Build a Strong Foundation For Your Modern Applications With .NET 5 on AWS

Josh Hurley
.NET Developer Advocate
Amazon Web Services

2020-11-17

Agenda

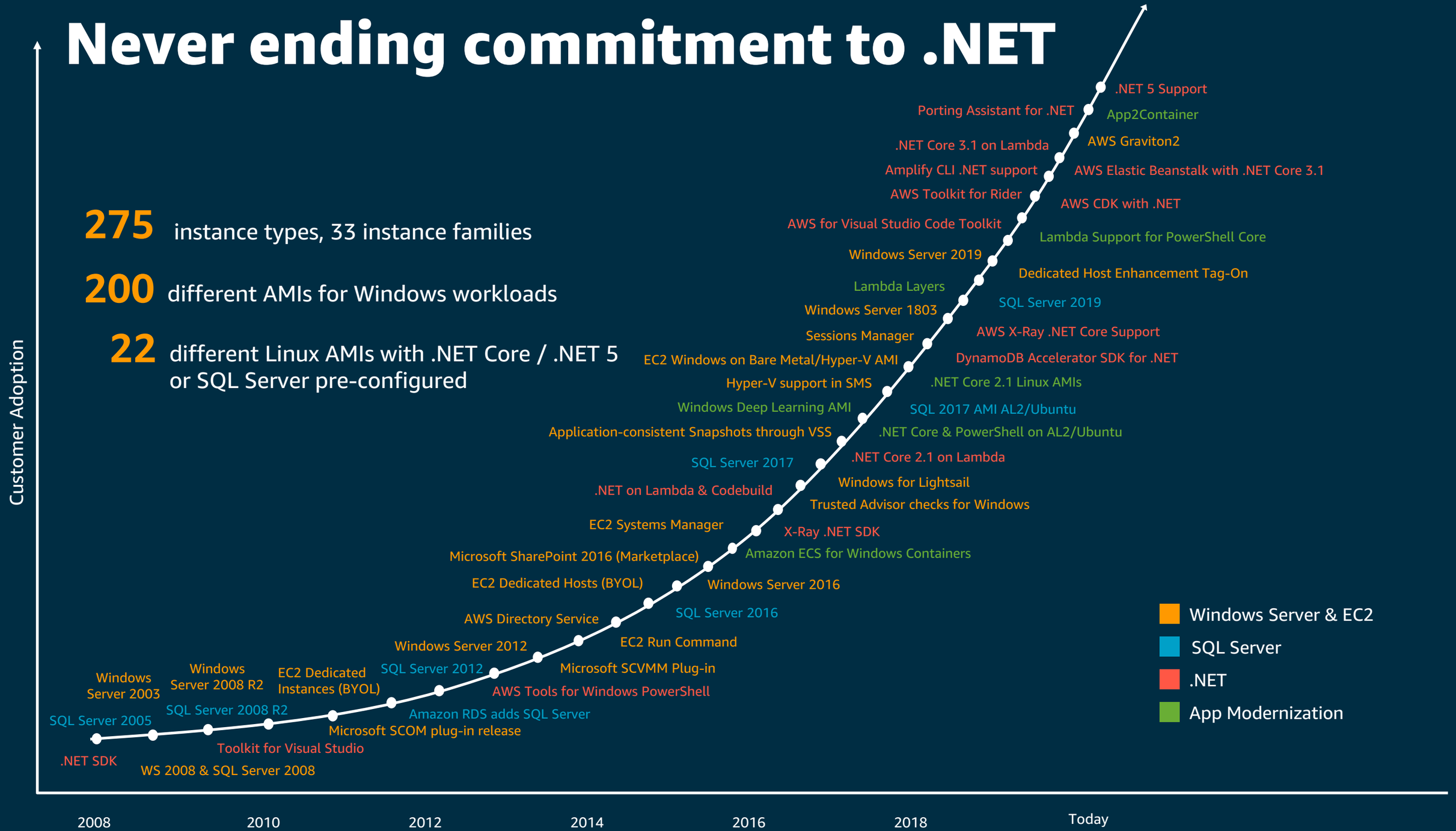
- Overview of .NET on AWS
- Hosting .NET applications on AWS
- Migrating .NET workloads to AWS
- New services and features for .NET 5
- AWS Tools for .NET developers
- AWS APN Partners
- Visual Studio Demo
- Wrap-up

aws

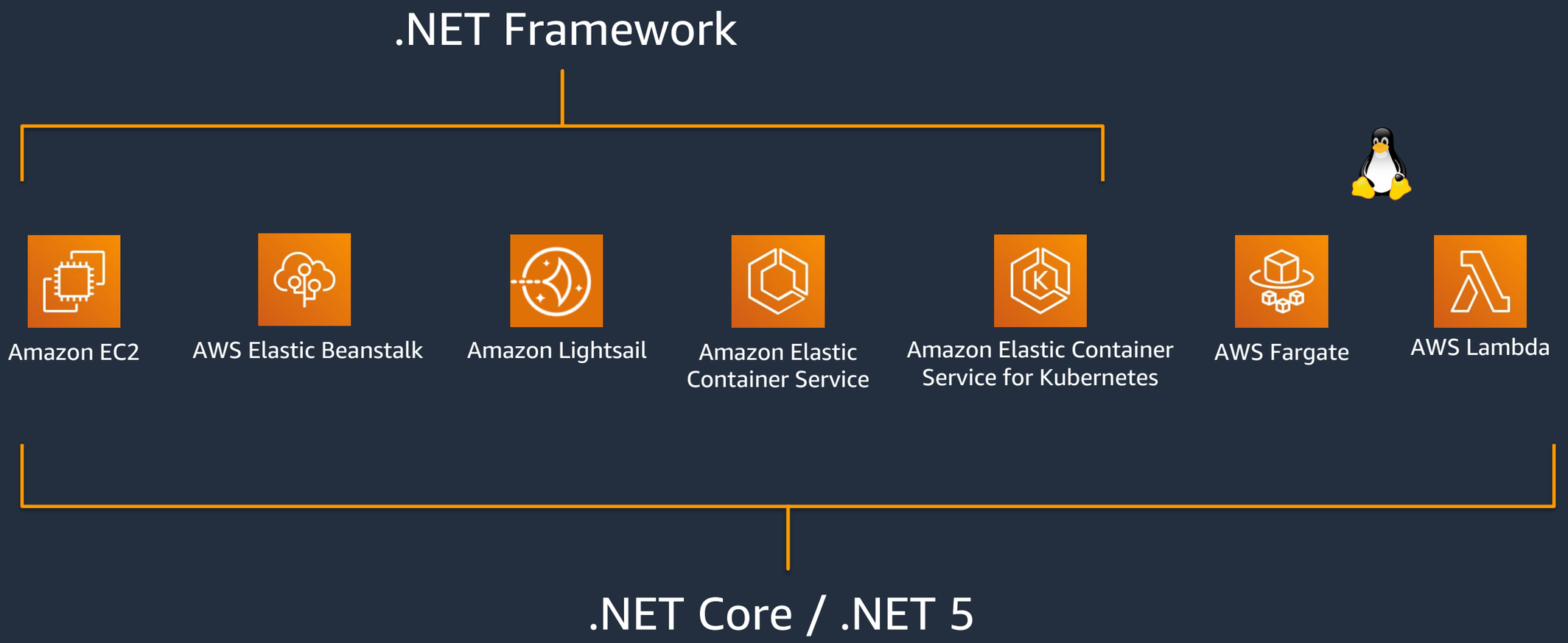

aws
.NET

Overview of .NET on AWS

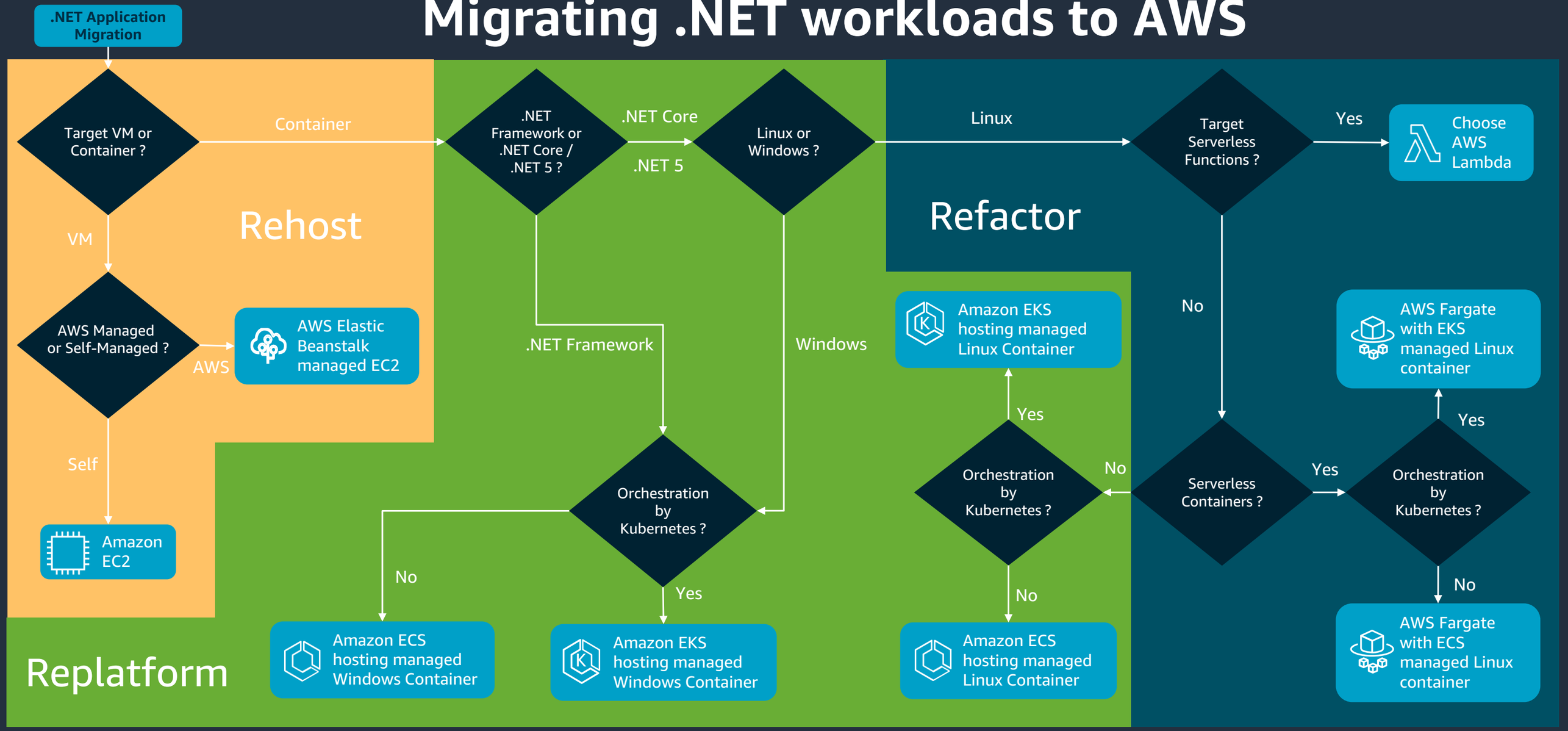
Never ending commitment to .NET



AWS Compute Services for .NET



Migrating .NET workloads to AWS

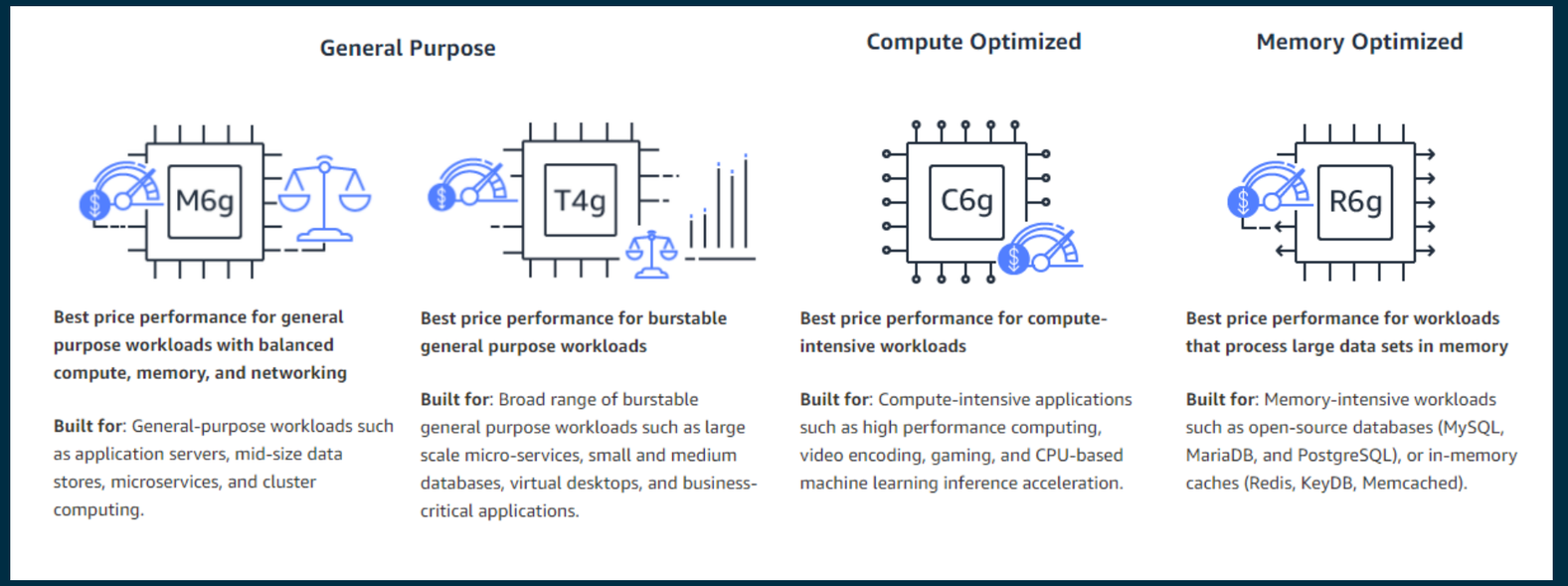


AWS Graviton Processor

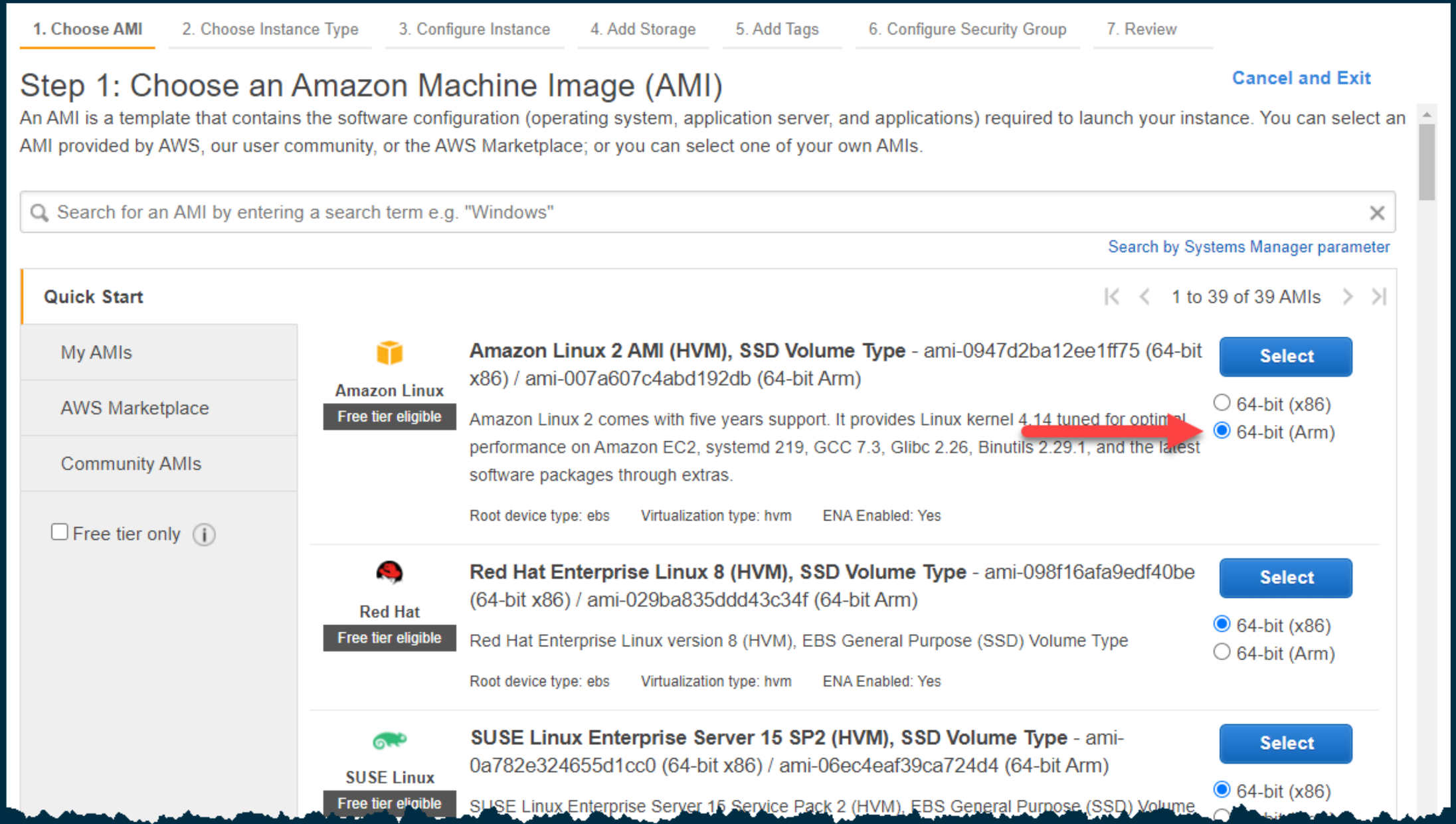
Improve price performance with the ARM64 and .NET 5

AWS is the only major cloud provider that enables an ARM64 architecture option using AWS Graviton2, helping customers running .NET 5 to realize ARM64 performance improvements with all .NET 5 Linux supported distributions (Alpine Linux, Debian, and Ubuntu).

Amazon EC2 instances running AWS Graviton2 provide up to 40% better price performance over comparable current generation x86-based instances.



https://aws.amazon.com/ec2/graviton/

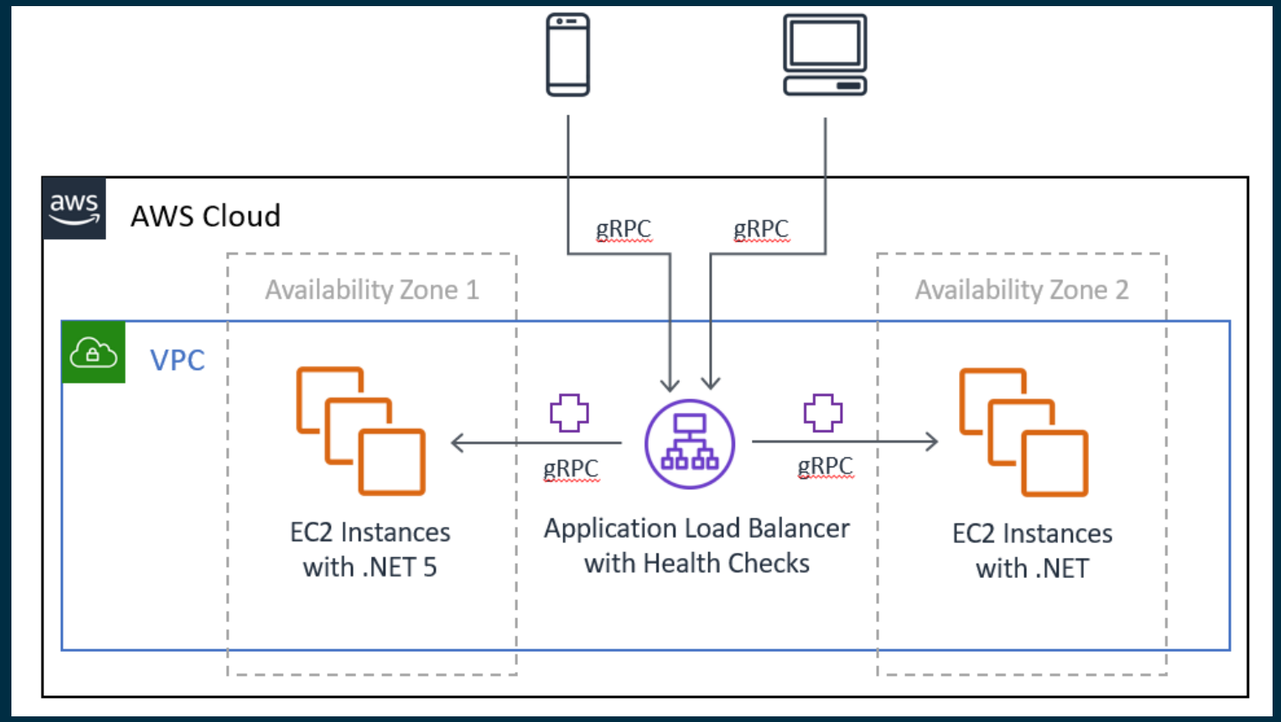


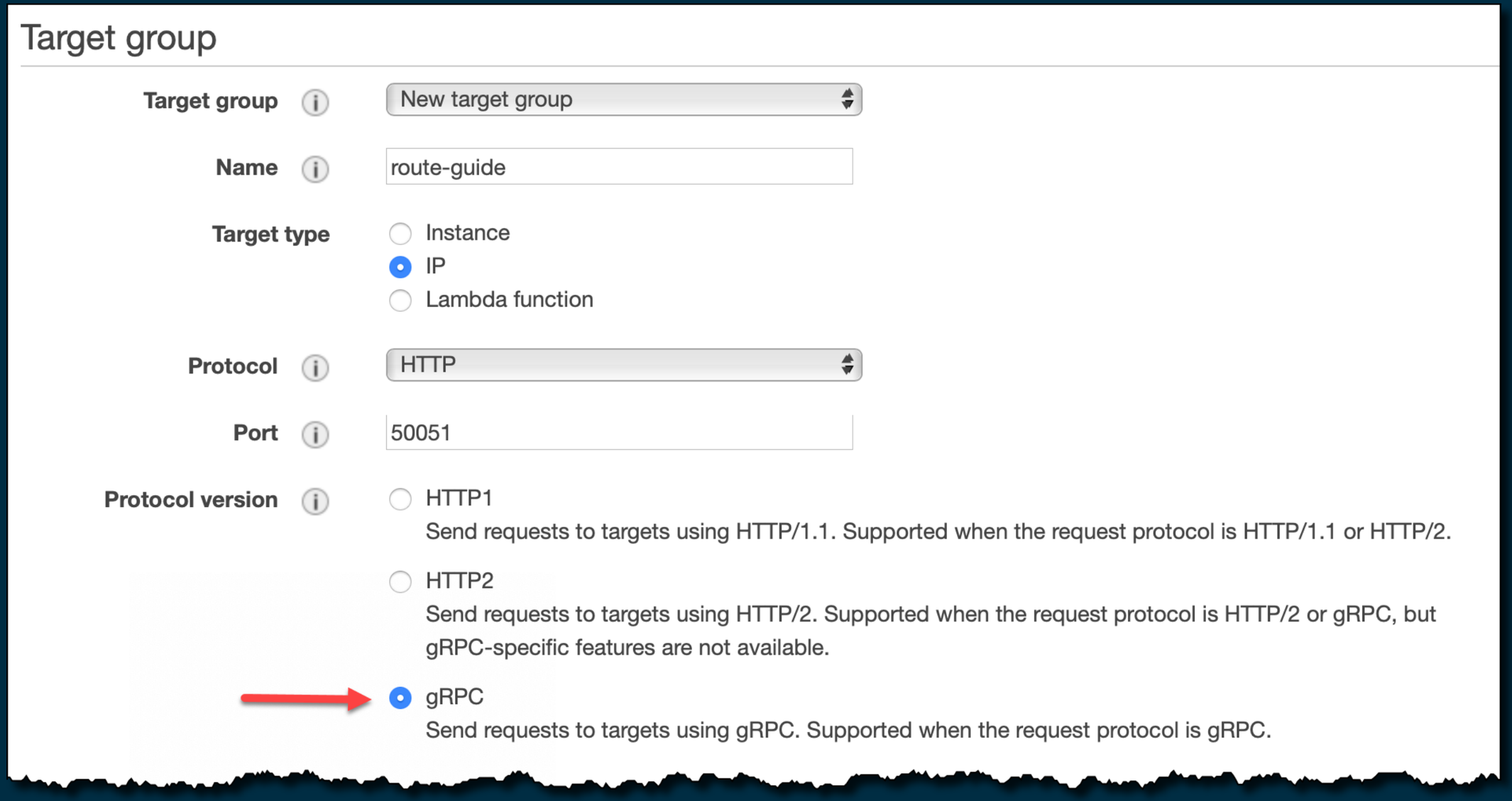
Application Load Balancer gRPC support

Accelerate microservice and mobile device communication with .NET 5

AWS is the only major cloud provider who, today, offers gRPC health checks and full status code reporting for gRPC

Gain faster communication and operational excellence with .NET 5 and gRPC for microservice architectures and mobile device communications.





https://amzn.to/2U1j57t

aws

AWS Tools for .NET Developers

What are the needs of a typical .NET developer?

Rich IDE and editor support

Intellisense

“Don’t make us leave the IDE”

“Getting started” project templates

Full featured, easily consumable SDK

Easy debugging and testing

Command-line support

PowerShell and Windows cmd.exe

‘dotnet CLI’ integration for .NET Core / .NET 5

CI/CD integration

Support for cross platform development with .NET Core / .NET 5

aws

Development Tools

IDE Integration

- AWS Toolkit for Visual Studio
- AWS Toolkit for Visual Studio Code
- AWS Toolkit for Rider

Programmable SDK

- AWS SDK for .NET
- AWS CDK for .NET

Command Line Tools

- AWS Tools for PowerShell
- ‘dotnet’ CLI extensions
- AWS CLI
- AWS SAM

CI/CD Integration

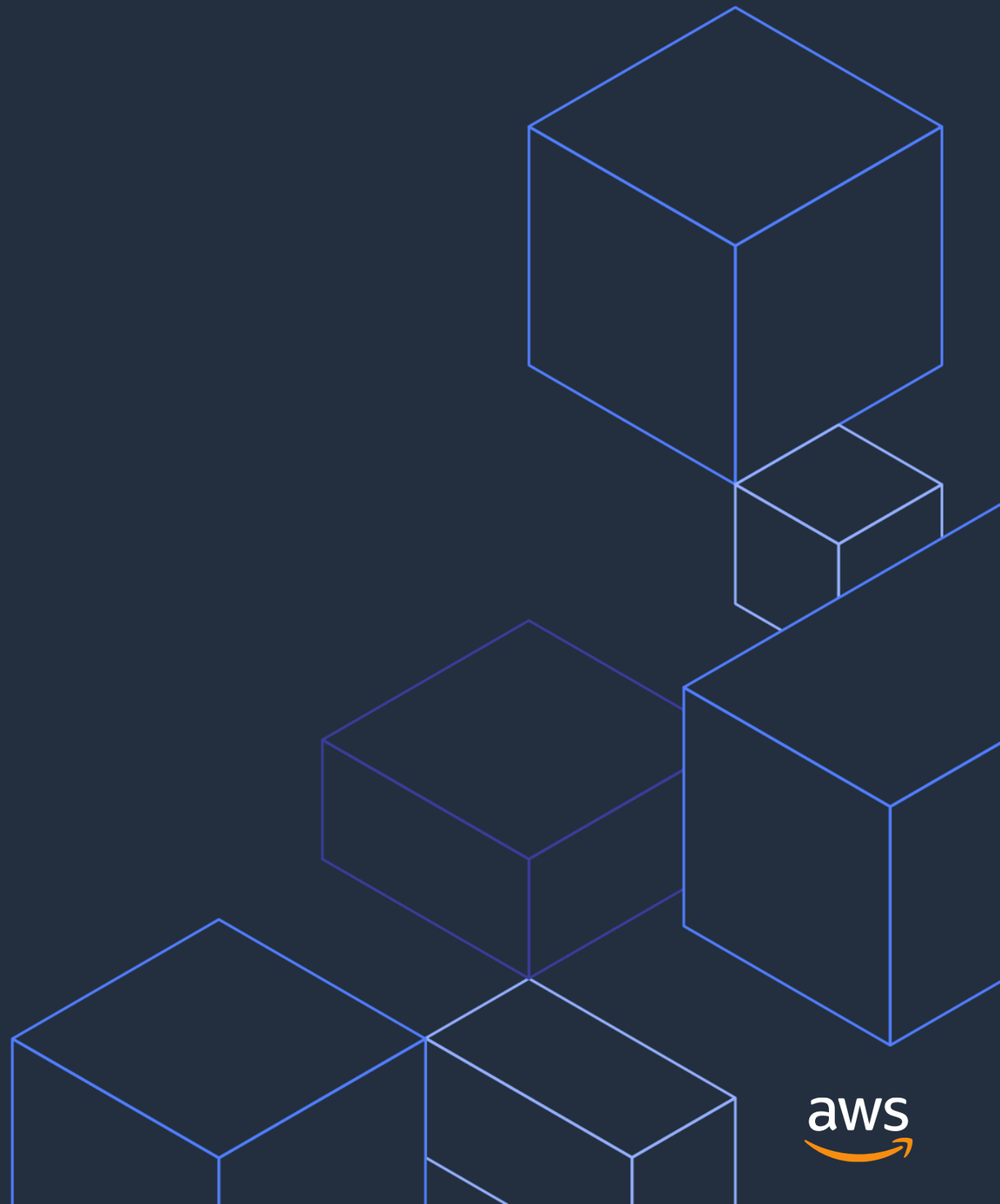
- AWS Toolkit for Azure DevOps
- AWS CodePipline/ CodeBuild

Migration / Modernization Tools

- App2Container
- Porting Assistant for .NET

aws

IDE Integration



AWS Toolkit for Visual Studio

AWS Toolkit for Visual Studio

AWS Explorer tool window and rich views

- Presents tree view of AWS services and resources of interest to developers
- Manage S3 Buckets
- View, create, edit, and delete Amazon DynamoDB tables
- Launch and manage EC2 compute instances

…and much more!

Wizards to easily deploy applications

- AWS Elastic Beanstalk
- Amazon Elastic Container Service
- AWS Lambda
- AWS CloudFormation

"Getting started" project templates ("blueprints") for serverless functions and applications

Code-editing Intellisense for AWS CloudFormation json templates

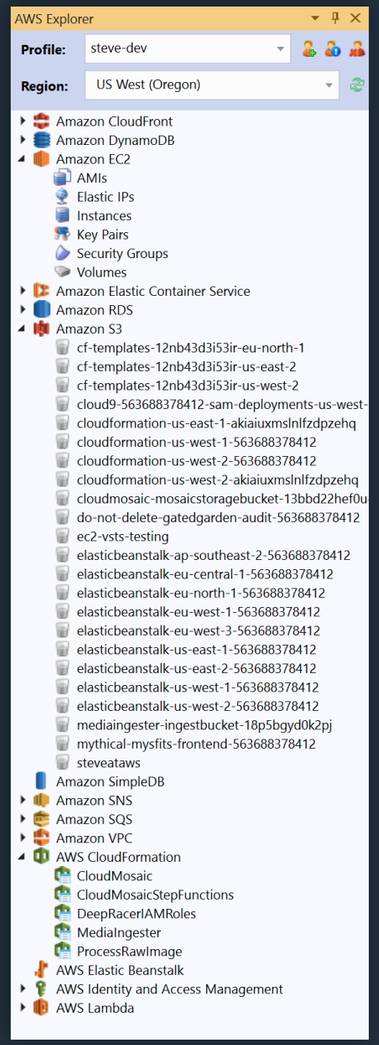
https://aws.amazon.com/visualstudio

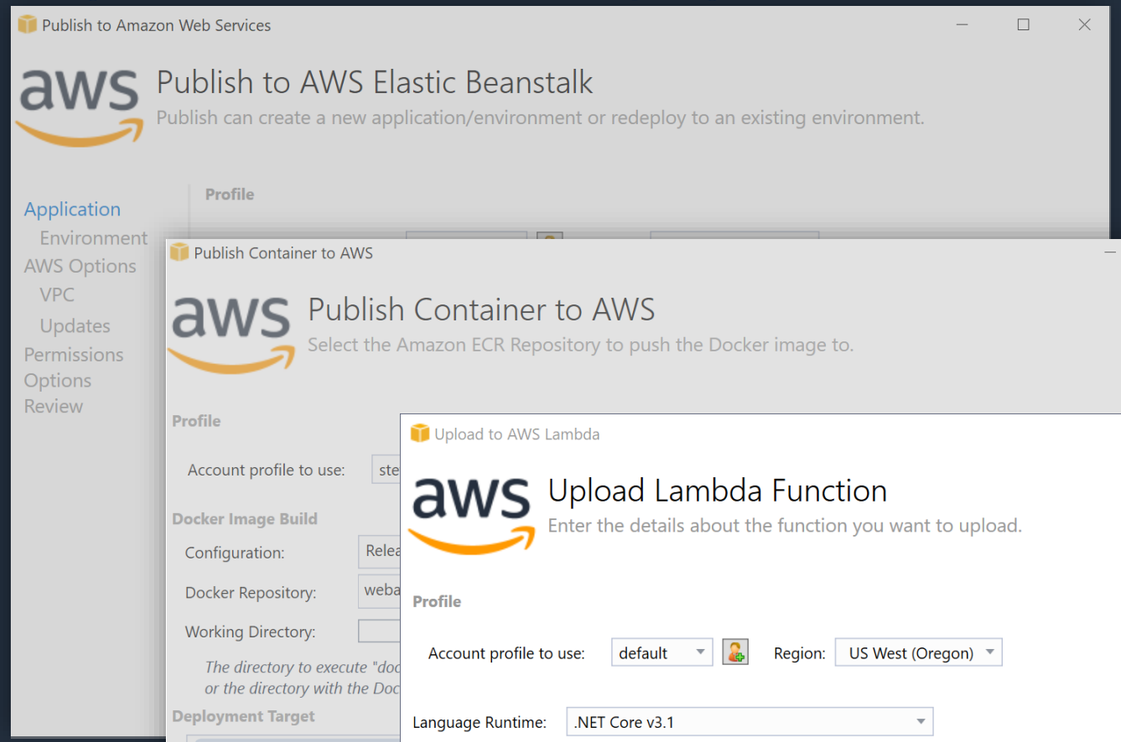
aws

AWS Toolkit for Visual Studio

AWS Toolkit for Visual Studio

Easy-to-use wizards to deploy…





Web applications to AWS Elastic Beanstalk

Containers to Amazon Elastic Container Service

Serverless functions and applications to AWS Lambda

Explore AWS services and resources

Demo

Getting started with the AWS Toolkit for Visual Studio

aws

AWS Toolkit for Visual Studio Code

Integrated experience targeting development of serverless applications in Node.js, Python and .NET:

"Getting started" project template

Step-through debugging

Deployment from the IDE

https://aws.amazon.com/visualstudiocode
https://github.com/aws/aws-toolkit-vscode

aws

AWS Toolkit for Rider

Open source plug-in for the Rider IDE that makes it easier to create, debug, and deploy .NET applications on Amazon Web Services:

Select a quickstart serverless application template.

Step-through debugging

Deployment from the IDE

Access CloudWatch From the IDE

https://aws.amazon.com/rider
https://github.com/aws/aws-toolkit-jetbrains

aws

Programmable SDK

aws

AWS SDK for .NET

AWS SDK for .NET

.NET

Central home for all SDK and extension libraries

https://github.com/aws/dotnet

Programmatic access to all AWS services

- Updates almost daily (in sync with service updates and launches)
- Simple programming model

Various extension libraries cover ‘higher level’ functionality

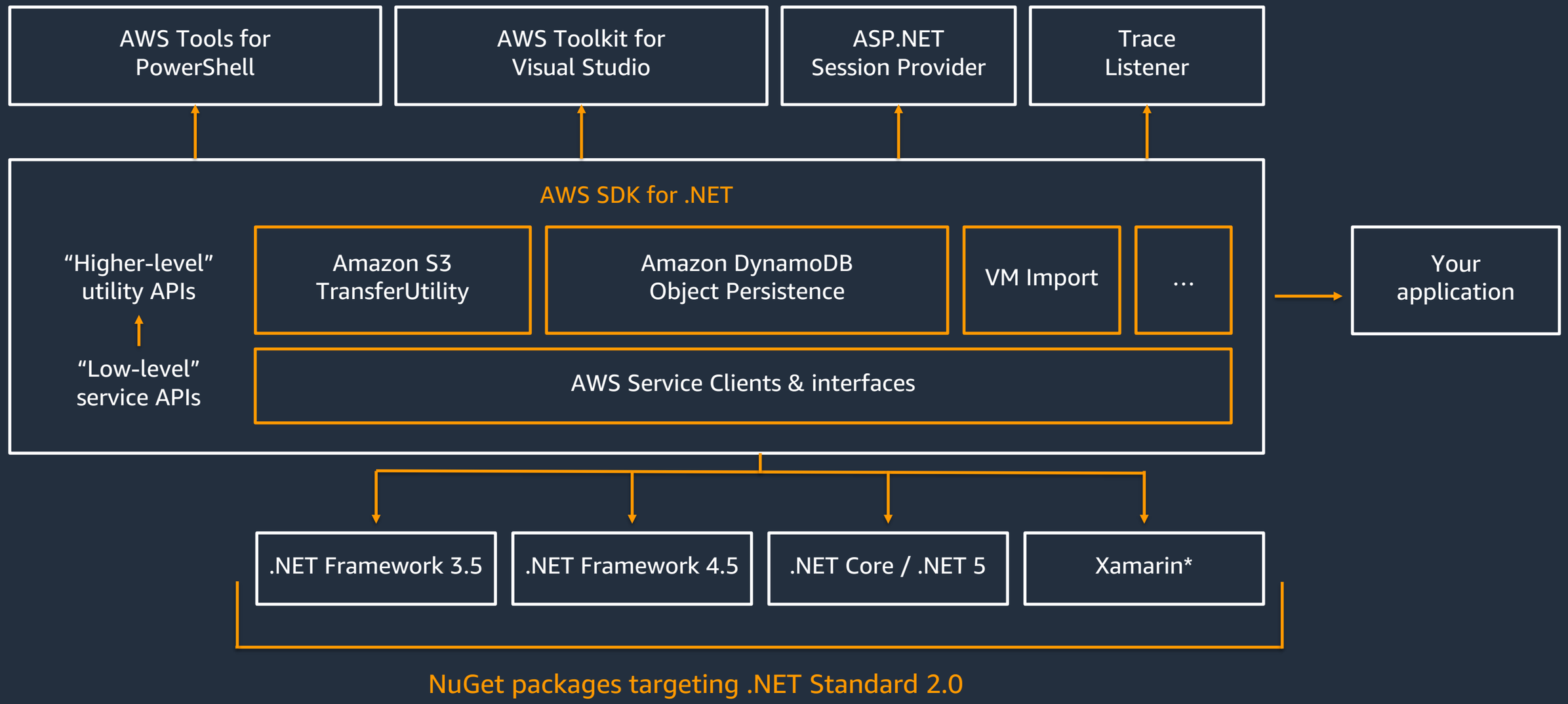
Cross-platform support for .NET

- Supports .NET Framework, .NET Core, .NET 5, and Xamarin development
- Open source

Distributed via NuGet

aws

Core Enabling Technology



Consistent SDK Coding Pattern

AWS SDK for .NET

.NET

```
// Add the namespaces for the service's client and request/response types
using servicename;
using servicename.Model;

// Create a client object representing the service in a region
var client = new AmazonservicenameClient(Amazon.RegionEndpoint.USWest2);

// Call a service operation, represented by a method on the client, with .NET types used to convey
request/response data. The SDK will serialize/deserialize the wire-level data format for you
operationnameResponse response = await client.operationnameAsync(new operationnameRequest
{
    RequestProperty1 = "some data",
    RequestProperty2 = new List<string> { "someother", "data" }
});

// process the deserialized data in the response object
foreach (var element in response.OutputDataMember)
{
   // do something
}
```


aws

Real-world example — list all objects in an S3 bucket

```
4   using Amazon.S3;
5   using Amazon.S3.Model;
6
7   namespace ConsoleApp
8   {
        0 references
9       class Program
10      {
            1 reference
11          static async Task ListS3BucketObjects()
12          {
13              using (var s3Client = new AmazonS3Client(Amazon.RegionEndpoint.USWest2))
14              {
15                  string nextPageMarker = null;
16                  do
17                  {
18                      var listObjectsResponse = await s3Client.ListObjectsAsync(new ListObjectsRequest
19                      {
20                          BucketName = "my-bucket-name",
21                          Marker = nextPageMarker
22                      });
23
24                      nextPageMarker = listObjectsResponse.NextMarker;
25
26                      foreach (var s3Object in listObjectsResponse.S3Objects)
27                      {
28                          Console.WriteLine($"{s3Object.Key}, last modified {s3Object.LastModified.ToString("g")}");
29                      }
30                  } while (!string.IsNullOrEmpty(nextPageMarker));
31              }
32          }
33
```


aws

Command Line Tools

AWS Tools for PowerShell

Available for Windows PowerShell and PowerShell 6+

Use to manage AWS resources and services

Over 7,300 cmdlets across 200+ services

Distributed on PowerShell Gallery

AWS.Tools.*

- Smaller, per-service modules
- Windows PowerShell v3-5.1
- PowerShell 6+
  (Windows, macOS & Linux)

AWSPowerShell

- Windows PowerShell v2-v5.1
- Pre-installed on Amazon-provided EC2 Windows images
- Legacy / Backwards Compatible

AWSPowerShell.NetCore

- Windows PowerShell v3-5.1
- PowerShell 6+
  (Windows, macOS & Linux)
- Legacy / Backwards Compatible

If you can code something in an SDK, you can script it in PowerShell

aws

‘dotnet’ CLI extensions

‘dotnet’ CLI extensions

Cross-platform ‘dotnet CLI’ extensions

- Replicate the deployment wizards inside Visual Studio
- Tools for Lambda, Elastic Beanstalk, and ECS
- Project templates for Lambda
- Use on developer workstation or in CI/CD scenarios

Round-trip capable with Visual Studio integration

- ‘defaults’ file (json) carries deployment settings
- Code shared between CLI tools and IDE

Simple to install with dotnet tool install –g *toolname*

aws

‘dotnet’ CLI – project templates for Lambda

Administrator: Windows PowerShell

```
PS D:\> dotnet new -l
Usage: new [options]

Options:
  -h, --help          Displays help for this command.
  -l, --list          Lists templates containing the specified name. If no name is specified, lists all templates.
  -n, --name          The name for the output being created. If no name is specified, the name of the current directory is used.
  -o, --output        Location to place the generated output.
  -i, --install       Installs a source or a template pack.
  -u, --uninstall     Uninstalls a source or a template pack.
  --nuget-source      Specifies a NuGet source to use during install.
  --type              Filters templates based on available types. Predefined values are "project", "item" or "other".
  --force             Forces content to be generated even if it would change existing files.
  -lang, --language   Filters templates based on language and specifies the language of the template to create.


Templates                                           Short Name                          Language          Tags
----------------------------------------------------------------------------------------------------------------------------------------------
Order Flowers Chatbot Tutorial                      lambda.OrderFlowersChatbot          [C#]              AWS/Lambda/Function
Lambda Detect Image Labels                          lambda.DetectImageLabels            [C#], F#          AWS/Lambda/Function
Lambda Empty Function                               lambda.EmptyFunction                [C#], F#          AWS/Lambda/Function
Lex Book Trip Sample                                lambda.LexBookTripSample            [C#]              AWS/Lambda/Function
Lambda Simple DynamoDB Function                     lambda.DynamoDB                     [C#], F#          AWS/Lambda/Function
Lambda Simple Kinesis Firehose Function             lambda.KinesisFirehose              [C#]              AWS/Lambda/Function
Lambda Simple Kinesis Function                      lambda.Kinesis                      [C#], F#          AWS/Lambda/Function
Lambda Simple S3 Function                           lambda.S3                           [C#], F#          AWS/Lambda/Function
Lambda Simple SQS Function                          lambda.SQS                          [C#]              AWS/Lambda/Function
Lambda ASP.NET Core Web API                         serverless.AspNetCoreWebAPI         [C#], F#          AWS/Lambda/Serverless
Lambda ASP.NET Core Web Application with Razor Pages serverless.AspNetCoreWebApp        [C#]              AWS/Lambda/Serverless
Serverless Detect Image Labels                      serverless.DetectImageLabels        [C#], F#          AWS/Lambda/Serverless
Lambda DynamoDB Blog API                            serverless.DynamoDBBlogAPI          [C#]              AWS/Lambda/Serverless
Lambda Empty Serverless                             serverless.EmptyServerless          [C#], F#          AWS/Lambda/Serverless
Lambda Giraffe Web App                              serverless.Giraffe                  F#                AWS/Lambda/Serverless
Serverless Simple S3 Function                       serverless.S3                       [C#], F#          AWS/Lambda/Serverless
Step Functions Hello World                          serverless.StepFunctionsHelloWorld  [C#], F#          AWS/Lambda/Serverless
```

Install: `dotnet new -i "Amazon.Lambda.Templates::*"`

aws

‘dotnet’ CLI tools – AWS Lambda

```
steve-demo@us-west-2 C:\AWS
> dotnet lambda help
Amazon Lambda Tools for .NET Core applications (4.1.0)
Project Home: https://github.com/aws/aws-extensions-for-dotnet-cli, https://github.com/aws/aws-lambda-dotnet



Commands to deploy and manage AWS Lambda functions:

        deploy-function         Command to deploy the project to AWS Lambda
        invoke-function         Command to invoke a function in Lambda with an optional input
        list-functions          Command to list all your Lambda functions
        delete-function         Command to delete a Lambda function
        get-function-config     Command to get the current runtime configuration for a Lambda function
        update-function-config  Command to update the runtime configuration for a Lambda function

Commands to deploy and manage AWS Serverless applications using AWS CloudFormation:

        deploy-serverless       Command to deploy an AWS Serverless application
        list-serverless         Command to list all your AWS Serverless applications
        delete-serverless       Command to delete an AWS Serverless application

Commands to publish and manage AWS Lambda Layers:

        publish-layer           Command to publish a Layer that can be associated with a Lambda function
        list-layers             Command to list Layers
        list-layer-versions     Command to list versions for a Layer
        get-layer-version       Command to get the details of a Layer version
        delete-layer-version    Command to delete a version of a Layer

Other Commands:

        package                 Command to package a Lambda project into a zip file ready for deployment
        package-ci              Command to use as part of a continuous integration system.

To get help on individual commands execute:
        dotnet lambda help <command>
```

Install: `dotnet tool install -g Amazon.Lambda.Tools`

aws

‘dotnet’ CLI tools – Amazon ECS

```
steve-demo@us-west-2 C:\AWS
> dotnet ecs help
Amazon EC2 Container Service Tools for .NET Core applications (3.1.0)
Project Home: https://github.com/aws/aws-extensions-for-dotnet-cli



Commands to deploy to Amazon EC2 Container Service:

        deploy-service          Push the application to ECR and runs the application as a long lived service on the ECS Cluster.
        deploy-task             Push the application to ECR and then runs it as a task on the ECS Cluster.
        deploy-scheduled-task   Push the application to ECR and then sets up CloudWatch Event Schedule rule to run the application.

Commands to manage docker images to Amazon EC2 Container Registry:

        push-image              Execute "dotnet publish", "docker build" and then push the image to Amazon ECR.

To get help on individual commands execute:
        dotnet ecs help <command>
```

Install: `dotnet tool install -g Amazon.ECS.Tools`

aws

‘dotnet’ CLI tools – AWS Elastic Beanstalk

```
steve-demo@us-west-2 C:\AWS
> dotnet eb help
Amazon Elastic Beanstalk Tools for .NET Core applications (4.0.0)
Project Home: https://github.com/aws/aws-extensions-for-dotnet-cli



Commands to manage environments:

        deploy-environment      Deploy the application to an AWS Elastic Beanstalk environment.
        list-environments       List the AWS Elastic Beanstalk environments.
        delete-environment      Delete an AWS Elastic Beanstalk environment.

Other Commands:

        package                 Package the application to a zip file to be deployed later to an Elastic Beanstalk environment

To get help on individual commands execute:
        dotnet eb help <command>
```

Install: dotnet tool install –g Amazon.ElasticBeanstalk.Tools

aws

CI/CD Integrations

AWS Toolkit for Azure DevOps

Supports online Azure DevOps (VSTS) and on-prem Azure DevOps Server (TFS)

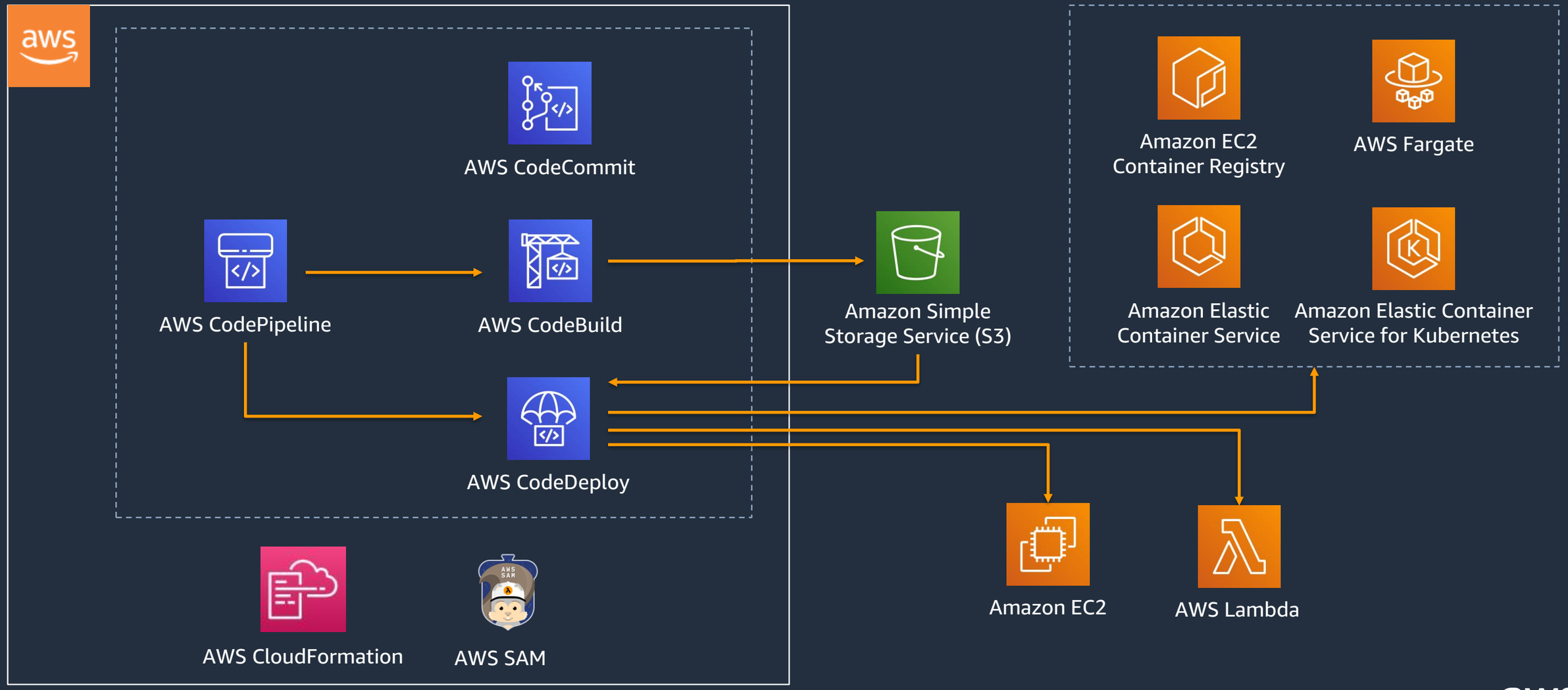
Collection of tasks for working with

Elastic Beanstalk	CloudFormation	CodeDeploy	Elastic Container Registry
S3 (upload and download)	Secrets Manager	Parameter Store	Lambda (all languages and .NET Core specific task)
SQS (Send SQS Messages)	SNS (Send SNS Messages)		

https://aws.amazon.com/vsts

aws

AWS CI/CD Services



aws

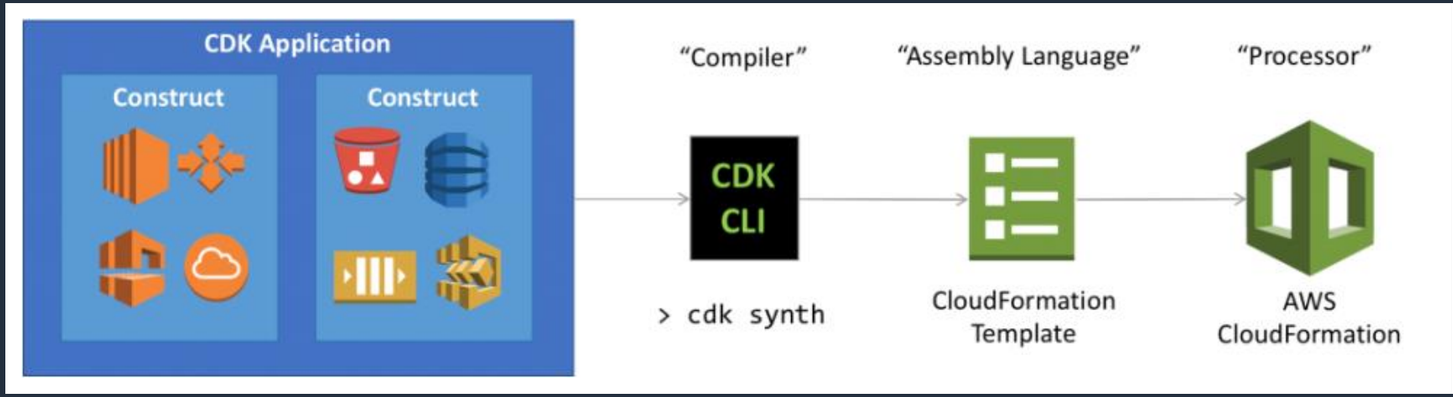
AWS Cloud Development Kit (CDK)

AWS CDK
for .NET

Define and provision cloud infrastructure and applications using .NET

Higher-level object-oriented abstraction to define and share AWS resources imperatively

Also supports Typescript, JavaScript and Python



aws

AWS CDK for .NET

The template defines a stack that contains an Amazon SQS queue and an Amazon SNS topic. The queue is subscribed to the topic, so whenever a message is sent to the topic, the message is sent to the queue. The code to do this is simply the following:

C#

```
var queue = new Queue(this, "MyFirstQueue", new QueueProps
{
    VisibilityTimeoutSec = 300
}
var topic = new Topic(this, "MyFirstTopic", new TopicProps
{
    DisplayName = "My First Topic Yeah"
});
```

The stack also uses a custom construct that creates a variable number of buckets, through which you can grant read permissions on as a collective.

C#

```
HelloConstruct hello = new HelloConstruct(this, "Buckets", new HelloConstructProps()
{
    BucketCount = 5
});
User user = new User(this, "MyUser", new UserProps());
hello.GrantRead(user);
```


aws

Migration and Modernization Tools

aws

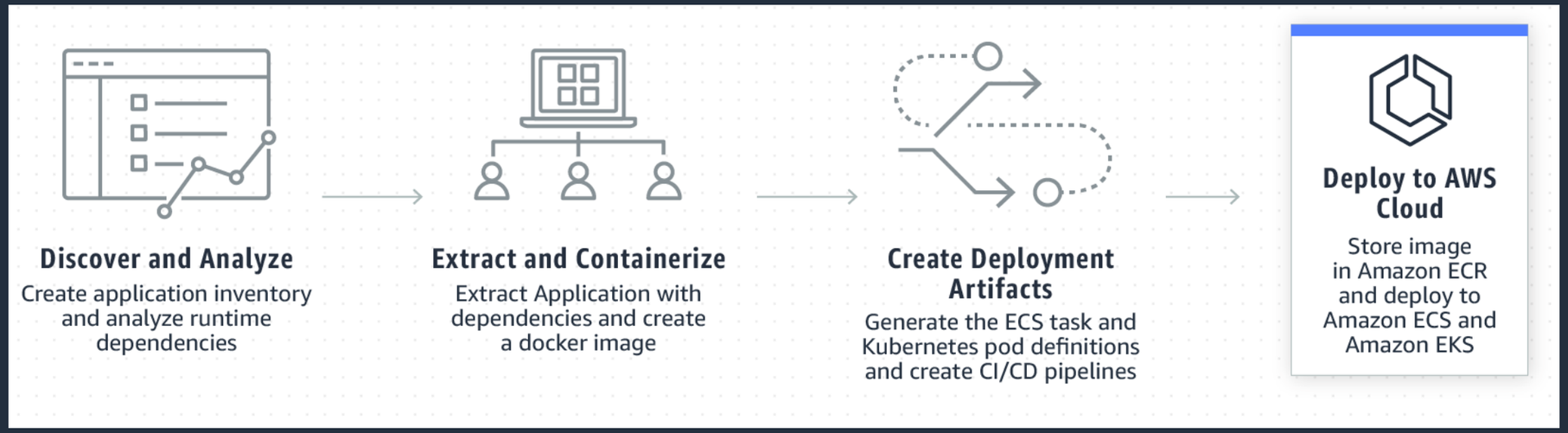
App2Container

Containerize and migrate existing applications

Streamline operations by containerizing your existing applications and standardize on a single set of tooling for monitoring, operations, and software delivery.

Analyze your applications and automatically generate a container image that is configured with the correct dependencies, network configurations, and deployment instructions for ECS or Kubernetes.

Easily deploy an existing application on the cloud that is provisioned with the correct networking and security configurations.



https://aws.amazon.com/app2container

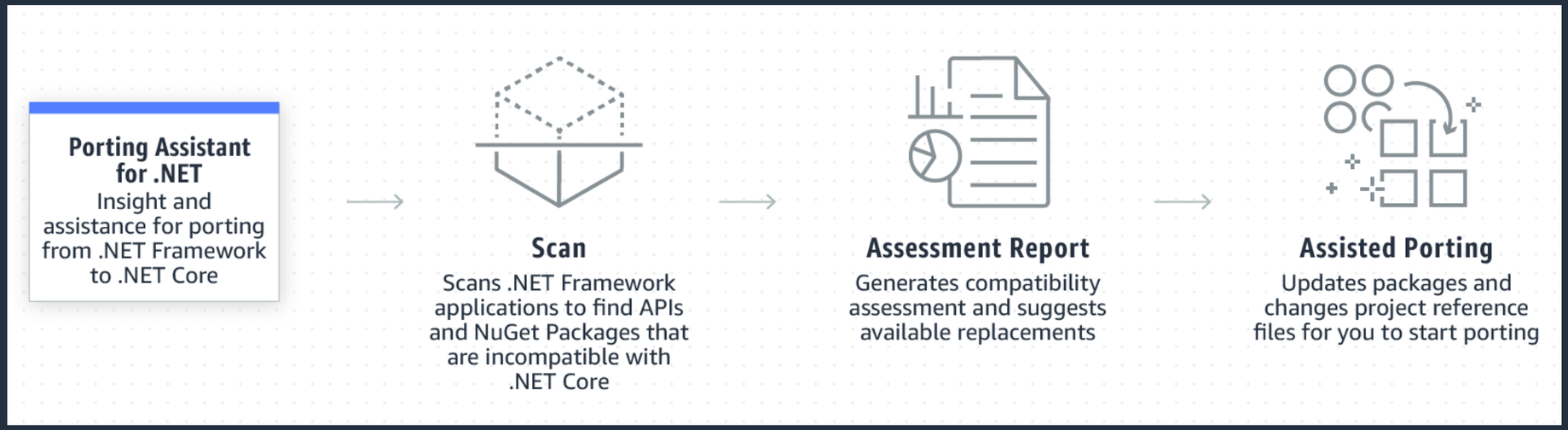
aws

Porting Assistant for .NET

Insight and assistance for porting from .NET Framework to .NET Core

Scan your entire .NET Framework application portfolio to generate .NET Core compatibility assessment reports. This makes it easy to prioritize applications for porting based on the level of effort required.

Identify incompatible .NET Core APIs and packages from your .NET Framework applications, and find known replacements reducing the manual effort of searching for replaceable packages and APIs.



https://aws.amazon.com/porting-assistant-dotnet

aws

AWS APN Partners

AWS Partner Network (APN)

GitLab provides CI/CD tools and services for the entire DevOps lifecycle.

GitLab CI with GitLab Runner supports building and testing applications, including supporting .NET.

GitLab is available both as a self-managed package to install, configure, and administer on your infrastructure or as a Software as a Service.

JetBrains has been building developer productivity software since 2000.

JetBrains has been an important partner as we have developed the AWS Toolkit for Rider, a Rider Plugin that that makes it easier to create, debug, and deploy .NET applications on AWS.

Atlassian provides team collaboration and productivity software.

The Atlassian CI/CD service Bamboo and Bitbucket, which is a Git compatible code management system, are tied into Atlassian product suite and can both integrate with AWS CodeDeploy.

aws

.NET 5 on AWS

Ready Today

Coming Soon

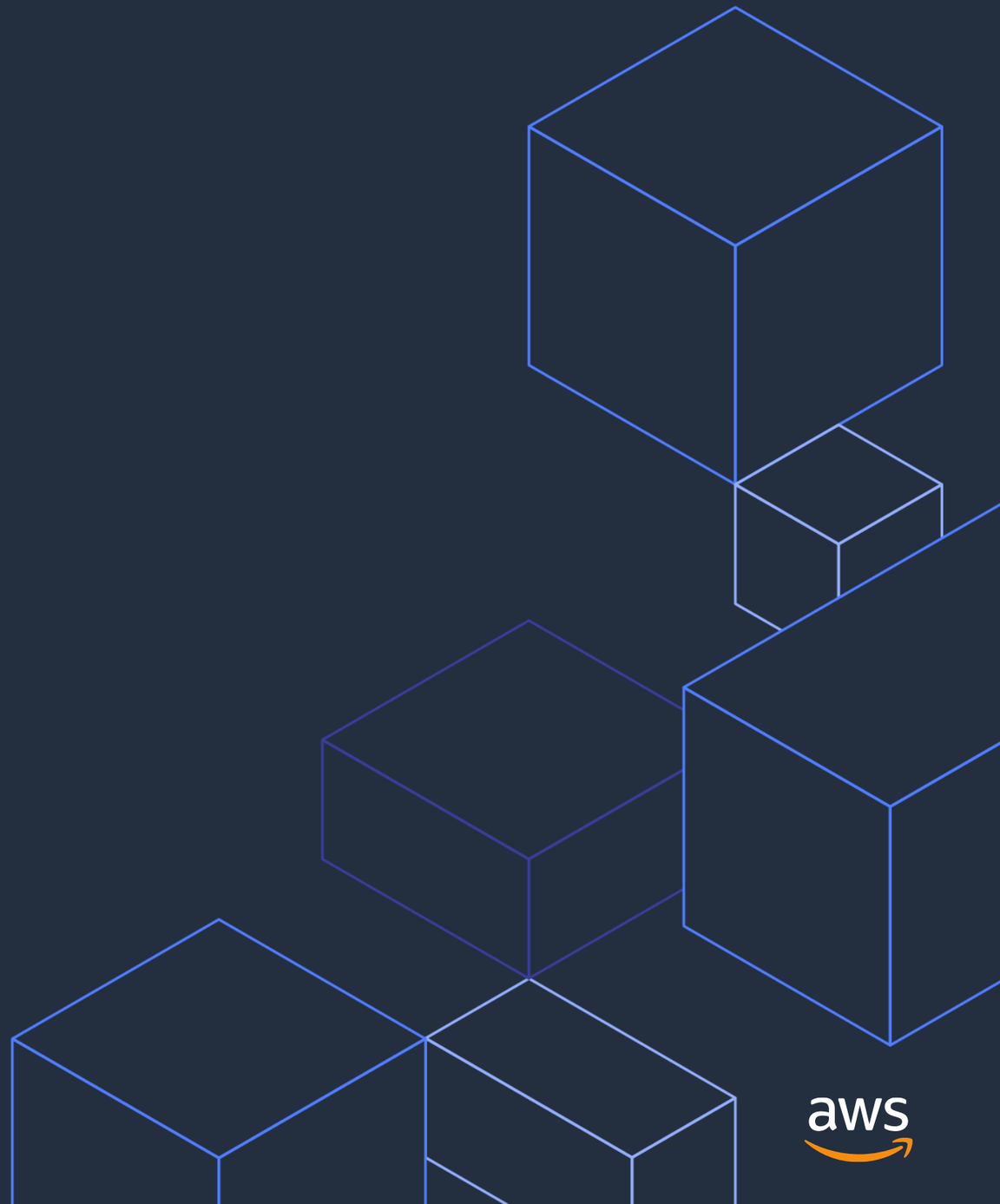
aws

Demo

Deploying from Visual Studio

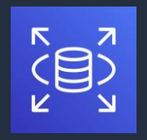
aws

Wrap-up



Other AWS Services of Interest


Amazon DynamoDB


Amazon RDS


Amazon SageMaker

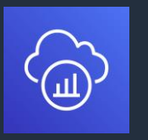

AWS Systems Manager


Amazon Cognito


AWS Step Functions


AWS Batch


Amazon CloudWatch


AWS X-Ray


High-level SDK libraries


Monitoring, insights into deployed code


PowerShell support throughout Systems Manager


Extension libs to publish logs to CloudWatch

aws

go.aws/build_net

aws


aws
.NET

Useful links

AWS .NET homepage
https://aws.amazon.com/net

Open source .NET tools homepage
https://github.com/aws/dotnet

Developing and Deploying .NET Applications on AWS Whitepaper
https://bit.ly/2QCznmJ

AWS Training and Certification course (fundamentals level)
Getting Started with .NET on AWS
https://bit.ly/2Fxn9bE

Pluralsight course from Julie Lerman:
Fundamentals of Building .NET Applications on AWS
https://bit.ly/2E3R8Y4

aws

Q&A

Thank You!