
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

ANT325

How Disney+ processes petabyte-scale data on Amazon EMR Serverless

Himateja Madala

Senior Engineering Manager
Disney Streaming

Radhika Ravirala

Principal Product Manager
Amazon EMR
AWS

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Overview

What is Amazon EMR Serverless

Common usage patterns

Frequently asked questions

Cost optimization

How Disney streaming processed petabyte-scale clickstream data on Amazon EMR Serverless

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Serverless

RUN BIG DATA APPLICATIONS USING OPEN-SOURCE FRAMEWORKS WITHOUT MANAGING CLUSTERS AND SERVERS



Simple to use

No servers to manage. Amazon EMR Serverless provisions, configures, and dynamically scales the compute and memory resources needed at each stage of your data processing application



Fast

Performance optimized runtime that is compatible with and over 2X faster than standard open source



Cost effective

Pay only for the compute time and resources that you use



Comprehensive

Integration with Amazon EMR Studio and familiar open-source tools to easily develop, visualize, and debug applications

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Core concepts

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Serverless key components

1

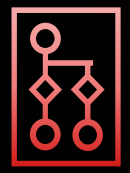
Application



- **Application = OSS framework & version**

2

Jobs



- Run on applications
- Run multiple jobs on an application
- Control authorization using per-job execution role

3

Workers



- Run your jobs
- Workers run the OSS framework you choose
- You can change the size of workers to control performance

4

Pre-initialized workers

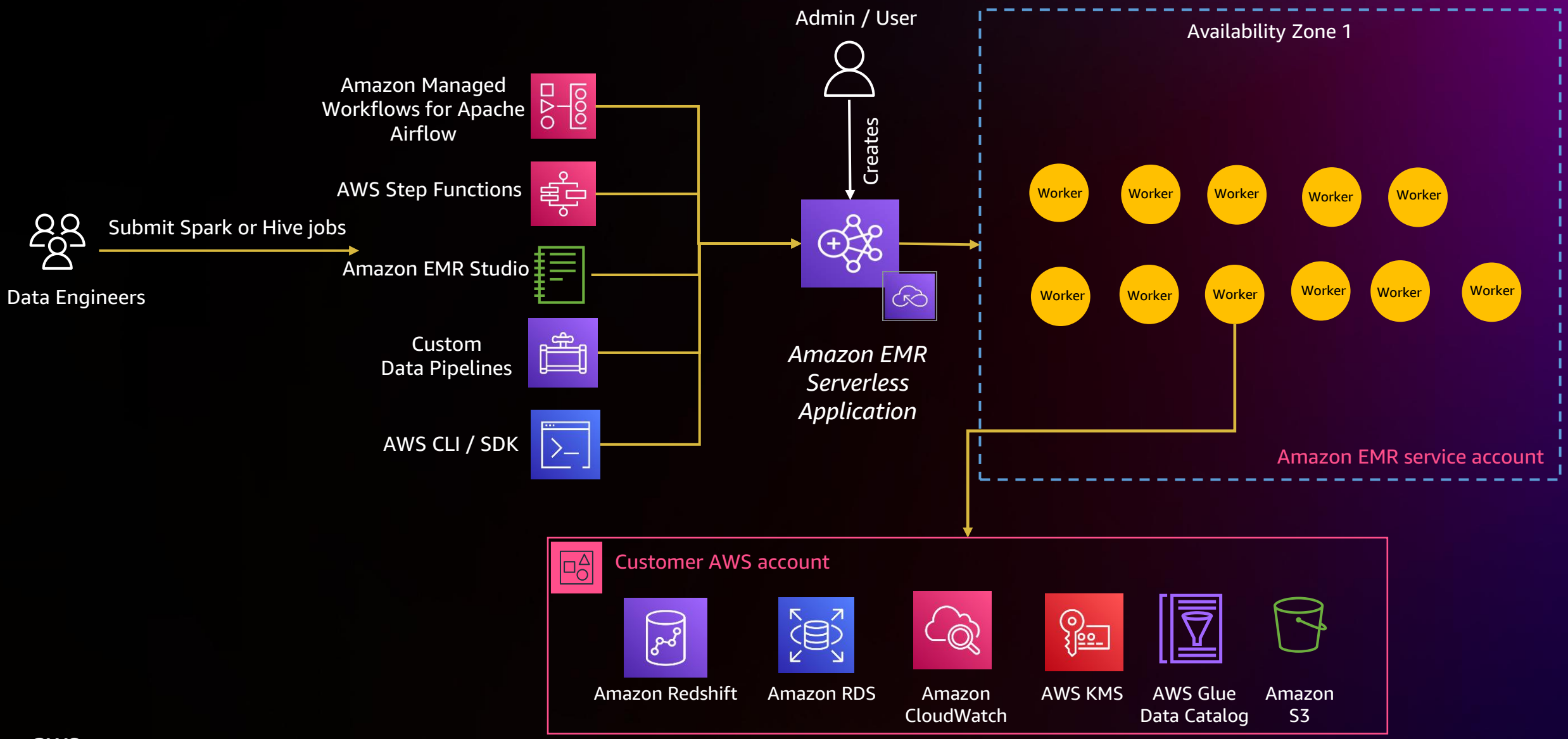


- Optional feature to pre-initialize workers
- Jobs start immediately
- Helps you maintain a serverless warm pool
- Serverless warm pools can auto-terminate with idle timeouts

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How it works



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

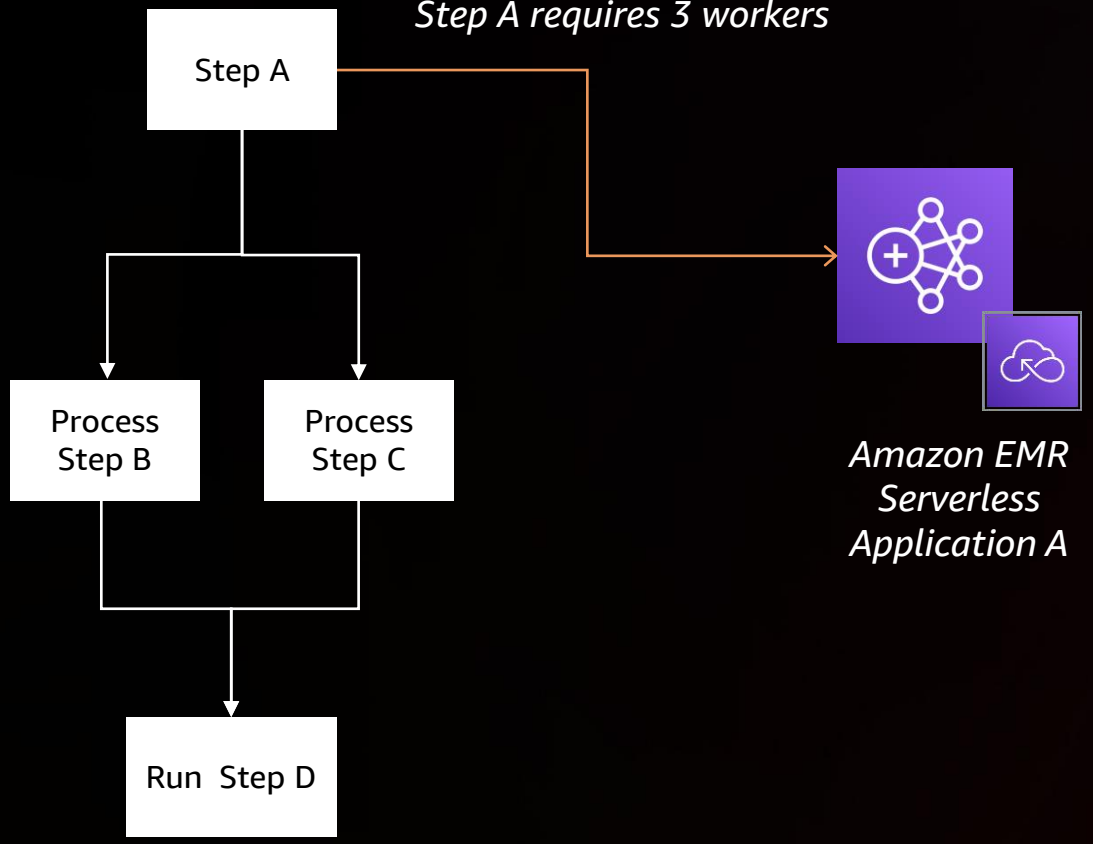
Common usage patterns

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pattern 1: Data pipelines

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pipelines are simple to run



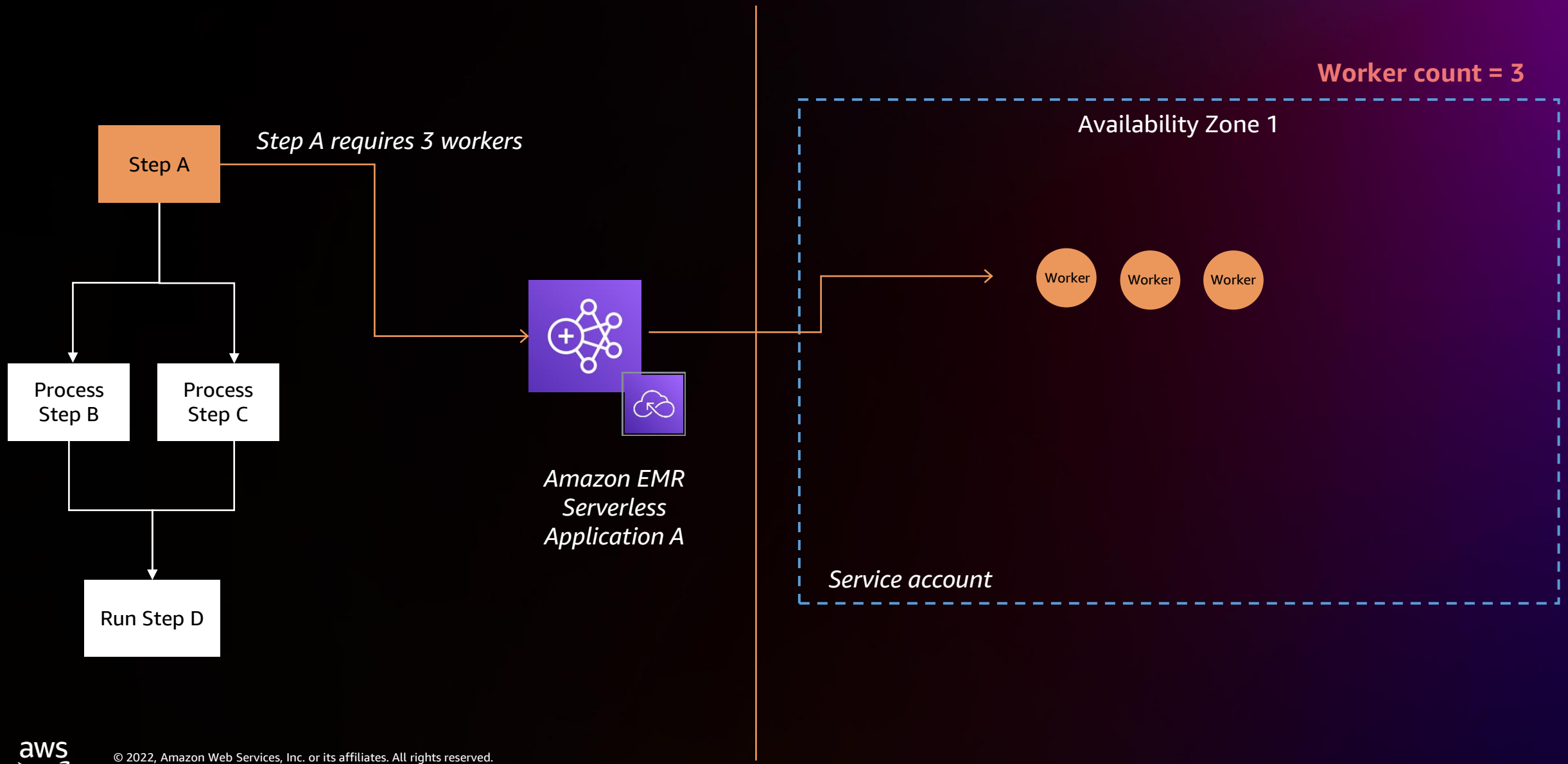
Worker count = 0

Availability Zone 1

Service account

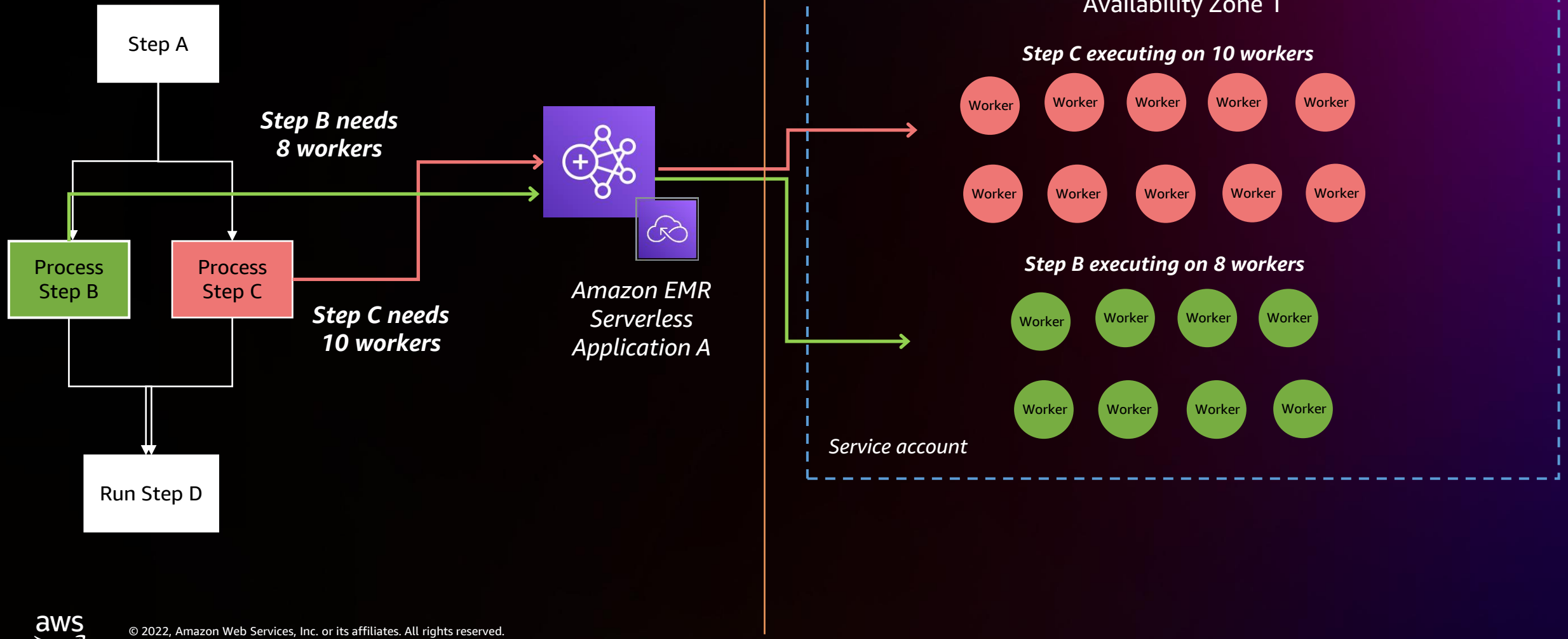
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pipelines are simple to run



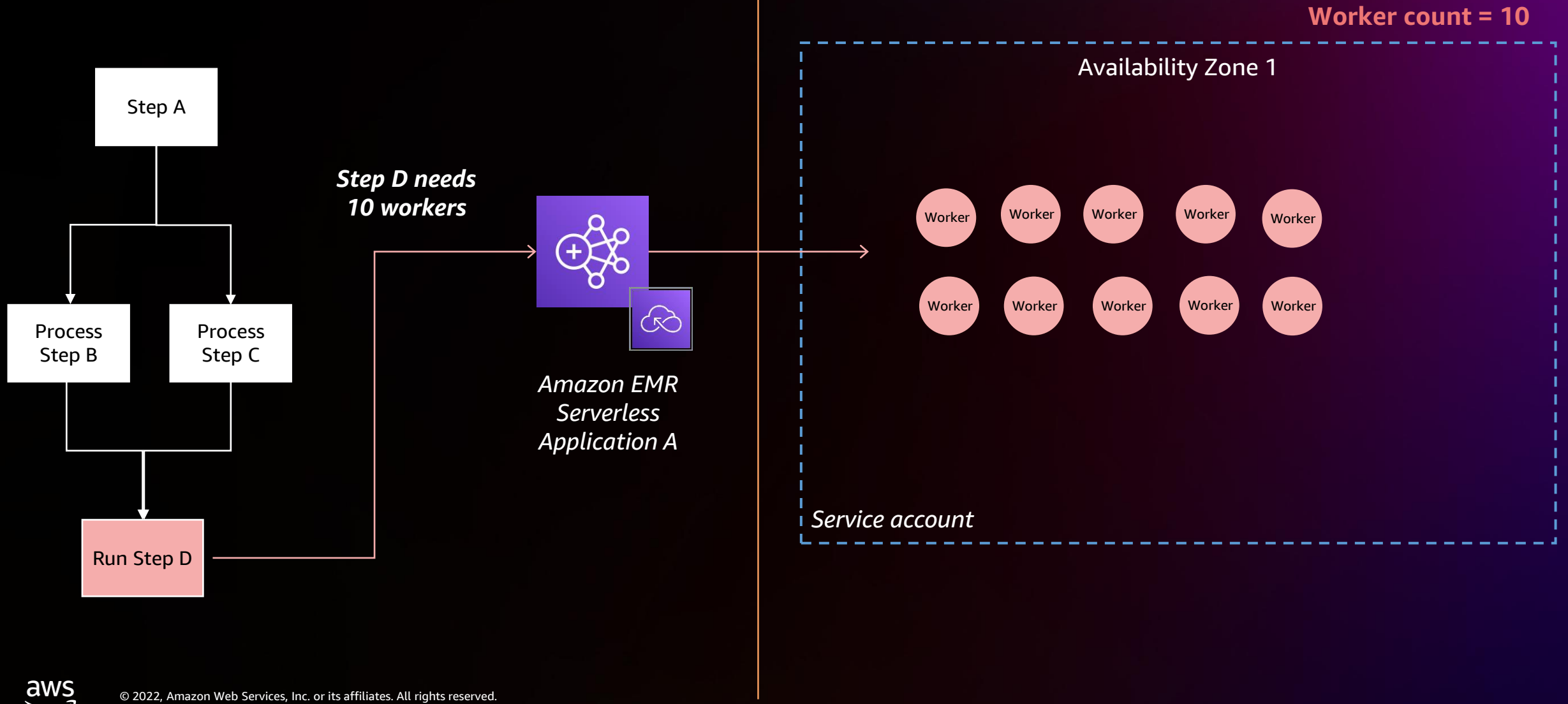
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Automatically scale when running stages in pipeline



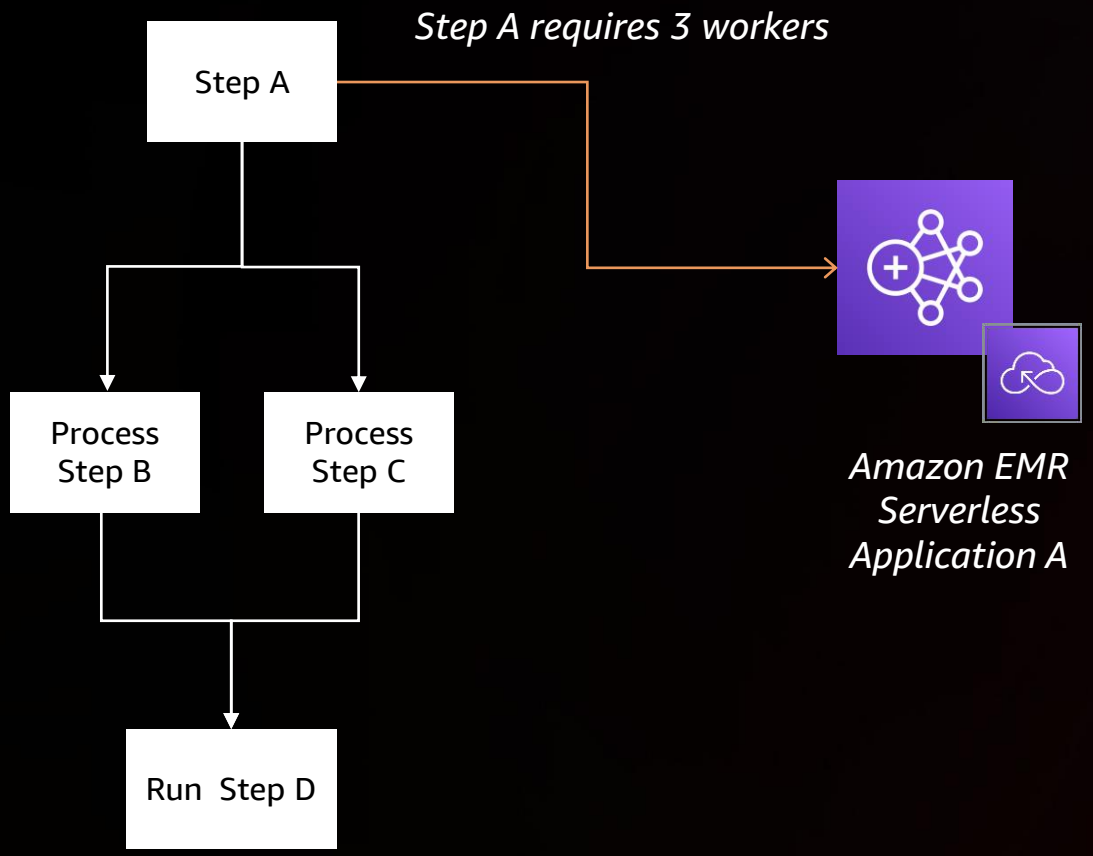
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workers are assigned to each job when needed



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workers are decommissioned when pipeline finishes



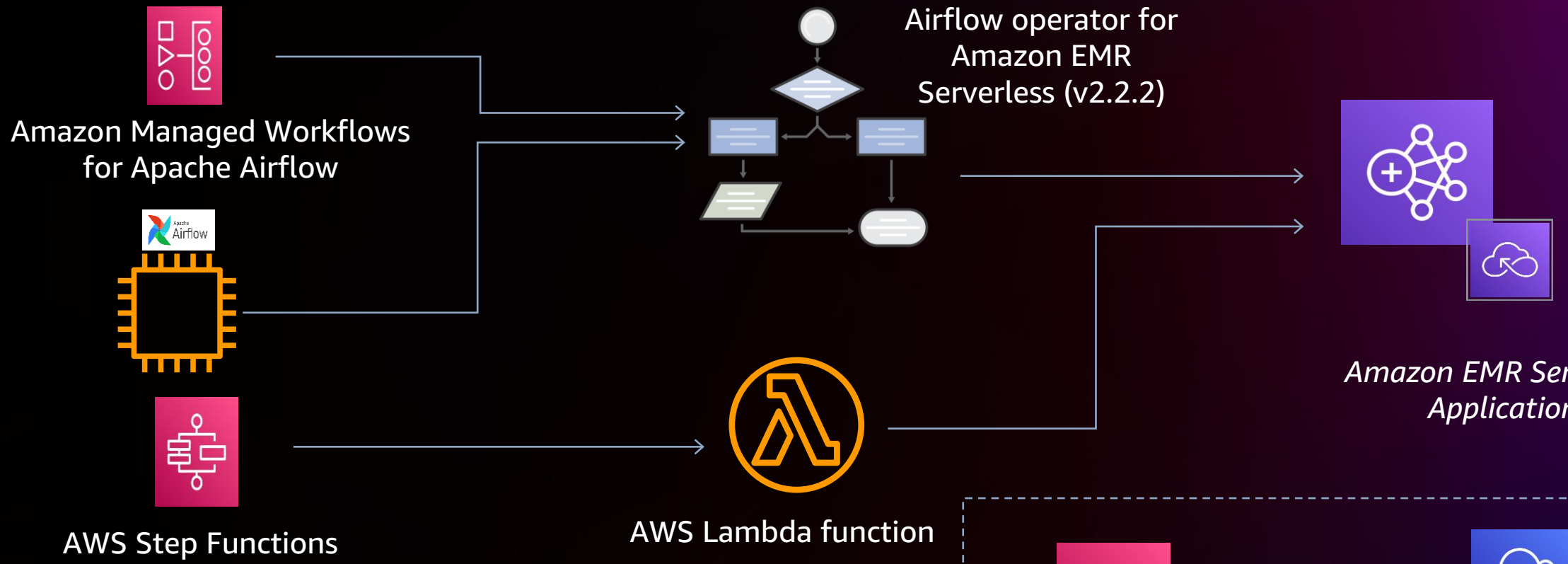
Worker count = 0

Availability Zone 1

Service account

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Orchestrate your pipelines with Apache Airflow and AWS Step Functions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



1. Amazon EMR Serverless launches new workers on demand on job submission
2. Workers take up to a minute to launch
3. Spark and Hive engines request resources dynamically (Dynamic Resource Allocation) for each stage of your job

aws

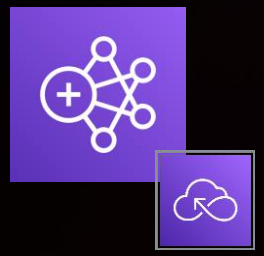
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pattern 2: Time-sensitive applications

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pre-initialized capacity is a serverless warm pool for applications

Amazon EMR Serverless Application A



"InitialCapacity = 10"

State = Stopped

Worker count = 0

Availability Zone 1

Worker Worker

Worker Worker

Worker Worker

Worker Worker

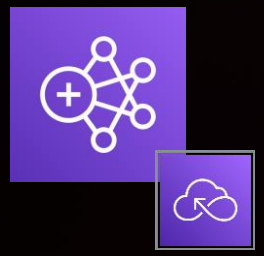
Worker Worker

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workers start when applications start

Amazon EMR Serverless Application A



"InitialCapacity = 10"

Workers are ready to receive workloads

State = Ready

Worker count = 10

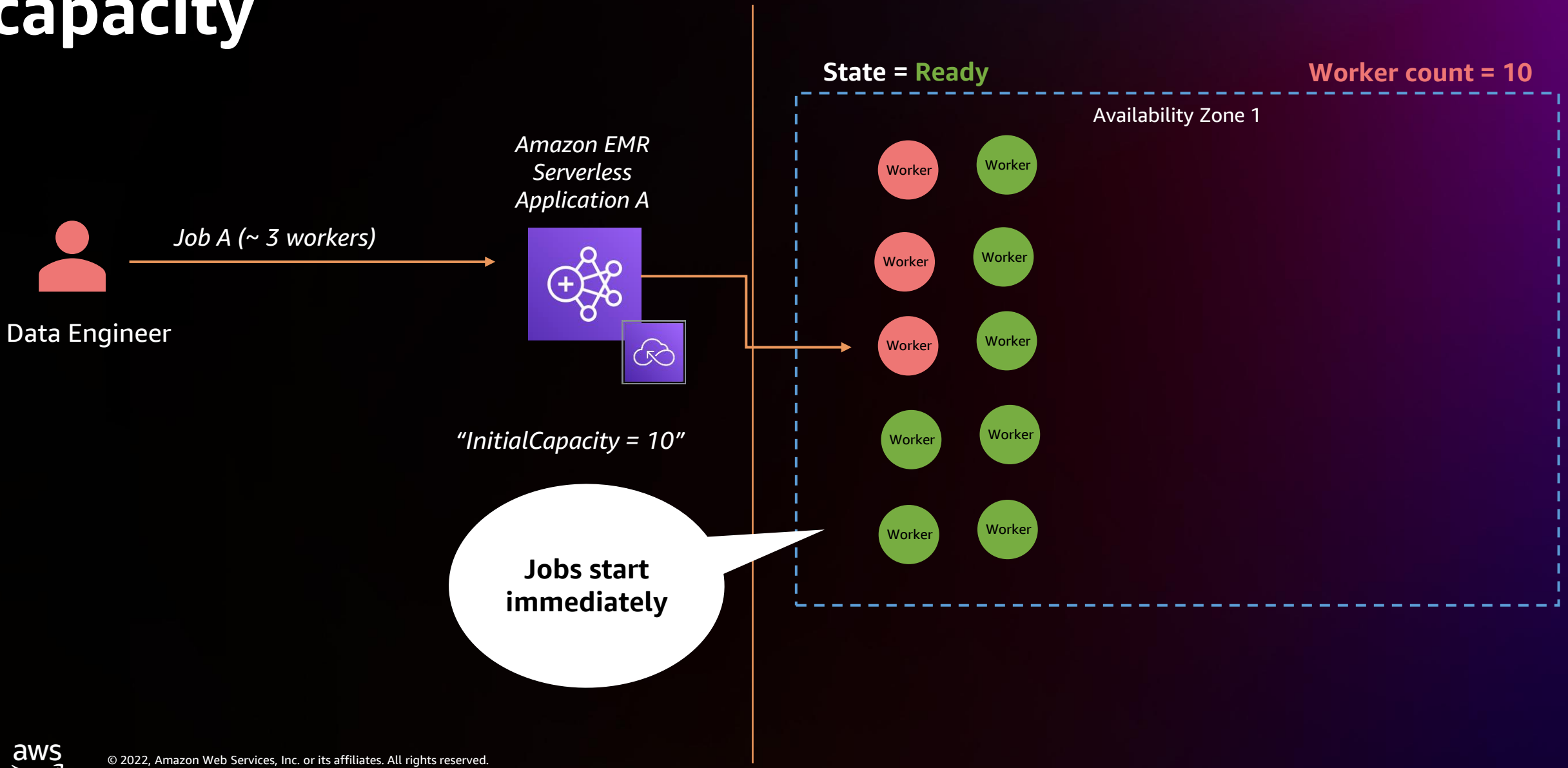
Availability Zone 1

Worker Worker
Worker Worker
Worker Worker
Worker Worker
Worker Worker

aws

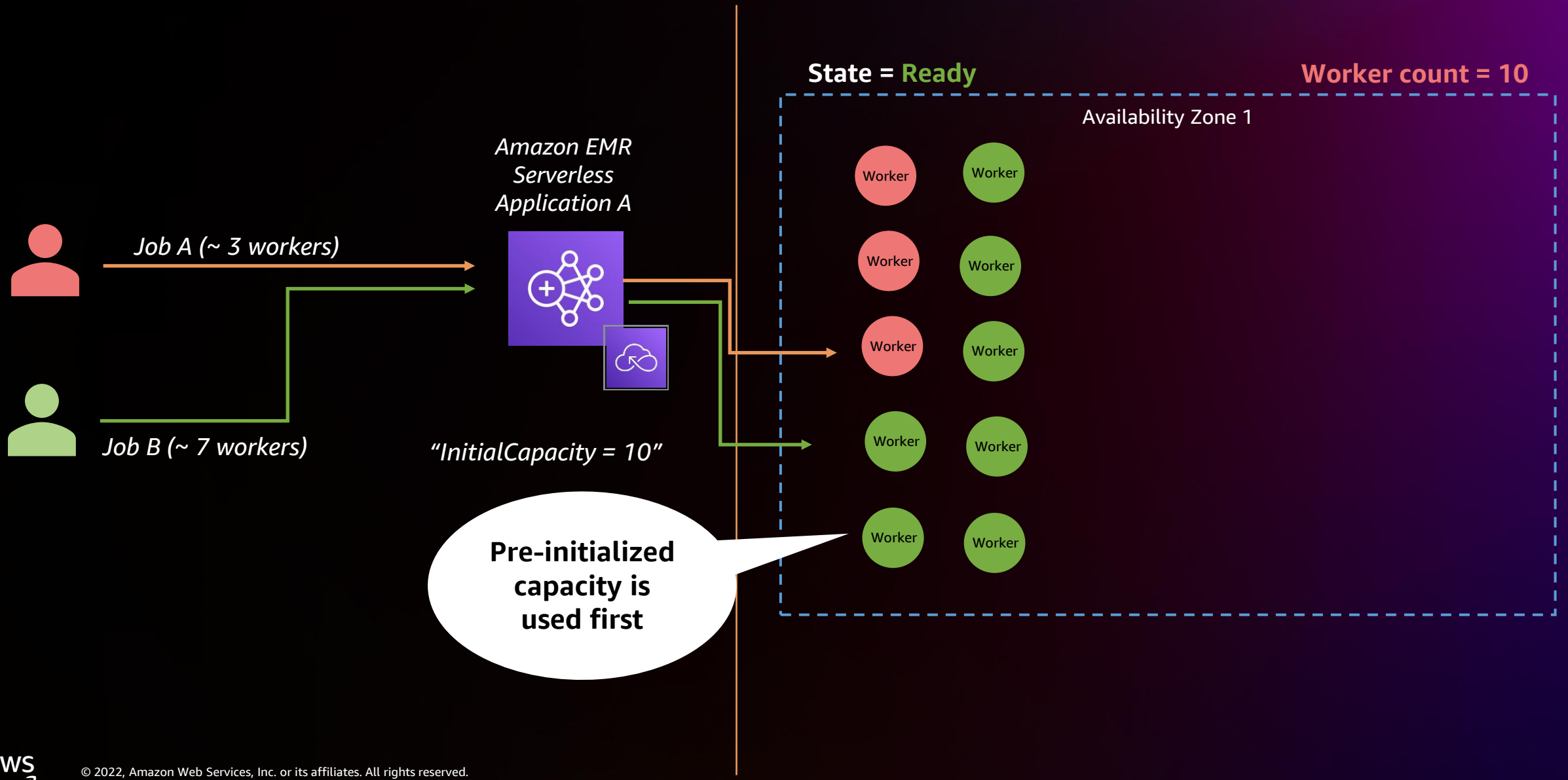
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Jobs start instantaneously with pre-initialized capacity



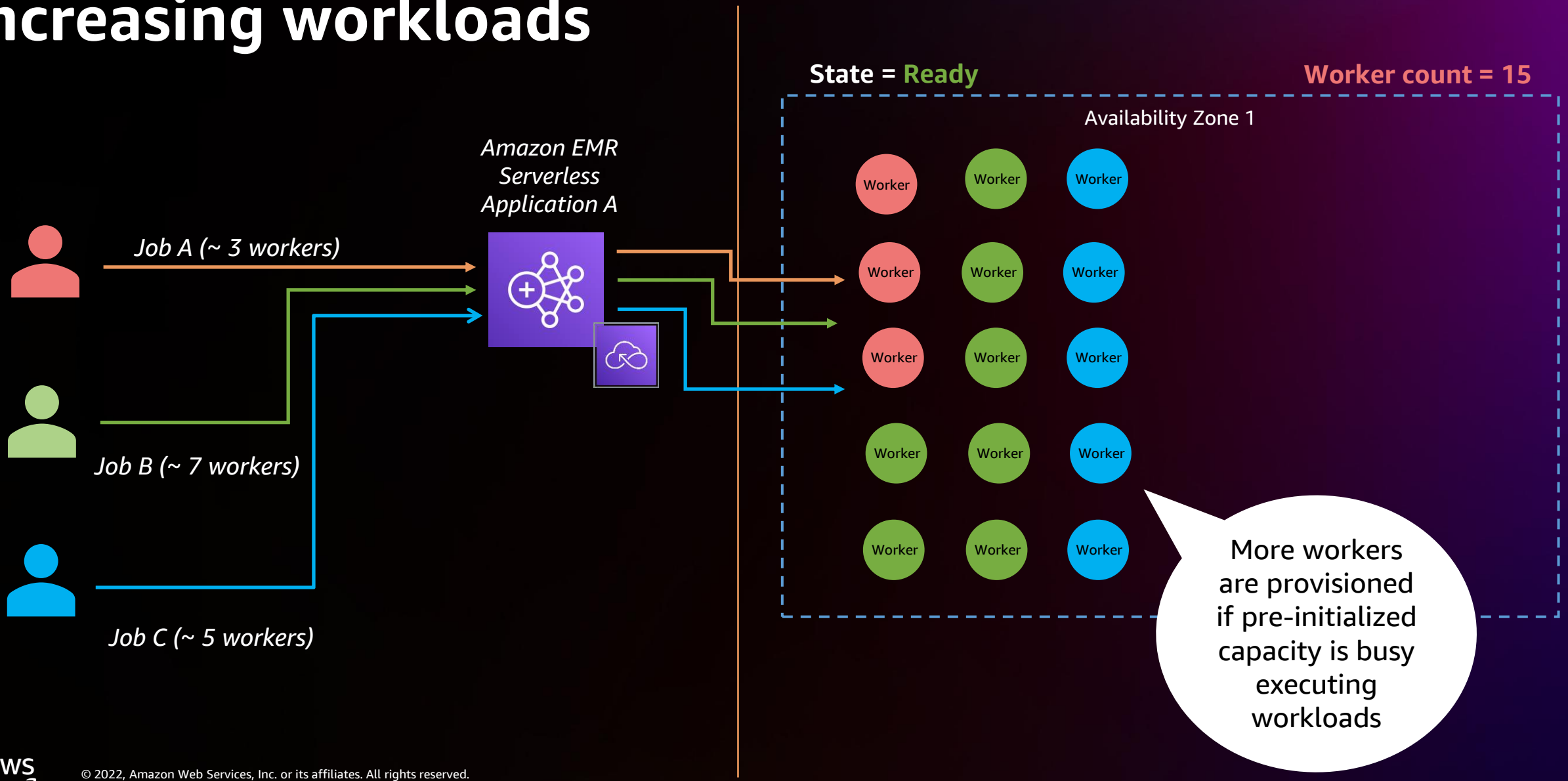
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Multiple jobs can start instantaneously



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Automatically request more capacity with increasing workloads



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

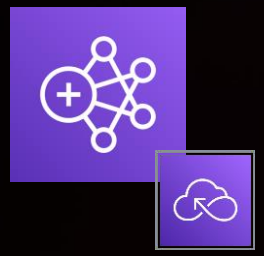
Pre-initialized capacity is maintained after all jobs complete

Job A (Finished)

Job B (Finished)

Job C (Finished)

Amazon EMR Serverless Application A



"InitialCapacity = 10"

When jobs finish the application excess workers are decommissioned

State = Ready

Worker count = 10

Availability Zone 1

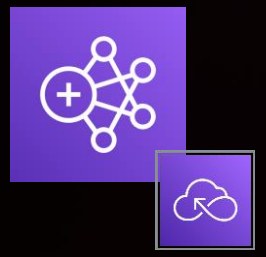
Worker Worker
Worker Worker
Worker Worker
Worker Worker
Worker Worker

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Stopping application decommissions capacity

Amazon EMR Serverless Application A



"InitialCapacity = 10"

All workers are stopped when applications stop

State = Stopped

Worker count = 0

Availability Zone 1

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



1. For time-sensitive workloads, configure pre-initialized capacity with your application launch
2. New drivers and executors are launched on-demand as needed by the job
3. Number of concurrent jobs you can run depends on many factors including but not limited to
   - Maximum capacity limits
   - Available IP address when using VPC

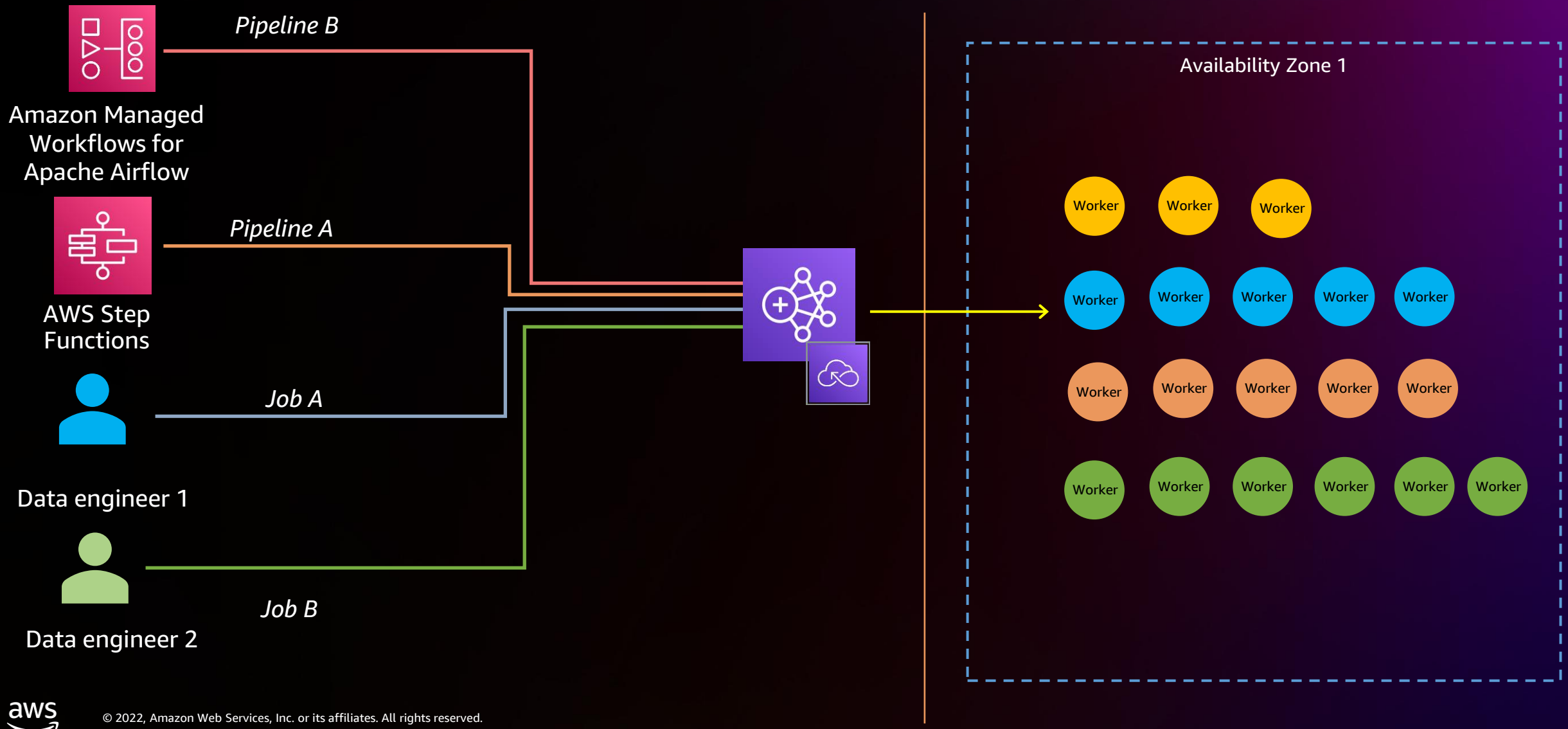
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pattern 3: Secure shared applications

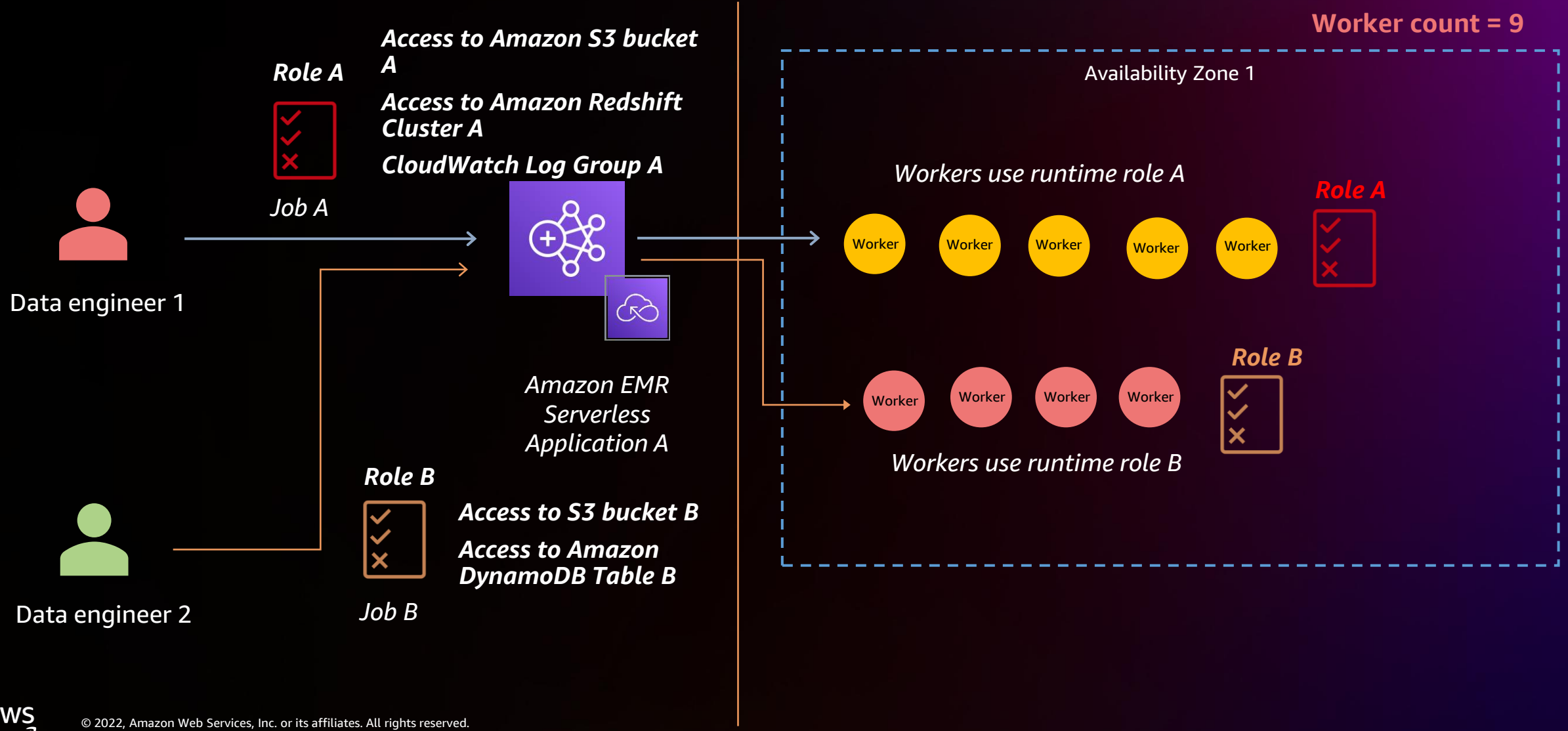
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Shared applications with Amazon EMR Serverless



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Per-job runtime role



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



1. Attach IAM permissions policies to the IAM user's role to allow the user to pass only approved roles

2. This allows administrators to control which users can pass specific job runtime roles to Amazon EMR Serverless jobs

aws

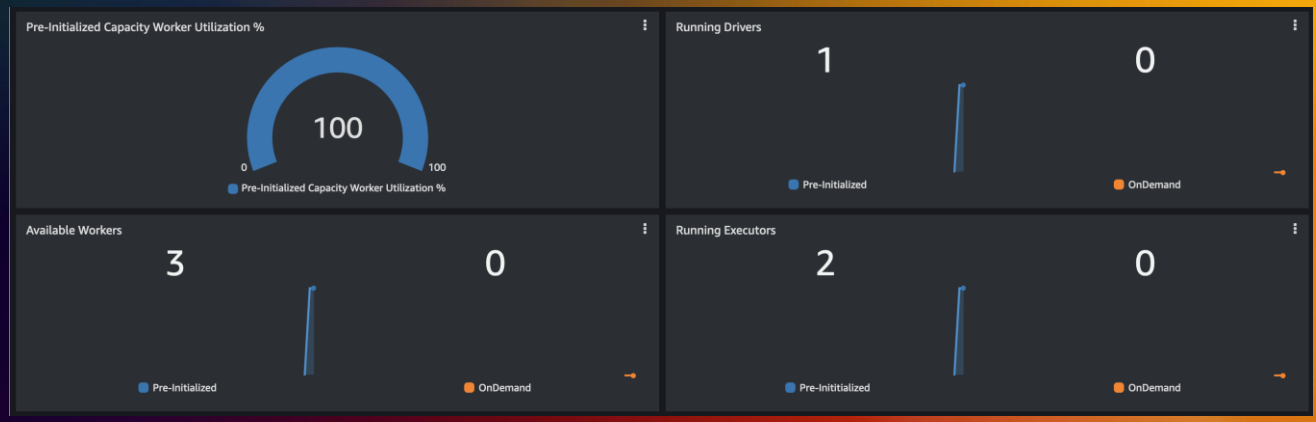
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Monitoring

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Monitor your jobs with CloudWatch metrics

- Access CloudWatch dashboards to view near real-time metrics and performance



- Application level – `RunningWorkerCount`, `CPUAllocated`, `MemoryAllocated`, `PendingCreationWorkerCount`, `IdleWorkerCount`

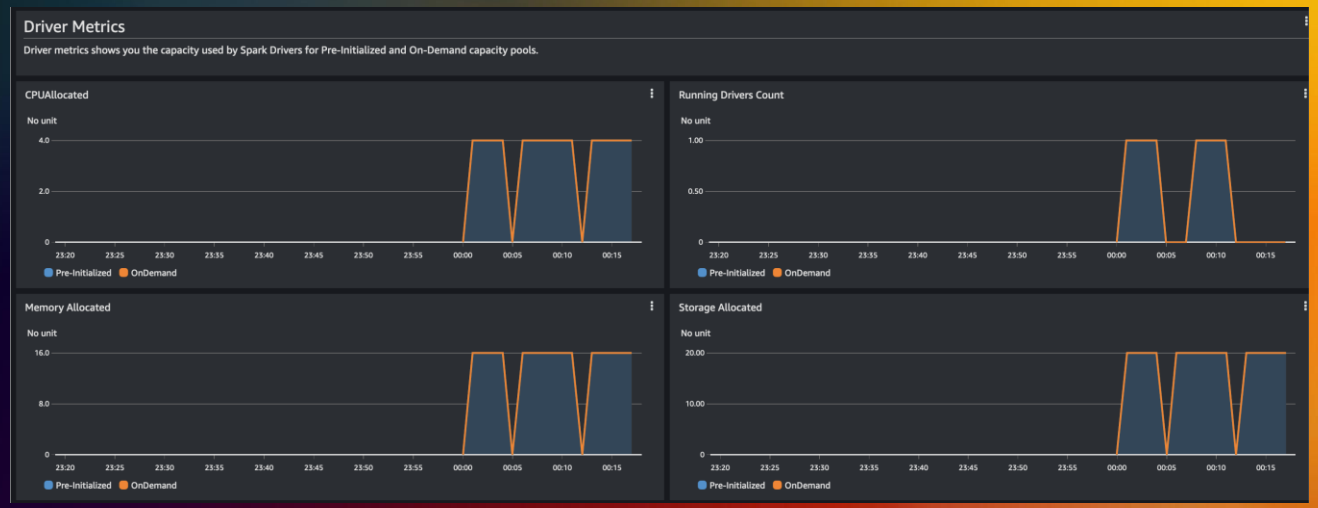


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Monitor your jobs with CloudWatch metrics

- Driver and Executor metrics provide a glimpse of capacity consumption of your jobs - *`Driver/Executor CPU/Memory allocated`*,

- Dimensions
  - Application
  - WorkerType
  - CapacityAllocationType

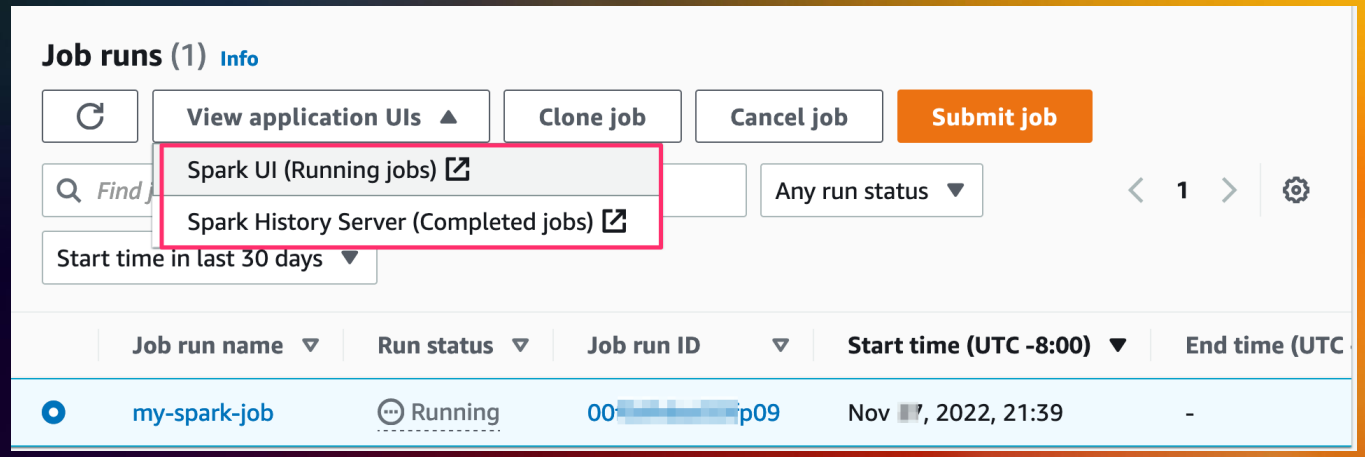


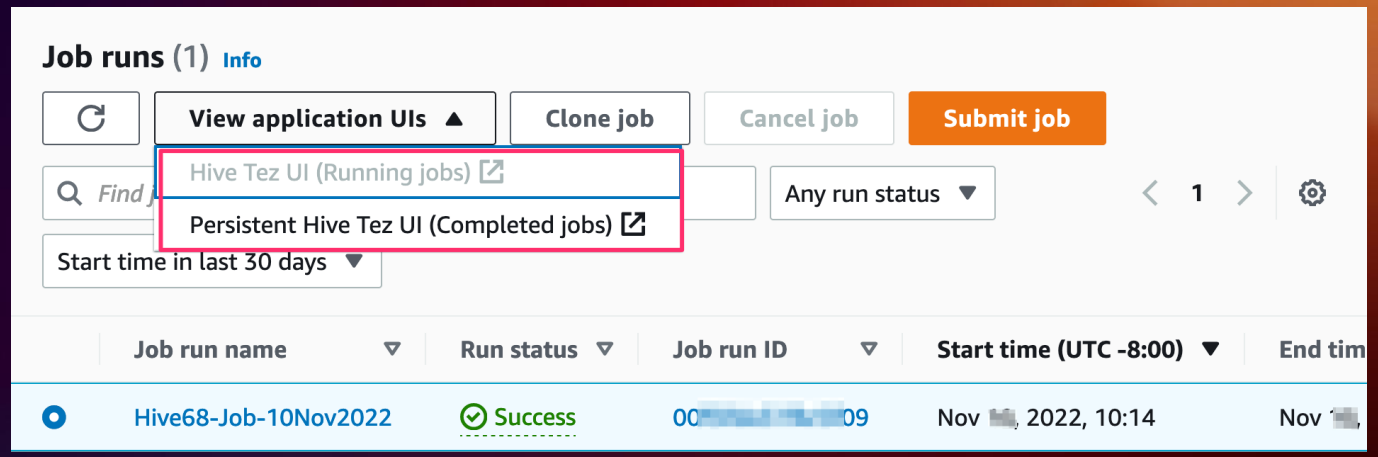


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Debugging is easy!

- Monitor engine-specific metrics using application UIs
- Live UI for running jobs
- Persistent UI for completed jobs
- `GetDashboardForJobRun` API generates URL to access your application UI for running jobs





© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



1. Metrics published at 1-minute interval in CloudWatch
2. Calibrate your future jobs based on capacity utilization
3. Use examples from git repo for building dashboards

   https://github.com/aws-samples/emr-serverless-samples/tree/main/cloudformation/emr-serverless-cloudwatch-dashboard

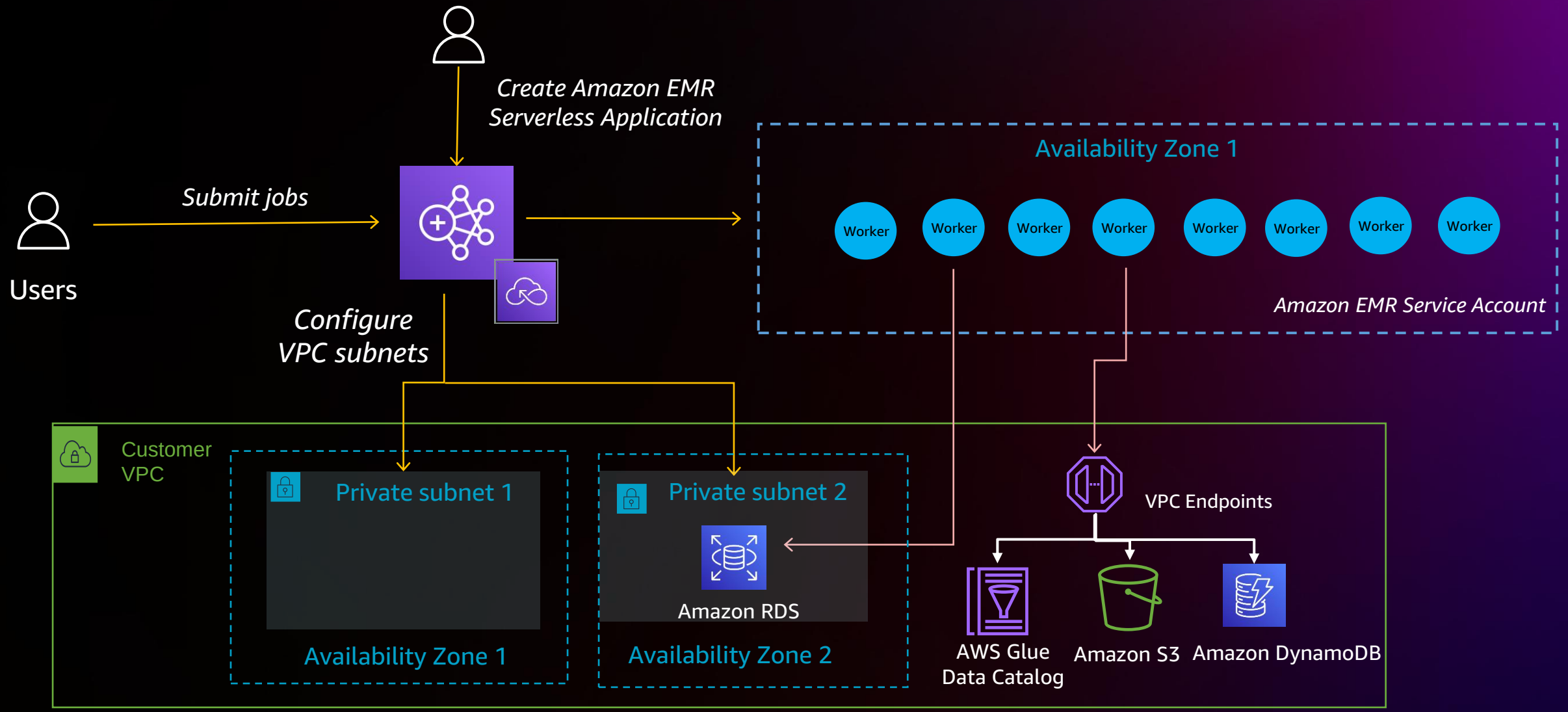
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Routing traffic

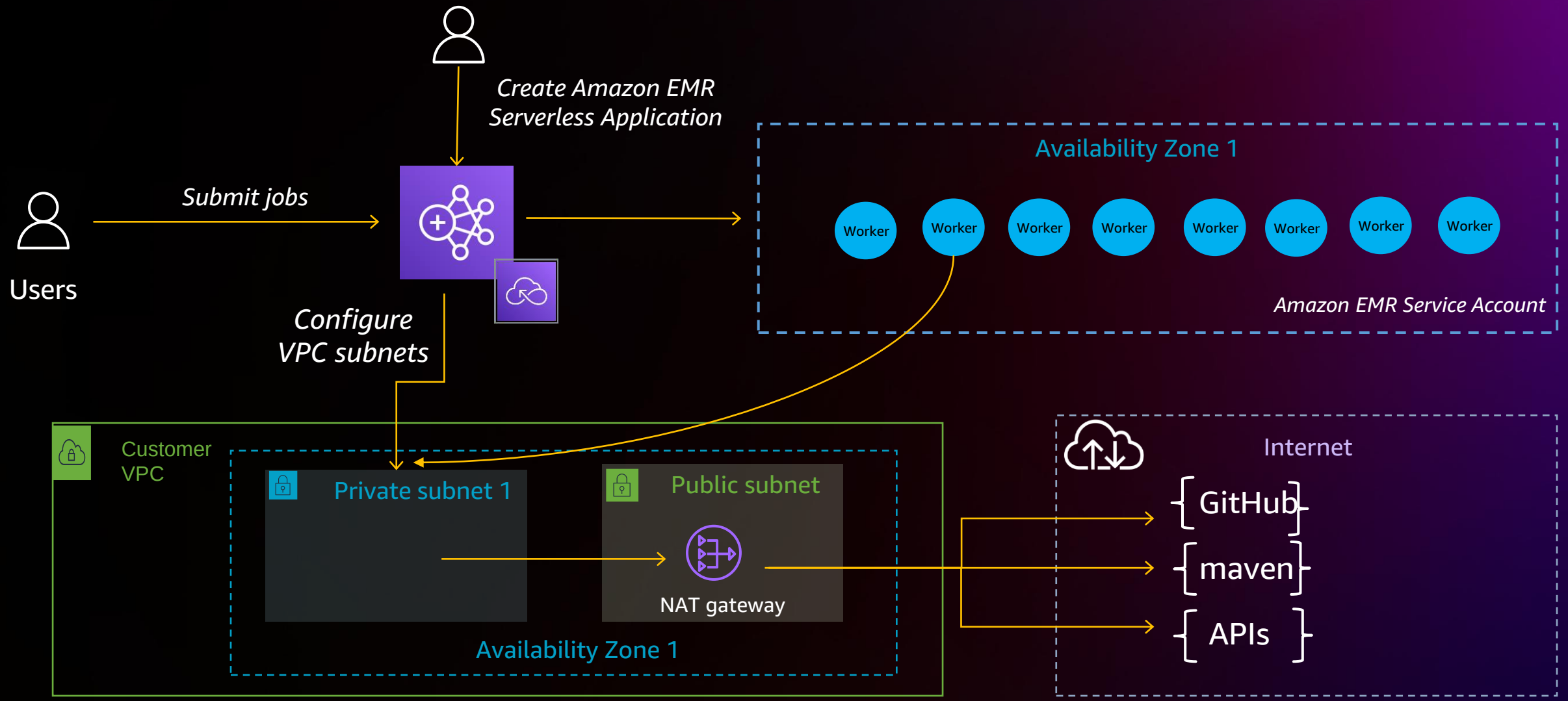
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Route traffic through your VPC



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Route traffic to access internet



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways

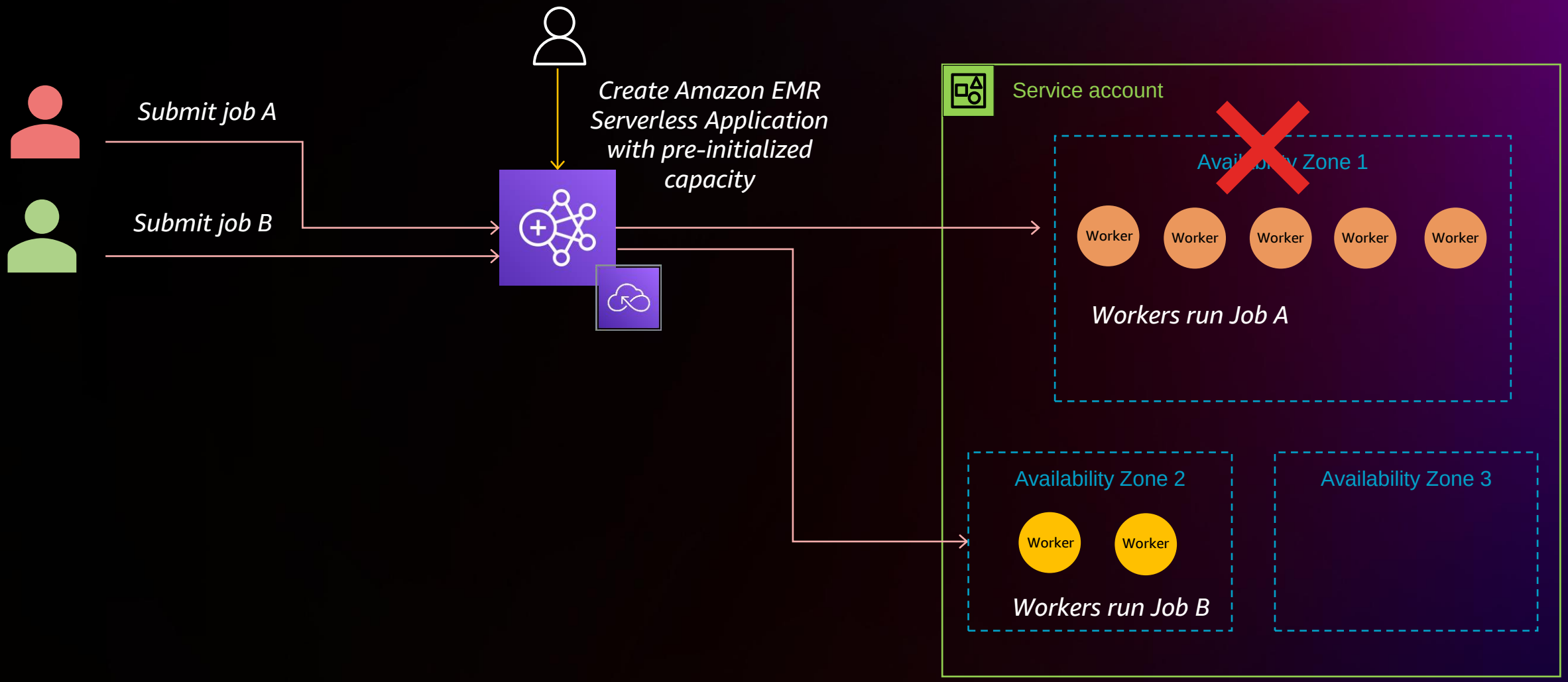


1. During application launch, specify the subnets in different AZs in your account
2. Applications are network isolated from other applications
3. Amazon EMR Serverless will set up ENIs to securely communicate with resources in your VPC
4. Pro tip: Make sure you have enough IP addresses in your subnet for the job

aws

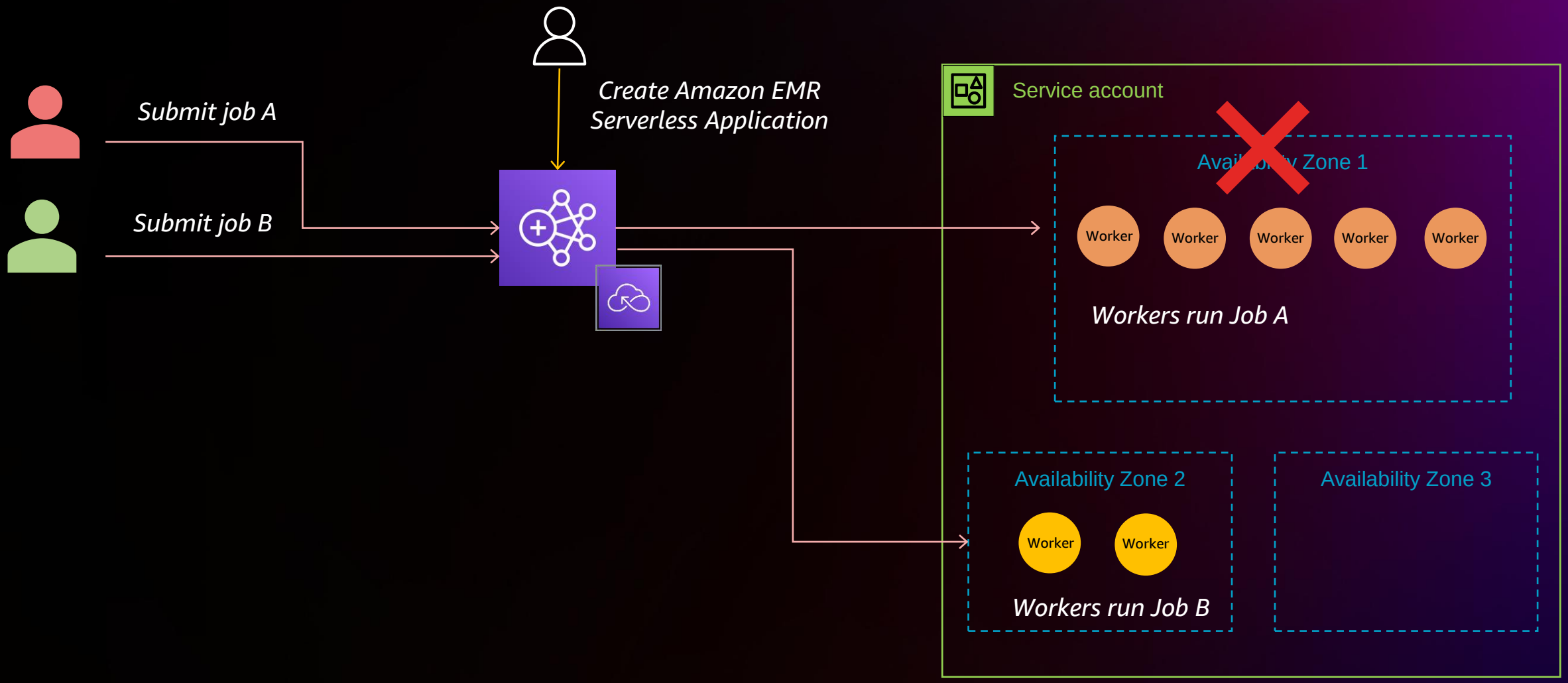
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AZ resilience with pre-initialized capacity



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AZ resilience with on-demand capacity



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



1. Amazon EMR Serverless selects a healthy AZ on job submission
2. Subsequent job launches have affinity to the selected AZ to minimize cross-AZ latency and cost
3. When an AZ failure is detected, subsequent jobs are launched in healthy AZs
4. When Amazon EMR Serverless application stops on idle timeout, it will auto-start on job submission in a healthy AZ

aws

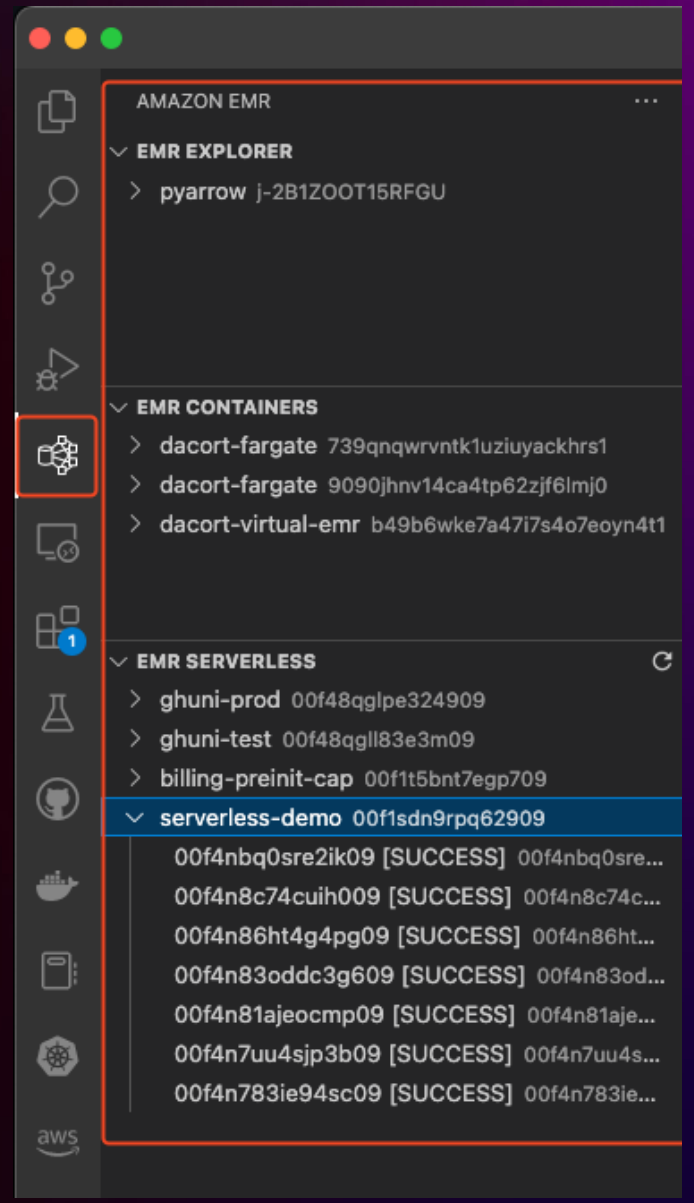
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Develop locally, build CI/CD pipelines

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

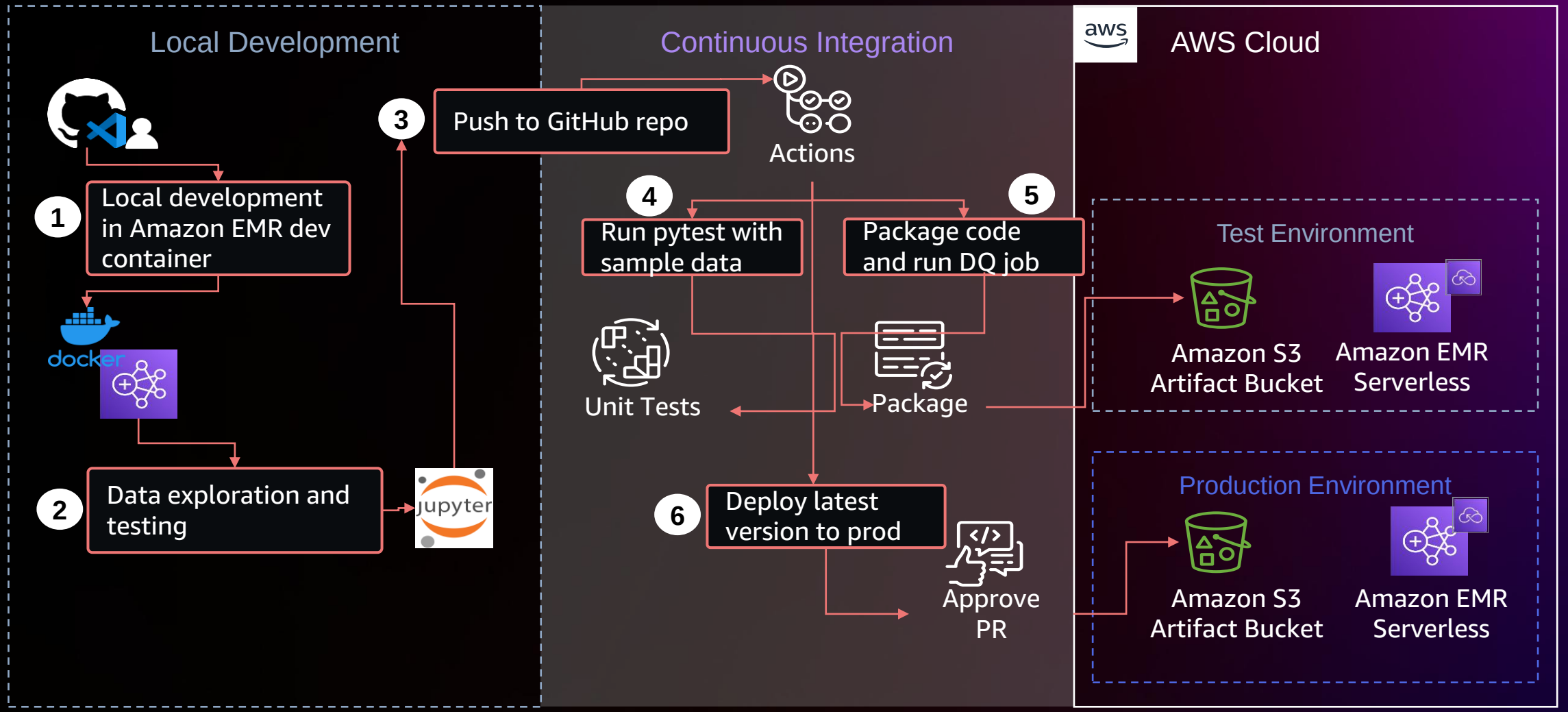
Use Visual Studio Code extension to develop Spark jobs locally

- Amazon EMR Toolkit for VS Code provides local Spark Environment on a container
- Container is based on Amazon EMR on Amazon EKS image for an specific Amazon EMR release
- Container can be used to develop Spark and PySpark code locally that is fully compatible with remote Amazon EMR Serverless environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Build CI/CD pipeline for serverless jobs using GitHub Actions

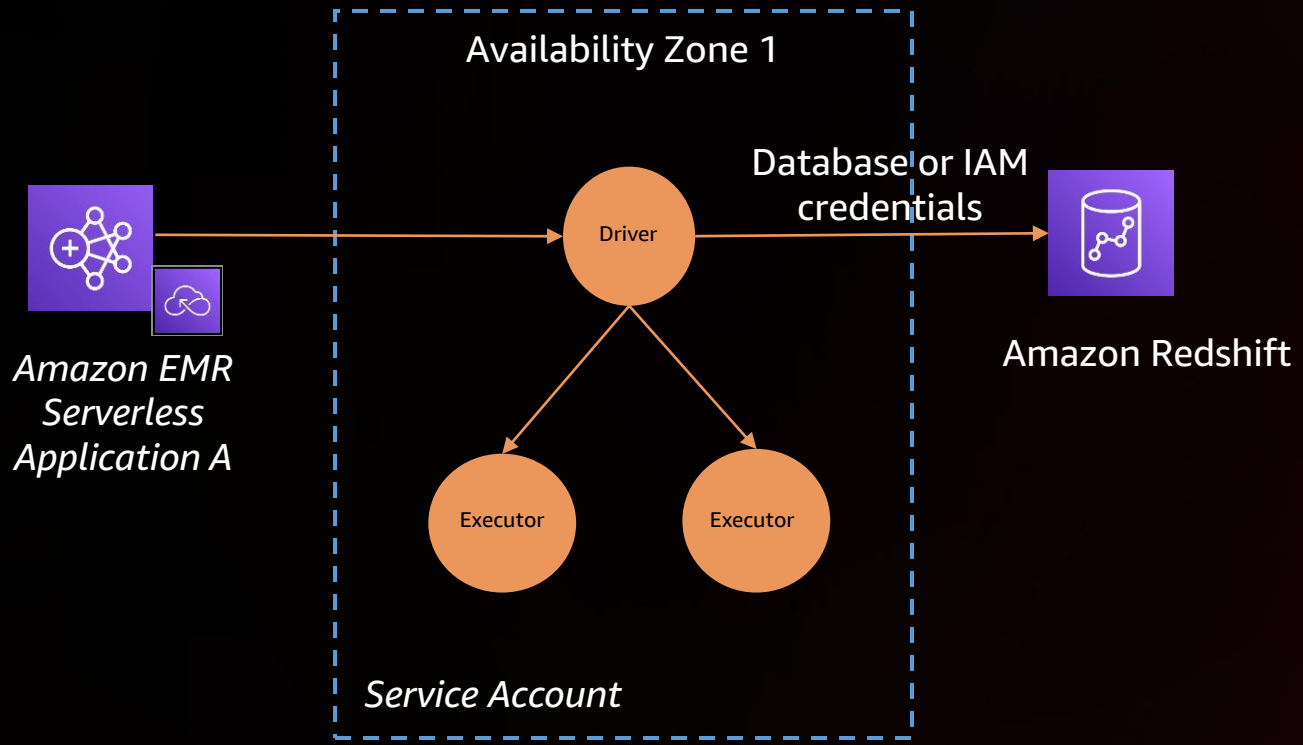


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Custom connectors

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

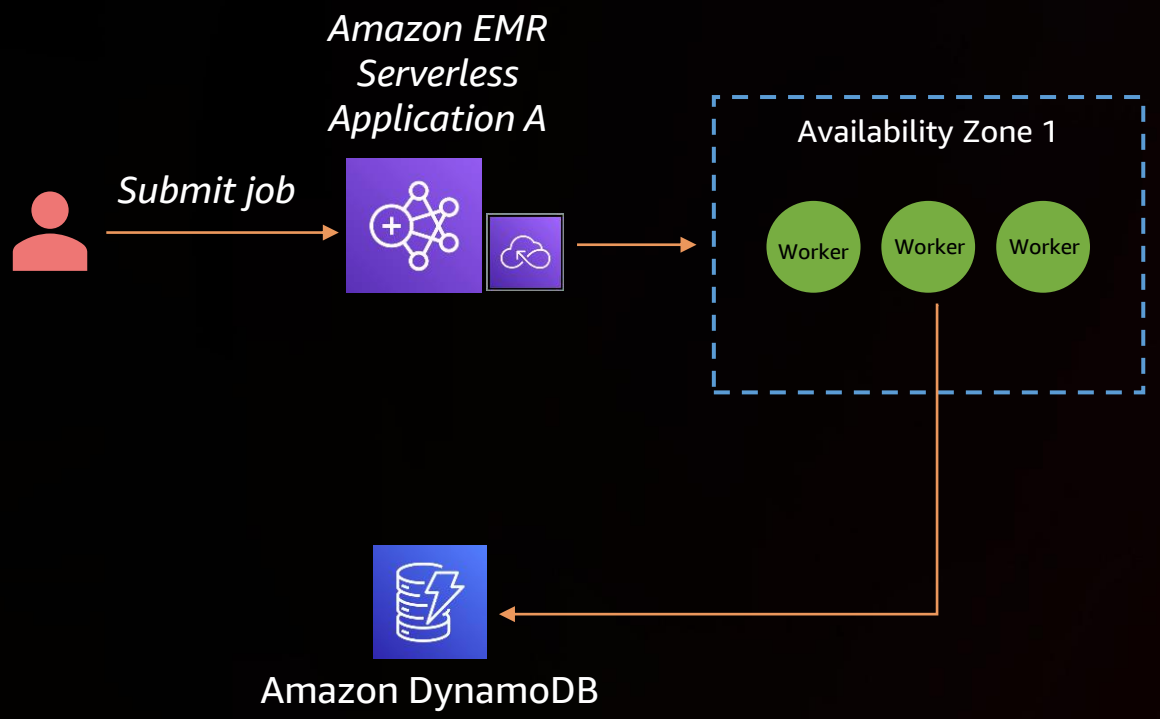
Simplify and speed up your Apache Spark applications accessing data on Amazon Redshift



- Connector and Amazon Redshift JDBC drivers are available out of the box in Amazon EMR Serverless 6.9
- Advanced pushdown optimizations for improved performance
- Use either DB credentials or IAM-based credentials for connecting to Redshift
- Connector uses Parquet by default for unloading data

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

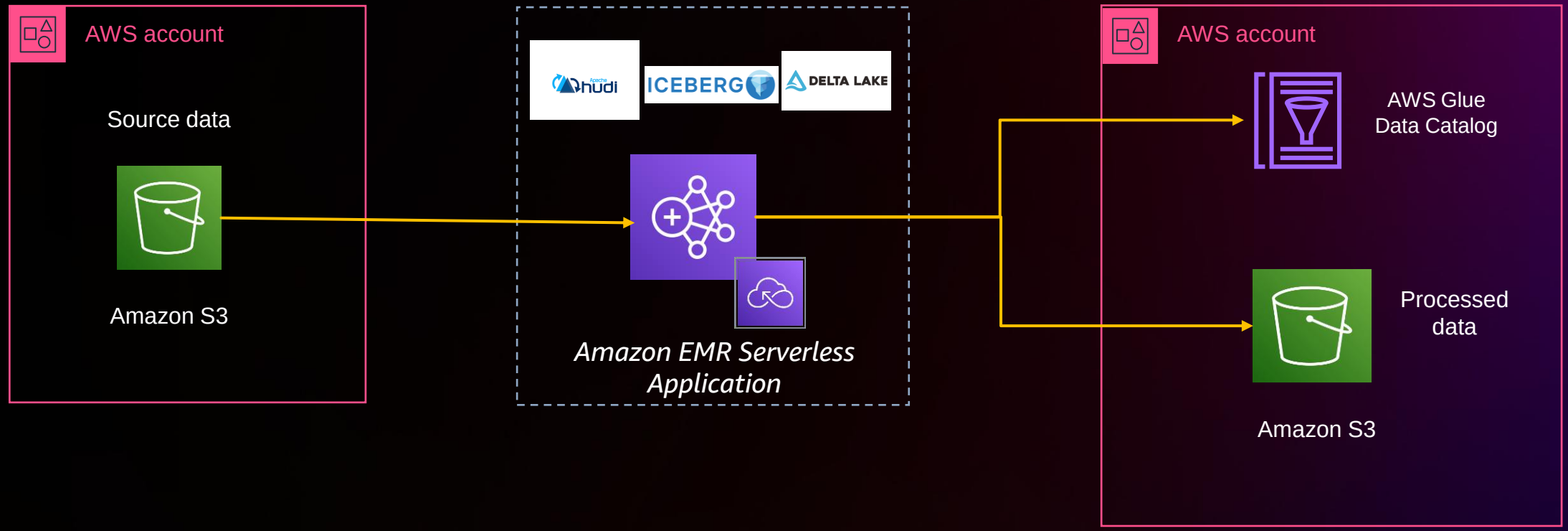
Export, import, query, and join tables in DynamoDB



- Access data in Amazon DynamoDB using Amazon EMR Serverless Spark or Hive applications
- Process data directly in DynamoDB, or join data in DynamoDB with data in Amazon S3, Amazon RDS, or other storage layers that can be accessed by Amazon EMR Serverless
- Access Amazon DynamoDB tables cross account and cross-Region

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Build transactional data lakes with open-source table formats: Apache Hudi, Iceberg, and Delta Lake



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Comparing Amazon EMR deployment options

Feature	Amazon EMR on EC2	Amazon EMR on EKS	Amazon EMR Serverless
Multi-AZ Availability	No (clusters run in a single AZ)	Yes (with multi-AZ EKS clusters)	Yes (Automated Job Redirection)
OSS Frameworks	Spark, Hive, Presto, Trino, Flink	Spark	Spark, Hive
Ability to choose OSS version	Yes	Yes	Yes
Automatic Resource Scaling	Yes	Yes	Yes
Ability to choose Instance Type	Yes	Optional (use EC2 instances or AWS Fargate)	No
Ability to use EC2 Spot	Yes	Yes	No
Pricing	By instance type used	By VPCU and Memory used	By VCPU and Memory used
Ability to allocate costs	Per Cluster	Per Application	Per Application or Per Job

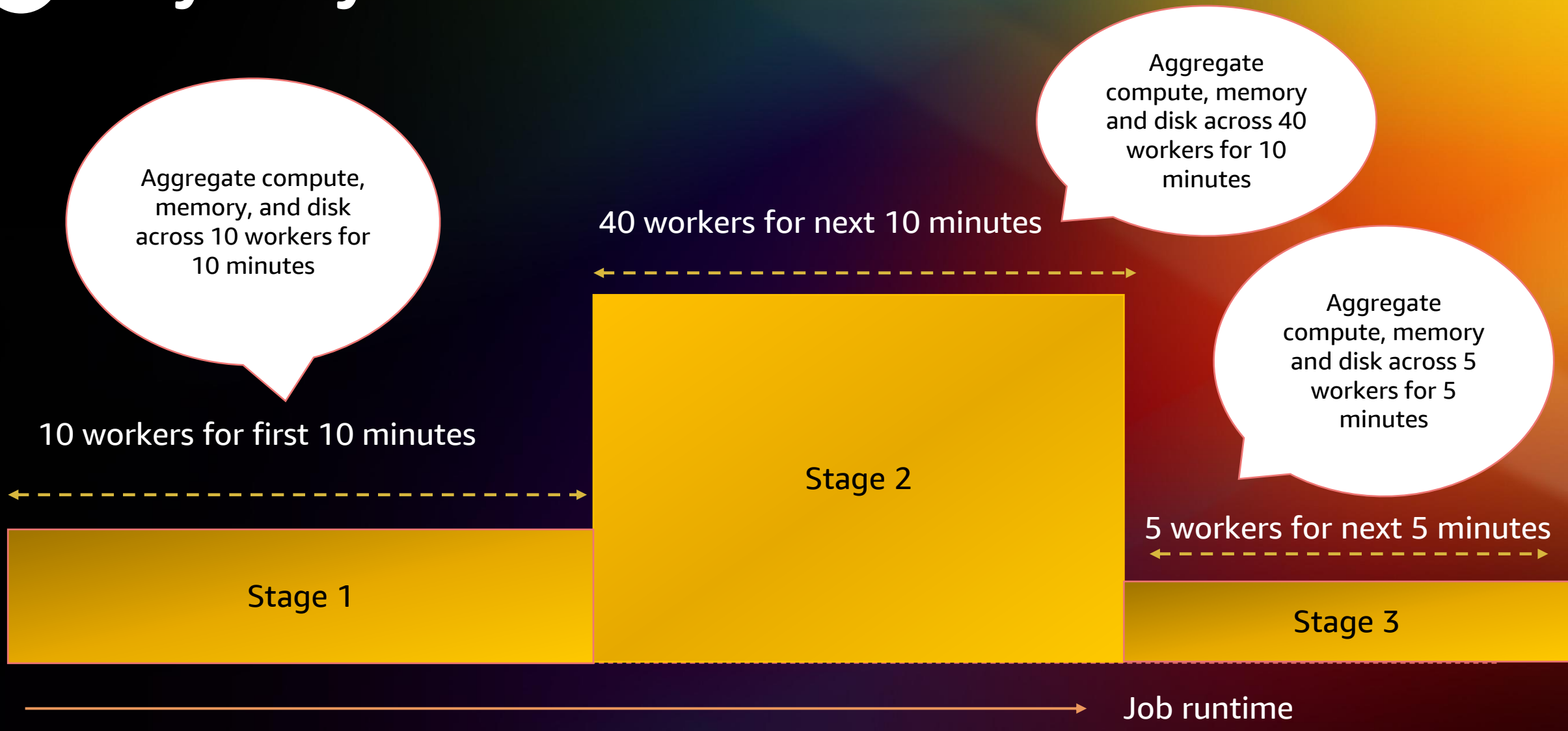
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Cost optimization

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1 Pay only for resources used



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2 Fine-grained scaling saves cost



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3 Application idle-timeouts

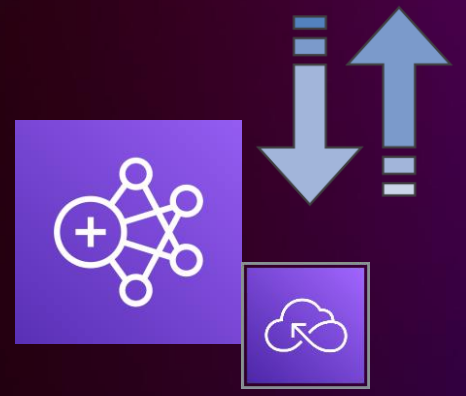
```
aws emr-serverless create-application
--name ”my-spark-app"
--type ”SPARK"
--auto-stop-config '{
  "enabled": true,
  "idleTimeout": 10
  }'
--release-label "emr-6.9.0"
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

4 Application and account-level limits

```
aws emr-serverless create-application
  --type ”SPARK"
  --name my_application_name
  --release-label emr-6.9.0
  --maximum-capacity '{
    "cpu": "400vCPU",
    "memory": "1024GB"
  }'
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5 Up to 35% better price-performance with Graviton2-based workers

- 15% improved performance
- Launch Spark or Hive applications using Graviton 2 (Arm64)-based architecture through the Studio Console or the AWS CLI
- Simply specify the architecture while creating the application

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Estimating cost for Amazon EMR on EC2

1. Amazon EC2 cost for running cluster instances 

+

2. Amazon EBS cost for attached volumes 

+

3. Amazon EMR cost for instances 

- Amazon EC2 cost depends on EC2 purchase option: On Demand, Spot, 1 year or 3 year Reserved Instances, Savings Plans
- Compute cost of 1, 2, 3 from the time the cluster is started until the time cluster is terminated
- Set up tags for Amazon EMR Clusters and use Cost Explorer to get the cost for each cluster

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Estimating cost for Amazon EMR Serverless

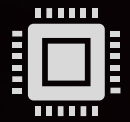
1. Total vCore-Seconds used by the application

+

2. Total memory MB-seconds used by the application

+

3. Total storage GB-seconds used by application







- Aggregate resources consumed by each application that ran on your Amazon EMR cluster
- Start from the time each container in the application is started to the time the container is terminated
- Alternatively, you can point your Spark event logs to the Amazon EMR Serverless estimator script on GitHub repository

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways



Amazon EMR Serverless improves cost-effectiveness with

1. No operational overhead
2. Efficient worker scaling under a minute
3. Performance-optimized runtime
4. Graviton2-based workers
5. Guardrails to control costs

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Recap: Serverless Spark and Hive on AWS



Run latest open-source Spark and Hive



Performance optimized runtime 2x faster



One-click debugging and detailed metrics



Build locally and deploy on cloud using VS Code plugin



Per-job authorization role



AZ resilience



Build once, run on any deployment framework



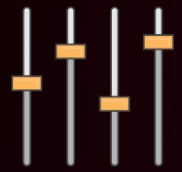
Airflow and Step Functions for orchestrating pipelines



Support for HUDI, Iceberg and Delta Lake



Provisioned mode for SLA sensitive workloads



Set cost-constraints by setting application limits



Multi-architecture support (X86 and Graviton2)

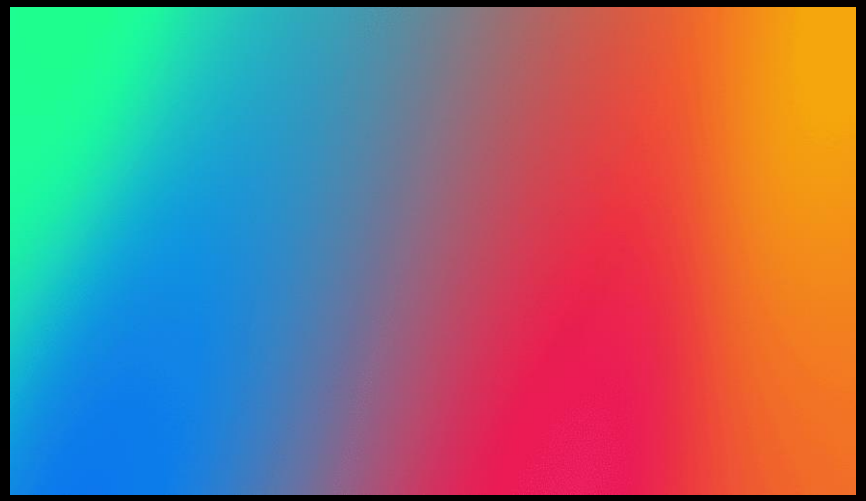
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How Disney+ processes petabyte-scale data with Amazon EMR Serverless

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Who we are: Data Foundation Platform



- Manage the cloud-based big data infrastructure for Disney Streaming
- Establish architecture, security standards and roll out best practices
- Manage the foundational services and tools for the platform

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Business challenge



Process petabytes of data across various business segments

Provide near real-time data to downstream systems

Power analytics, data science, and executive reporting to derive insights

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Platform at scale



Category	Total
AWS accounts	~100
AWS services used	50+
Storage by size	~20 petabytes
Teams using the platform	15+
Business entities running workloads	Disney+, ESPN+, Hulu

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Design considerations for data processing



- Support for data mesh architecture
  - Multi-tenancy and parallelism
    - High availability and reliability
      - Cost optimization and pre-emptive scaling
    - Observability – logging, monitoring, metrics
  - Security posture
- Applications and integrations

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR on EC2: Decisions





Cluster startup and tear down times



Thresholds identification for cluster scaling



High availability and reliability



Lack of fine-grained scaling – Higher costs



Amazon EMR log processing



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why Amazon EMR Serverless?

Pre-initialized capacity for faster job starts and fine-grained scaling at the job stage level

Support for multi-tenancy with multiple applications and parallelism with multiple jobs



Ability to submit new jobs to a healthy AZ in the event of AZ failure

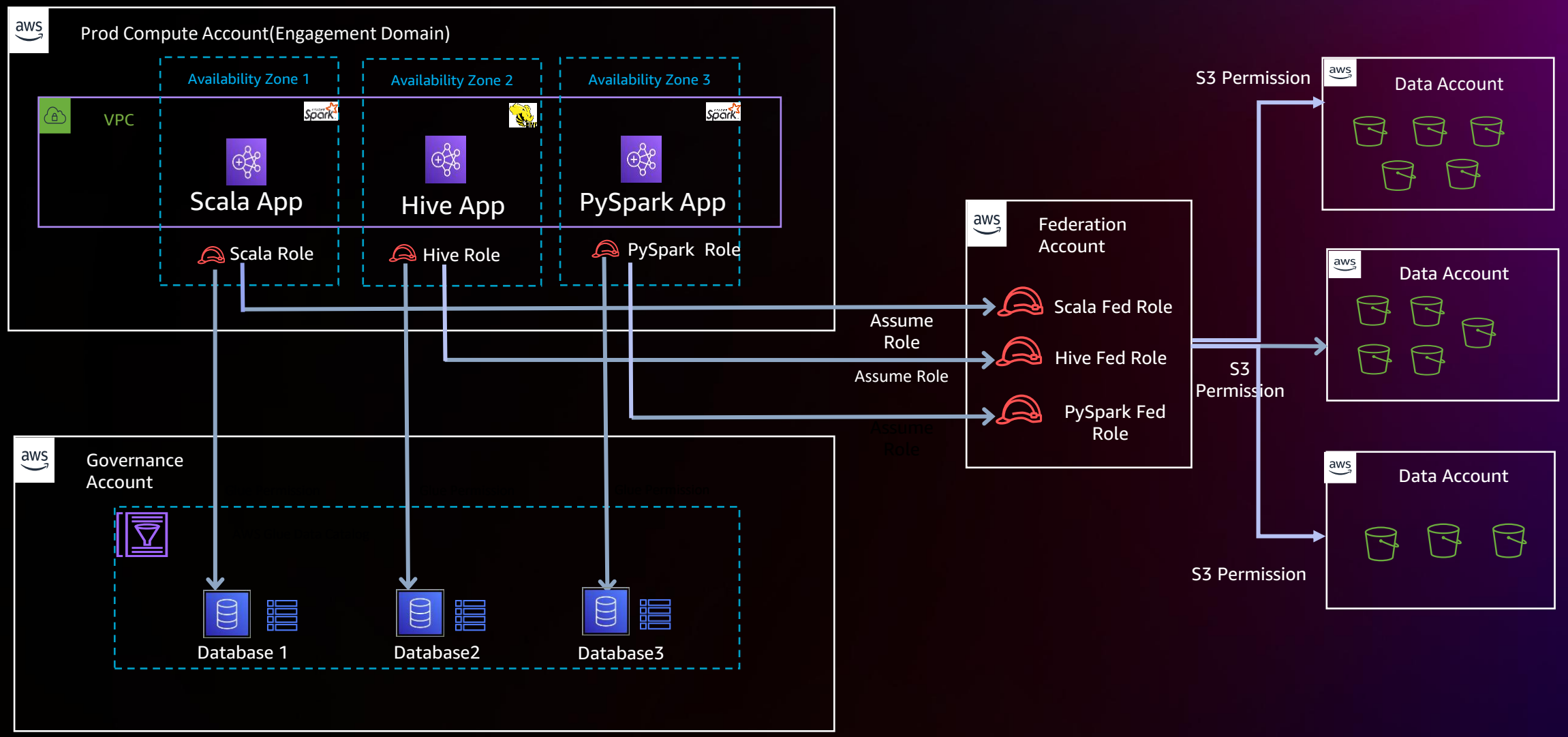
Support for cross-account and federated role to access data in Amazon S3

Support for application & job-level metrics in CloudWatch, real-time and historical Spark UI server, Hive Tez UI server

aws

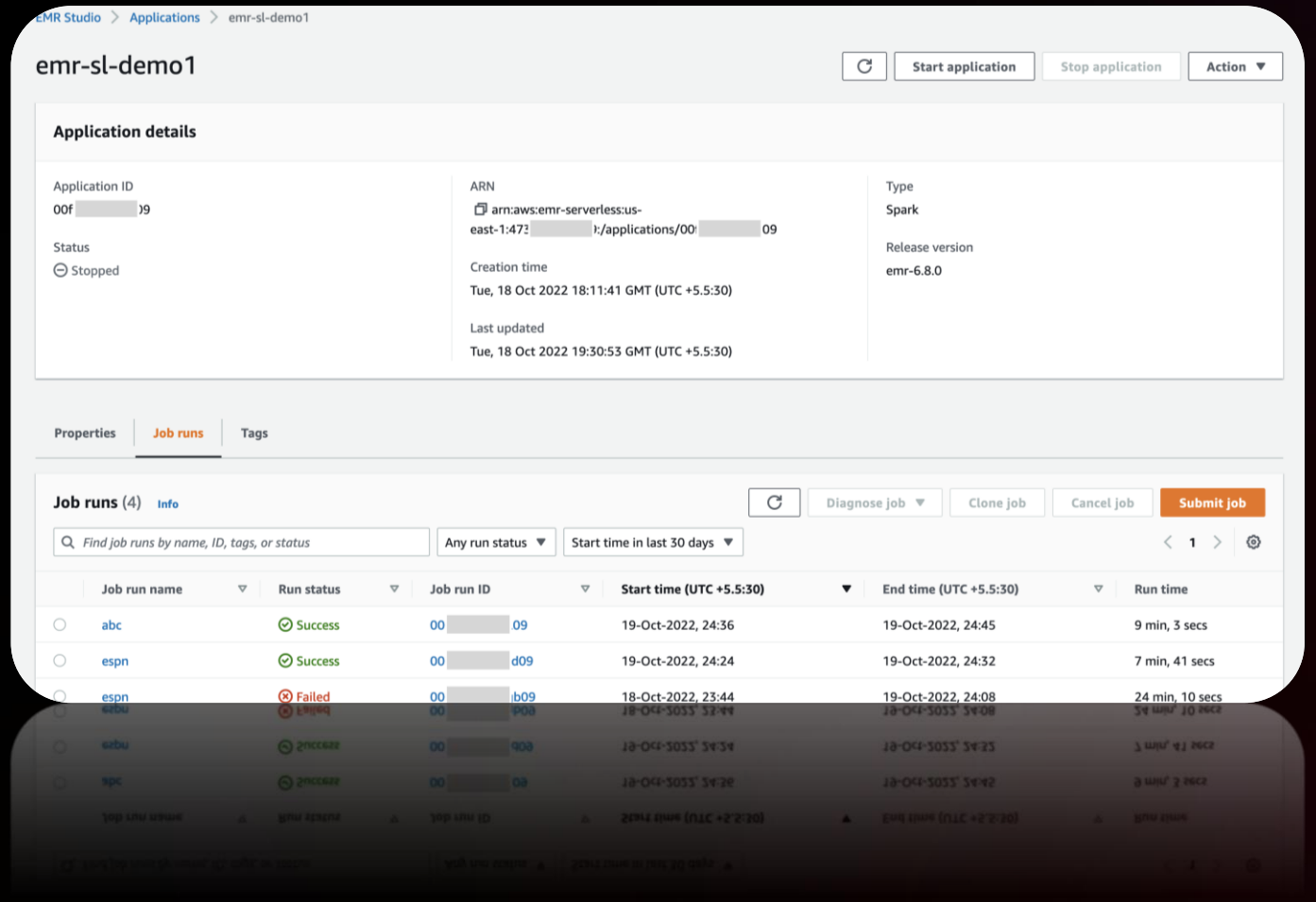
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

High-level architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Applications & jobs

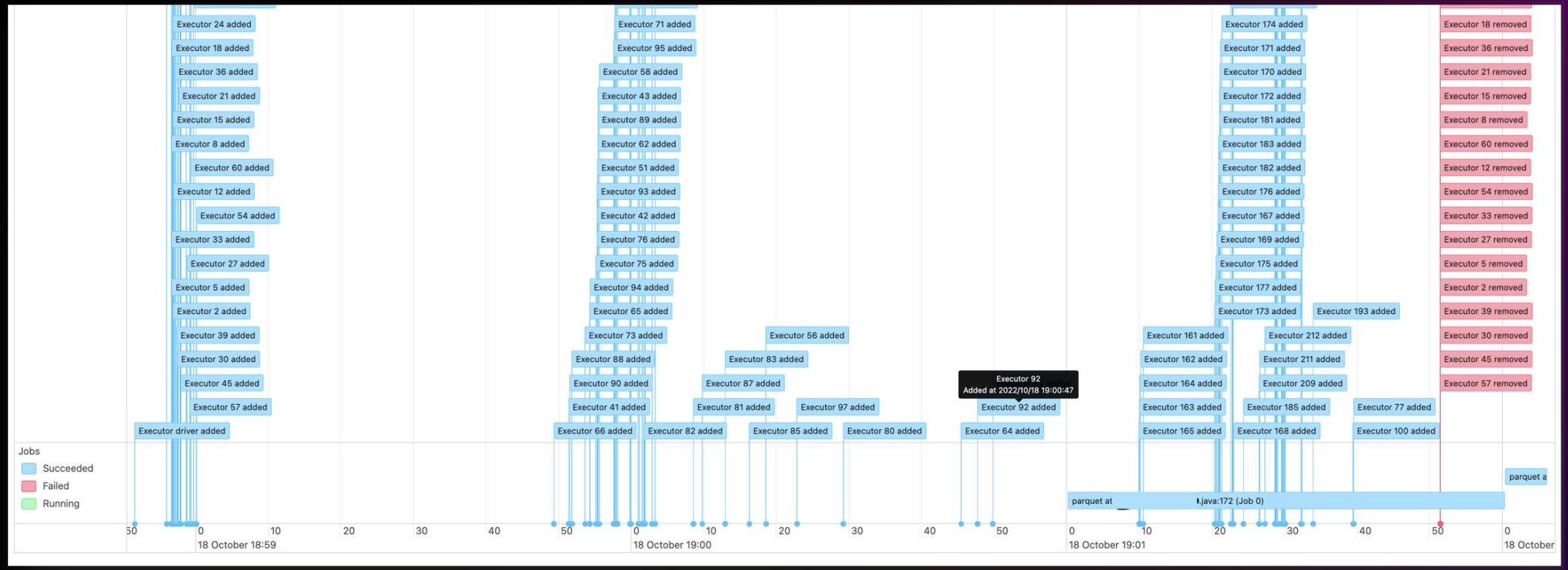



```
aws emr-serverless --region us-east-1 create-application \
  --type "SPARK" \
  --name emr-sl-demo \
  --release-label emr-6.8.0 \
  --initial-capacity '{
    "DRIVER": {
        "workerCount": 3,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "30GB"
        }
    },
    "EXECUTOR": {
        "workerCount": 100,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "30GB"
        }
    }
```

*All Jobs comply with the application config

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Fine-grained scaling

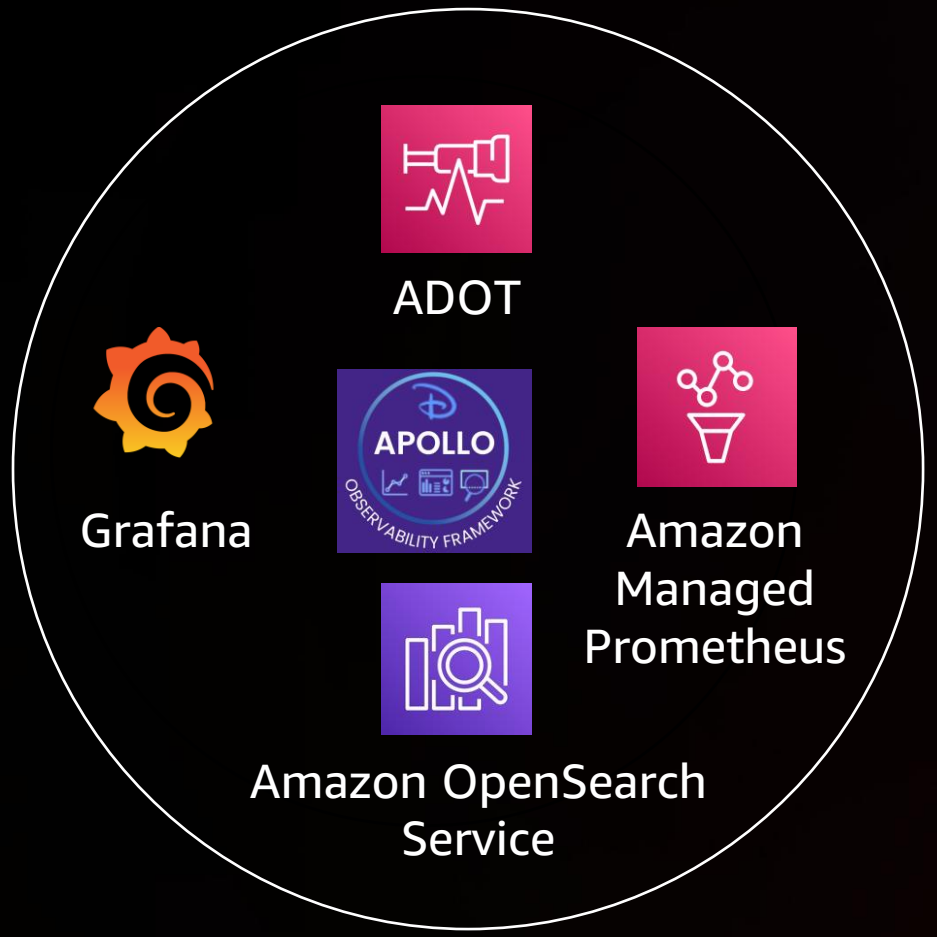


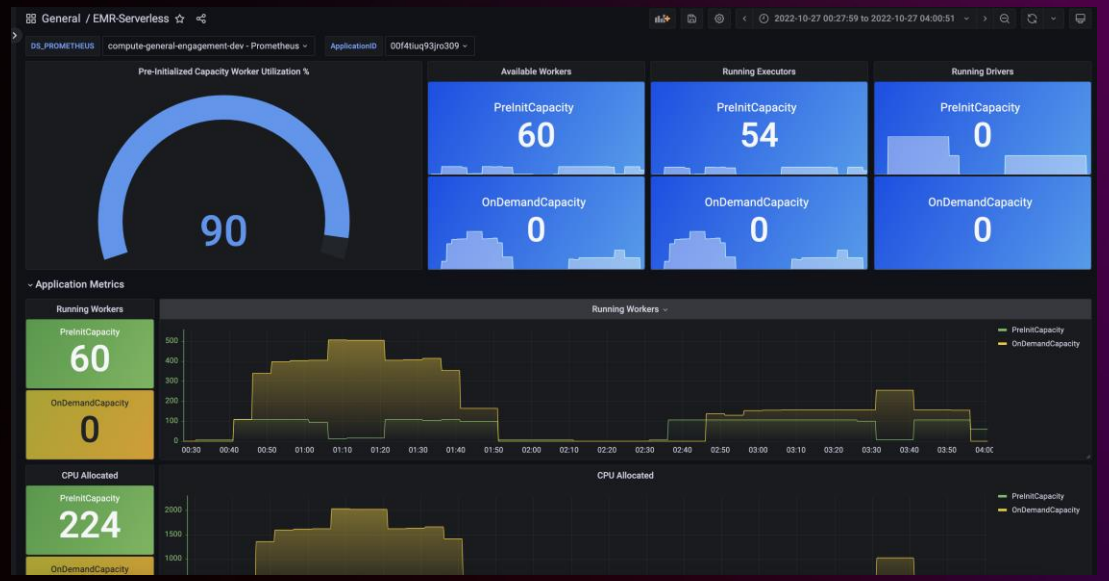
Initial capacity of 100 executors added within 1 minute before the spark jobs start at 1th minute

As the job starts executing, more executors are added on demand as required by each stage of the job and removed accordingly

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Observability for Amazon EMR Serverless







© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What we achieved...

- On-demand data processing
- Simplified job setup and organized job workflows
- No cluster management overhead
- Integration with Airflow using native operator
- Cost savings



Amazon EMR on EC2

1. Cluster config:
   Min nodes: 50
   Max nodes: 150
2. Time taken to run:
   Monolith job: 1hr 18 min
3. Cost to run:
   $274.01

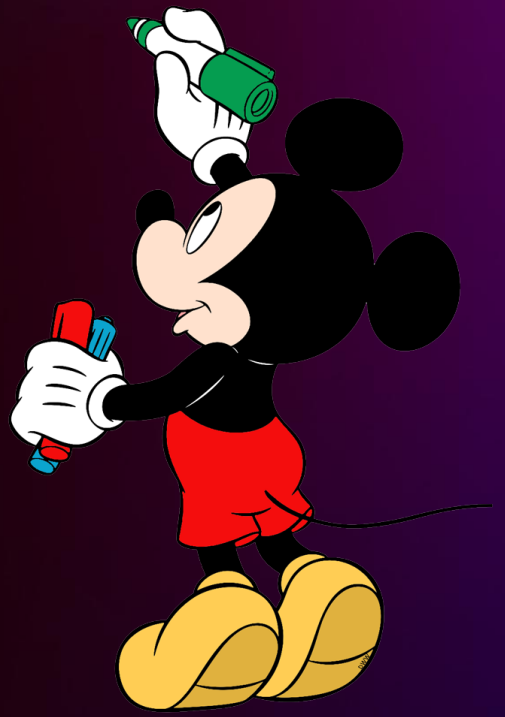
Amazon EMR Serverless

1. Cluster config:
   Min nodes: 50
   Max nodes: 150
2. Time taken to run:
   Parallel jobs: 54 mins 32s
3. Cost to run:
   $64.2

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What we learned…

1. Single application vs. multiple application
   - A/B testing with multiple applications
2. Fine-tuning configurations to speed up jobs
3. Configuring IAM roles for security adherence
4. Criteria for pre-initialized worker capacity
5. Taming the cost



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Radhika Ravirala
ravirala@amazon.com

Himateja Madala
Himateja.X.Madala@disney.com



Please complete the session survey in the **mobile app**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.