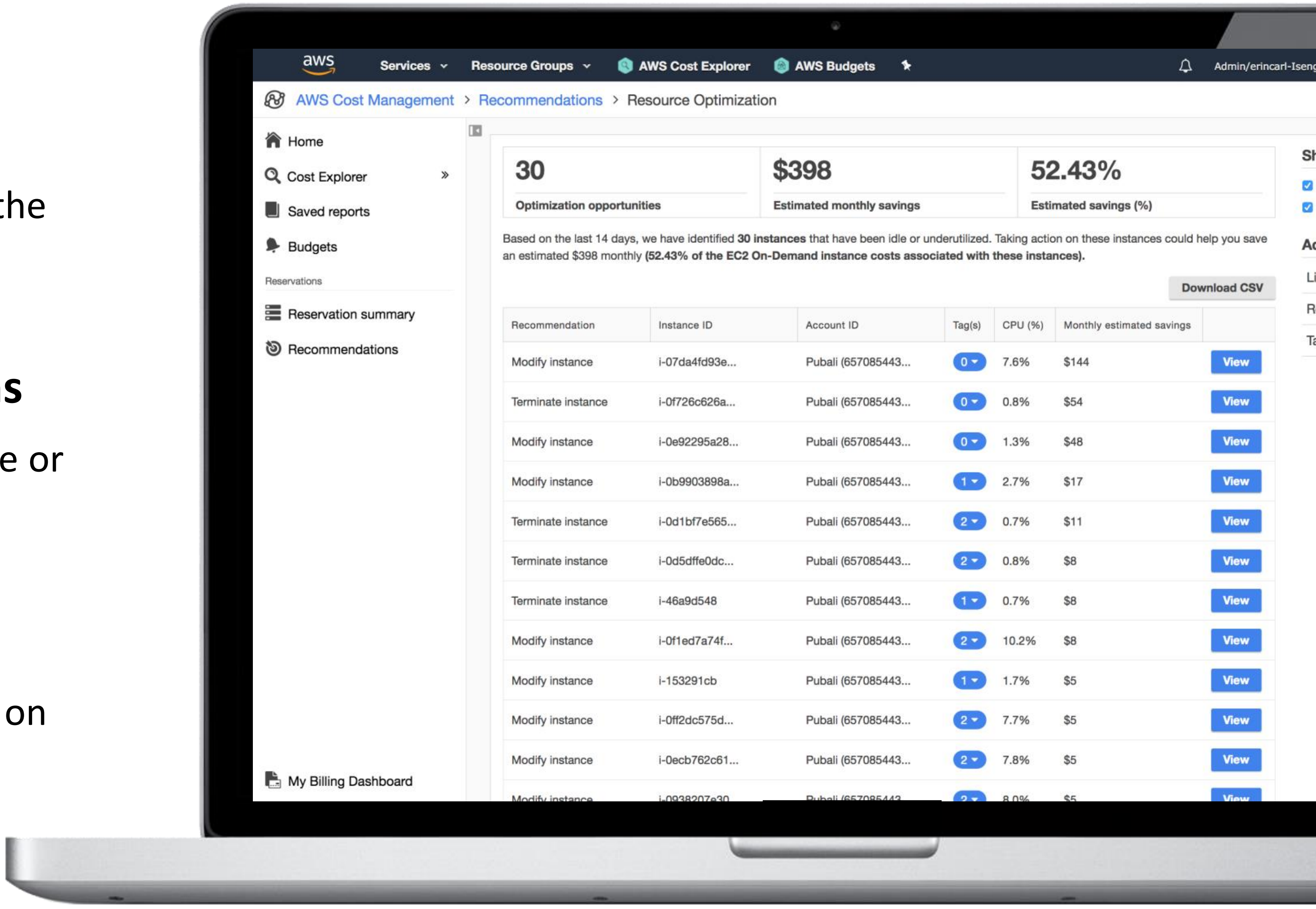
- With the option to enhance those recommendations due to integration with the CloudWatch agent.

## Receive actionable recommendations

- Know the recommended action to downsize or terminate on an instance by instance basis

## Quantify your potential savings

- Understand how much you can save based on your recommended action, and how your reservations impact savings.



| 30 | $398 | 52.43% |
|---|---|---|
| Optimization opportunities | Estimated monthly savings | Estimated savings (%) |

Based on the last 14 days, we have identified **30 instances** that have been idle or underutilized. Taking action on these instances could help you save an estimated $398 monthly **(52.43% of the EC2 On-Demand instance costs associated with these instances).**

| Recommendation | Instance ID | Account ID | Tag(s) | CPU (%) | Monthly estimated savings | |
|---|---|---|---|---|---|---|
| Modify instance | i-07da4fd93e... | Pubali (657085443... | 0 | 7.6% | $144 | View |
| Terminate instance | i-0f726c626a... | Pubali (657085443... | 0 | 0.8% | $54 | View |
| Modify instance | i-0e92295a28... | Pubali (657085443... | 0 | 1.3% | $48 | View |
| Modify instance | i-0b9903898a... | Pubali (657085443... | 1 | 2.7% | $17 | View |
| Terminate instance | i-0d1bf7e565... | Pubali (657085443... | 2 | 0.7% | $11 | View |
| Terminate instance | i-0d5dffe0dc... | Pubali (657085443... | 2 | 0.8% | $8 | View |
| Terminate instance | i-46a9d548 | Pubali (657085443... | 1 | 0.7% | $8 | View |
| Modify instance | i-0f1ed7a74f... | Pubali (657085443... | 2 | 10.2% | $8 | View |
| Modify instance | i-153291cb | Pubali (657085443... | 1 | 1.7% | $5 | View |
| Modify instance | i-0ff2dc575d... | Pubali (657085443... | 2 | 7.7% | $5 | View |
| Modify instance | i-0ecb762c61... | Pubali (657085443... | 2 | 7.8% | $5 | View |
| Modify instance | i-0938207e30 | Pubali (657085443 | 2 | 8.0% | $5 | View |

## Demo Resource Optimization

# Output Example

| Account Name | Instance ID | Instance Type | OS | Region | Running Hrs | RI Hrs | OD Hrs | CPU Util | Rec Action | Rec Instance Type | Estimated Savings |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PROD A | i-XXXXXXXX | r5d.metal | Linux/UNIX | US West (Oregon) | 308 | - | 308 | - | Terminate |  | $ 4,625 |
| PROD A | i-XXXXXXXX | r4.16xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 310 | - | 310 | 0.78 | Terminate |  | $ 2,954 |
| DEV | i-XXXXXXXX | r4.16xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 310 | - | 310 | 0.91 | Terminate |  | $ 2,954 |
| PROD B | i-XXXXXXXX | m4.4xlarge | Windows | US East (N. Virginia) | 308 | - | 308 | 9.05 | Modify | m4.2xlarge | $ 2,548 |
| DEV | i-XXXXXXXX | r3.8xlarge | Linux/UNIX | US West (Oregon) | 309 | - | 309 | 0.32 | Terminate |  | $ 1,786 |
| DEV | i-XXXXXXXX | r3.8xlarge | Linux/UNIX | US West (Oregon) | 309 | - | 309 | 0.31 | Terminate |  | $ 1,786 |
| DEV | i-XXXXXXXX | r3.8xlarge | Linux/UNIX | US West (Oregon) | 307 | - | 307 | 0.41 | Terminate |  | $ 1,774 |
| DEV | i-XXXXXXXX | r3.8xlarge | Linux/UNIX | US West (Oregon) | 308 | 1 | 307 | 0.52 | Terminate |  | $ 1,774 |
| PROD A | i-XXXXXXXX | i3.8xlarge | Red Hat Enterprise Linux | US West (Oregon) | 310 | - | 310 | 0.76 | Terminate |  | $ 1,769 |
| PROD A | i-XXXXXXXX | i3.8xlarge | Red Hat Enterprise Linux | US West (Oregon) | 309 | - | 309 | 0.66 | Terminate |  | $ 1,763 |
| PROD B | i-XXXXXXXX | r4.8xlarge | Red Hat Enterprise Linux | US West (N. California) | 308 | - | 308 | 0.83 | Terminate |  | $ 1,674 |
| DEV | i-XXXXXXXX | r4.8xlarge | Red Hat Enterprise Linux | US West (N. California) | 308 | - | 308 | 0.81 | Terminate |  | $ 1,674 |
| DEV | i-XXXXXXXX | r4.8xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 310 | - | 310 | 0.76 | Terminate |  | $ 1,521 |
| DEV | i-XXXXXXXX | m4.10xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 309 | - | 309 | 0.60 | Terminate |  | $ 1,430 |
| DEV | i-XXXXXXXX | c5.12xlarge | Linux/UNIX | US East (N. Virginia) | 309 | - | 309 | - | Terminate |  | $ 1,370 |
| DEV | i-XXXXXXXX | r4.2xlarge | Windows | US West (N. California) | 308 | - | 308 | 22.71 | Modify | r4.xlarge | $ 1,345 |
| DEV | i-XXXXXXXX | r4.2xlarge | Windows | US East (N. Virginia) | 309 | - | 309 | 28.27 | Modify | r4.xlarge | $ 1,327 |
| DEV | i-XXXXXXXX | c5n.9xlarge | Linux/UNIX | US East (N. Virginia) | 308 | - | 308 | - | Terminate |  | $ 1,301 |
| DEV | i-XXXXXXXX | c5.9xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 309 | - | 309 | - | Terminate |  | $ 1,114 |
| PROD B | i-XXXXXXXX | c5.9xlarge | Red Hat Enterprise Linux | US East (N. Virginia) | 308 | - | 308 | - | Terminate |  | $ 1,111 |

# In Summary, AWS Resource Optimization

- Check Amazon EC2 utilization from the last 14 days
- Differentiate between On Demand and Reserved Instances usage
- Recommend to:
  - Terminate if instance is idle: If the max CPU util is at or below 1%
  - Downsize if instance is underutilized: If the max CPU util is between 1% and 40%
- Can consider Memory utilization in case CloudWatch agent* is enabled
- Estimate potential monthly savings per instance, account and tag
- Recommend up to 3 instances to downsize within the same family

*For more info https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/ce-rightsizing.html

# AWS Cost Explorer RI Recommendations

## Get customized Reservation Recommendations

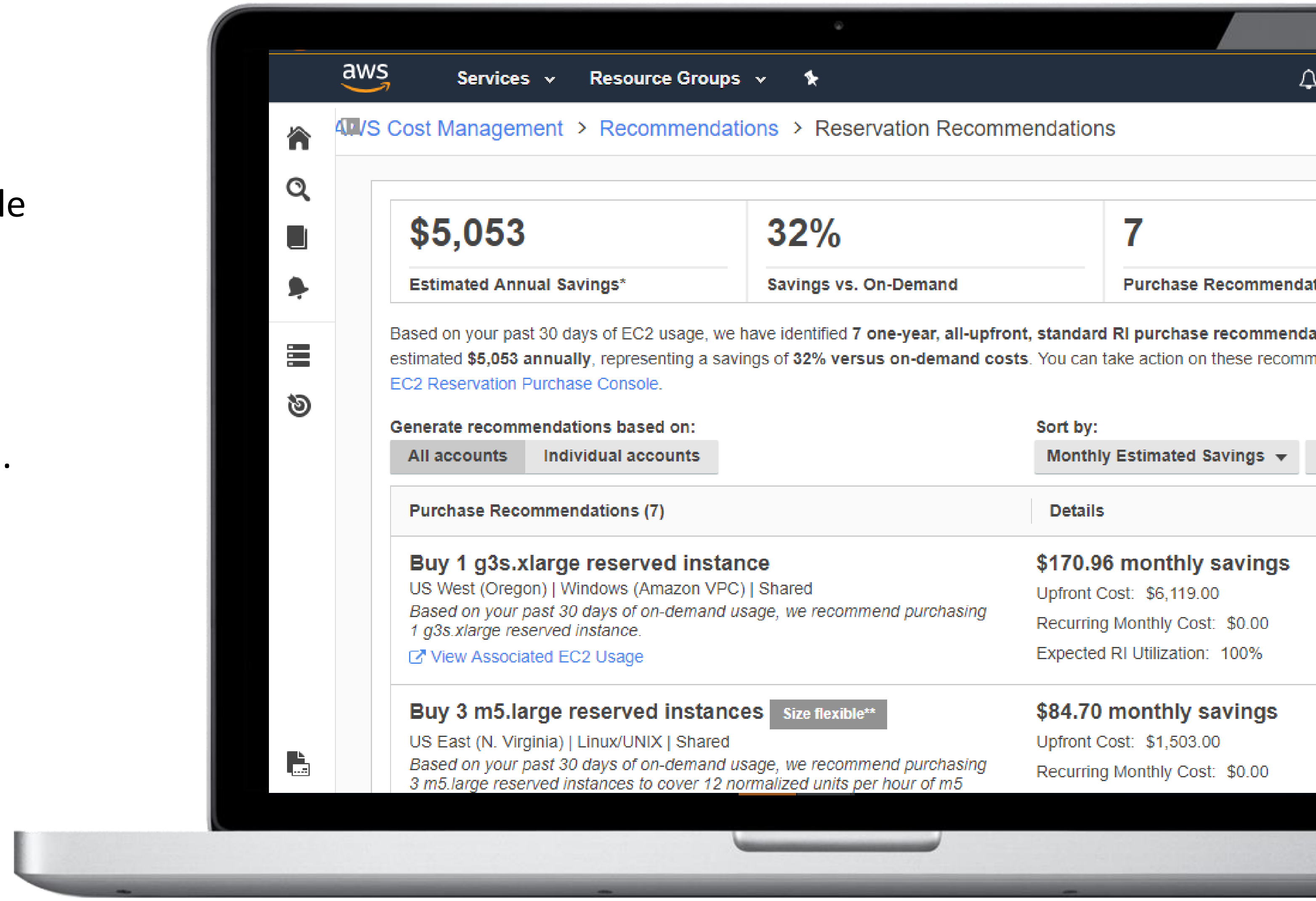
Understand available savings, expected utilization, and cost commitment from a single place

## Available across five AWS services

EC2, RDS, Redshift, ElastiCache, ElasticSearch.

## Define the best recommendation parameters for you

Pick your desired payment option, term, and more.



AWS Cost Management > Recommendations > Reservation Recommendations

| $5,053 | 32% | 7 |
|---|---|---|
| Estimated Annual Savings* | Savings vs. On-Demand | Purchase Recommenda |

Based on your past 30 days of EC2 usage, we have identified **7 one-year, all-upfront, standard RI purchase recommenda** estimated **$5,053 annually**, representing a savings of **32% versus on-demand costs**. You can take action on these recomm EC2 Reservation Purchase Console.

**Generate recommendations based on:** All accounts | Individual accounts

**Sort by:** Monthly Estimated Savings

| Purchase Recommendations (7) | Details |
|---|---|
| **Buy 1 g3s.xlarge reserved instance**<br>US West (Oregon) \| Windows (Amazon VPC) \| Shared<br>*Based on your past 30 days of on-demand usage, we recommend purchasing 1 g3s.xlarge reserved instance.*<br>View Associated EC2 Usage | **$170.96 monthly savings**<br>Upfront Cost: $6,119.00<br>Recurring Monthly Cost: $0.00<br>Expected RI Utilization: 100% |
| **Buy 3 m5.large reserved instances** Size flexible**<br>US East (N. Virginia) \| Linux/UNIX \| Shared<br>*Based on your past 30 days of on-demand usage, we recommend purchasing 3 m5.large reserved instances to cover 12 normalized units per hour of m5* | **$84.70 monthly savings**<br>Upfront Cost: $1,503.00<br>Recurring Monthly Cost: $0.00 |

## Right Sizing Best Practices

## Rightsizing Best Practices

- Start simple: idle resources, non-critical dev/qa and old generation instances
- 'Launch time' can be a good indicator
- Aggregate instances per autoscaling group and tags
- Rightsizing cadence trade offs: Continuous vs. Scheduled rightsizing
- Measure Twice, Cut Once: Test, then test some more
- Co-Term RI covered resources with rightsizing at RI renewal time
    - Standard RIs: Purchase RIs after rightsizing
    - Convertible RIs: Exchange RIs after rightsizing
- ROI analysis may include cost of a 3rd party tool
- Apply Right Sizing learnings for new workloads

Shutdown, hibernate or terminate ;)

# Resources to get you started

AWS Cost Optimization

aws.amazon.com/pricing/cost-optimization/

AWS Well Architect Cost Opt Whitepaper

d1.awsstatic.com/whitepapers/architecture/AWS-Cost-Optimization-Pillar.pdf

Cost Optimization Well Architect Labs

awscostlabs.com

AWS Cost Management Products

aws.amazon.com/aws-cost-management/

FinOps Foundation

finops.org

AWS Cost Management Blog

aws.amazon.com/blogs/aws-cost-management/

Laying the foundation for Cost Opt Whitepaper

d1.awsstatic.com/whitepapers/cost-optimization-laying-the-foundation.pdf

Case studies and research

aws.amazon.com/solutions/case-studies

AWS Cost Management Tools Partners

aws.amazon.com/products/management-tools/partner-solutions/

Do you plan to Right Size your resources after this session?

Poll #4

# Q&A

## Thank you

Arthur Basbaum basbauma@amazon.com
AWS Cloud Financial Management (CFM)

Erin Carlson erincarl@amazon.com
Product Marketing Manager AWS

# Appendix

# Rightsizing Considerations

- Focus on instances that are not covered by RIs
- Exclude resources that are ”off”
- Modernize previous generation families before rightsizing
- Skip rightsizing T family – these are designed to run at low utilization
- Example conditions to be met before switching to a “new” instance
    - The **vCPU** of the new instance is equal to that of the old instance *or* the application’s observed vCPU is less than 80% of the vCPU capacity of the new instance.
    - The **memory** of the new instance is equal to that of the old instance *or* the application’s observed memory peak is less than 80% of the memory capacity of the new instance.
    - The **network** throughput of the new instance is equal to that of the old instance *or* the application’s network peak is less than the network capacity of the new instance.
    - Note: Maximum NetworkIn and NetworkOut values are measured in bytes-per-minute. Use the following formula to convert these metrics to megabits per second: *Maximum NetworkIn (or NetworkOut) x 8 (bytes to bits) /1024/1024/ 60 = Number of Mbps*
- If the ephemeral **storage** disk I/O is less than 3,000, you can use Amazon Elastic Block Store (Amazon EBS) storage. If not, use instance families that have ephemeral storage

# Rightsizing Considerations (Cross Family)

- **Virtualization type** – The instances must have the same Linux AMI virtualization type (PV AMI versus HVM) and platform (EC2-Classic versus EC2-VPC). For more information, see Linux AMI Virtualization Types.
- **Network** – Some instances are not supported in EC2-Classic and must be launched in a virtual private cloud (VPC). For more information, see Instance Types Available Only in a VPC.
- **Platform** – If your current instance type supports 32-bit AMIs, make sure to select a new instance type that also supports 32-bit AMIs (not all EC2 instance types do). To check the platform of your instance, go to the Instances screen in the Amazon EC2 console and choose **Show/Hide Columns, Architecture**.
- **Ephemeral Storage** - When you resize an EC2 instance, the resized instance usually has the same number of instance store volumes that you specified when you launched the original instance. You cannot attach instance store volumes to an instance after you've launched it, so if you want to add instance store volumes, you will need to migrate to a new instance type that contains the higher number of volumes.

***May need to rebuild your AMI/instance from scratch***