Introduction to DevOps on AWS

DevOps is the combination of cultural philosophies, engineering practices, and tools which increase an organization's ability to deliver applications and services at high velocity and better quality. Over time, several essential practices have emerged when adopting DevOps: continuous integration (CI), continuous delivery (CD), Infrastructure as Code (IaC), and monitoring and logging.

Topics:

Continuous integration/ Continuous delivery (CI/CD)

- AWS CodeCommit
- AWS CodeBuild
- AWS CodeArtifact
- AWS CodeDeploy
- AWS CodePipeline

Infrastructure as code

- AWS CloudFormation
- AWS Serverless Application Model
- AWS Cloud Development Kit (AWS CDK)
- AWS Cloud Development Kit for Kubernetes
- AWS Cloud Development Kit for Terraform
- AWS Cloud Control API

Automation and tooling

- AWS OpsWorks
- AWS Elastic Beanstalk
- AWS Proton
- AWS Cloud9
- AWS CloudShell
- Amazon CodeGuru

Continuous integration/ Continuous delivery (CI/CD)

Continuous integration (CI) is a software development practice where developers regularly merge their code changes into a central code repository, after which automated builds and tests are run. CI helps find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.

Continuous delivery (CD) is a software development practice where code changes are automatically prepared for a release to production. A pillar of modern application development, continuous delivery expands upon continuous integration by deploying all code changes to a testing environment and/or a production environment after the build stage. When properly implemented, developers will always have a deployment-ready build artifact that has passed through a standardized test process.

AWS offers the following services for CI/CD:

AWS CodeCommit

AWS CodeCommit is a secure, highly scalable, managed source control service that hosts private git repositories. CodeCommit reduces the need for you to operate your own source control system and there is no hardware to provision and scale or software to install, configure, and operate.

You can use CodeCommit to store anything from code to binaries, and it supports the standard functionality of GitHub, allowing it to work seamlessly with your existing Git-based tools. Your team can also use CodeCommit's online code tools to browse, edit, and collaborate on projects. AWS CodeCommit has several benefits:

- **Collaboration** — AWS CodeCommit is designed for collaborative software development. You can easily commit, branch, and merge your code, which helps you easily maintain control of your team's projects. CodeCommit also supports pull requests, which provide a mechanism to request code reviews and discuss code with collaborators.
- **Encryption** — You can transfer your files to and from AWS CodeCommit using HTTPS or SSH, as you prefer. Your repositories are also automatically encrypted at rest through AWS Key Management Service (AWS KMS) using customer-specific keys.
- **Access control** — AWS CodeCommit uses AWS Identity and Access Management (IAM) to control and monitor who can access your data in addition to how, when, and where they can access it. CodeCommit also helps you monitor your repositories through AWS CloudTrail and Amazon CloudWatch.

- **High availability and durability** — AWS CodeCommit stores your repositories in Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB. Your encrypted data is redundantly stored across multiple facilities. This architecture increases the availability and durability of your repository data.
- **Notifications and custom scripts** — You can now receive notifications for events impacting your repositories. Notifications will come as Amazon Simple Notification Service (Amazon SNS) notifications. Each notification will include a status message as well as a link to the resources whose event generated that notification. Additionally, using AWS CodeCommit repository cues, you can send notifications and create HTTP webhooks with Amazon SNS or invoke AWS Lambda functions in response to the repository events you choose.

AWS CodeBuild

AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy. You don't need to provision, manage, and scale your own build servers. CodeBuild can use either of GitHub, GitHub Enterprise, BitBucket, AWS CodeCommit, or Amazon S3 as a source provider.

CodeBuild scales continuously and can process multiple builds concurrently. CodeBuild offers various pre-configured environments for various versions of Microsoft Windows and Linux.

Customers can also bring their customized build environments as Docker containers. CodeBuild also integrates with open source tools such as Jenkins and Spinnaker.

CodeBuild can also create reports for unit, functional, or integration tests. These reports provide a visual view of how many tests cases were run and how many passed or failed. The build process

can also be run inside an Amazon Virtual Private Cloud (Amazon VPC) which can be helpful if your integration services or databases are deployed inside a VPC.

AWS CodeArtifact

AWS CodeArtifact is a fully managed artifact repository service that can be used by organizations to securely store, publish, and share software packages used in their software development process. CodeArtifact can be configured to automatically fetch software packages and dependencies from public artifact repositories, so developers have access to the latest versions.

Software development teams increasingly rely on open-source packages to perform common tasks in their application package. It has become critical for software development teams to maintain

control on a particular version of the open-source software to ensure the software is free of vulnerabilities. With CodeArtifact, you can set up controls to enforce this.

CodeArtifact works with commonly used package managers and build tools such as Maven, Gradle, npm, yarn, twine, and pip, making it easy to integrate into existing development workflows.

AWS CodeDeploy

AWS CodeDeploy is a fully managed deployment service that automates software deployments to a variety of compute services such as Amazon Elastic Compute Cloud (Amazon EC2), AWS Fargate, AWS Lambda, and your on-premises servers. AWS CodeDeploy makes it easier for you to rapidly release new features, helps you avoid downtime during application deployment, and handles the complexity of updating your applications. You can use CodeDeploy to automate software deployments, reducing the need for error-prone manual operations. The service scales to match your deployment needs.

CodeDeploy has several benefits that align with the DevOps principle of continuous deployment:

- **Automated deployments** — CodeDeploy fully automates software deployments, allowing you to deploy reliably and rapidly.
- **Centralized control** — CodeDeploy enables you to easily launch and track the status of your application deployments through the AWS Management Console or the AWS CLI. CodeDeploy gives you a detailed report enabling you to view when and to where each application revision was deployed. You can also create push notifications to receive live updates about your deployments.
- **Minimize downtime** — CodeDeploy helps maximize your application availability during the software deployment process. It introduces changes incrementally and tracks application health according to configurable rules. Software deployments can easily be stopped and rolled back if there are errors.
- **Easy to adopt** — CodeDeploy works with any application, and provides the same experience across different platforms and languages. You can easily reuse your existing setup code. CodeDeploy can also integrate with your existing software release process or continuous delivery toolchain (for example, AWS CodePipeline, GitHub, Jenkins).

AWS CodeDeploy supports multiple deployment options. For more information, refer to the *Deployment strategies* section of this document.

AWS CodePipeline

AWS CodePipeline is a continuous delivery service that you can use to model, visualize, and automate the steps required to release your software. With AWS CodePipeline, you model the full release process for building your code, deploying to pre-production environments, testing your application, and releasing it to production. AWS CodePipeline then builds, tests, and deploys your application according to the defined workflow every time there is a code change. You can integrate partner tools and your own custom tools into any stage of the release process to form an end-to-end continuous delivery solution.

AWS CodePipeline has several benefits that align with the DevOps principle of continuous deployment:

- **Rapid delivery** — AWS CodePipeline automates your software release process, allowing you to rapidly release new features to your users. With CodePipeline, you can quickly iterate on feedback and get new features to your users faster.
- **Improved quality** — By automating your build, test, and release processes, AWS CodePipeline enables you to increase the speed and quality of your software updates by running all new changes through a consistent set of quality checks.
- **Easy to integrate** — AWS CodePipeline can easily be extended to adapt to your specific needs. You can use the pre-built plugins or your own custom plugins in any step of your release process. For example, you can pull your source code from GitHub, use your on-premises Jenkins build server, run load tests using a third-party service, or pass on deployment information to your custom operations dashboard.
- **Configurable workflow** — AWS CodePipeline enables you to model the different stages of your software release process using the console interface, the AWS CLI, AWS CloudFormation, or the AWS SDKs. You can easily specify the tests to run and customize the steps to deploy your application and its dependencies.

Infrastructure as code

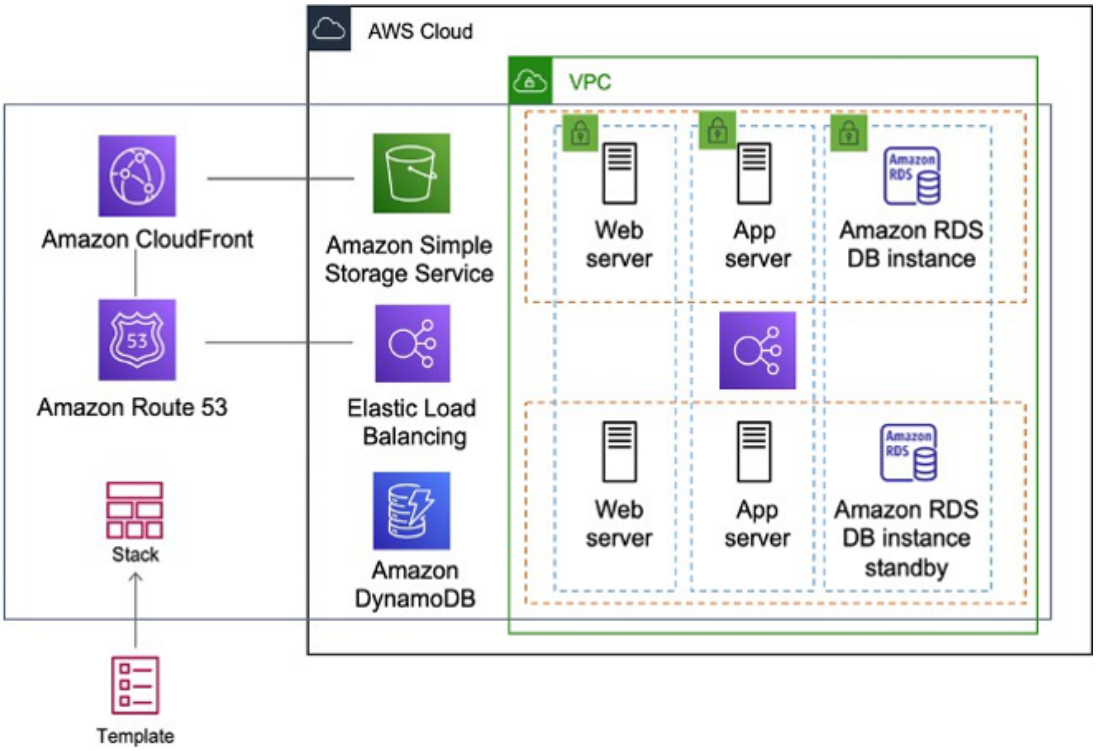
In contrast, AWS provides a DevOps-focused way of creating and maintaining infrastructure. Similar to the way software developers write application code, AWS provides services that enable the creation, deployment and maintenance of infrastructure in a programmatic, descriptive, and declarative way. These services provide rigor, clarity, and reliability. The AWS services discussed in this paper are core to a DevOps methodology and form the underpinnings of numerous higher-level AWS DevOps principles and practices.

AWS offers the following services to define infrastructure as code.

AWS CloudFormation

AWS CloudFormation is a service that enables developers to create AWS resources in an orderly and predictable fashion. Resources are written in text files using JSON or YAML format. The templates require a specific syntax and structure that depends on the types of resources being created and managed. You author your resources in JSON or YAML with any code editor such as AWS Cloud9, check it into a version control system, and then CloudFormation builds the specified services in safe, repeatable manner.

A CloudFormation template is deployed into the AWS environment as a stack. You can manage stacks through the AWS Management Console, AWS Command Line Interface, or AWS CloudFormation APIs. If you need to make changes to the running resources in a stack you update the stack. Before making changes to your resources, you can generate a change set, which is a summary of your proposed changes. Change sets enable you to see how your changes might impact your running resources, especially for critical resources, before implementing them.



AWS CloudFormation creating an entire environment (stack) from one template

You can use a single template to create and update an entire environment or separate templates to manage multiple layers within an environment. This enables templates to be modularized, and also provides a layer of governance that is important to many organizations.

When you create or update a stack in the CloudFormation console, events are displayed, showing the status of the configuration. If an error occurs, by default the stack is rolled back to its previous state. Amazon SNS provides notifications on events. For example, you can use Amazon SNS to track stack creation and deletion progress using email and integrate with other processes programmatically.

AWS CloudFormation makes it easy to organize and deploy a collection of AWS resources, and lets you describe any dependencies or pass in special parameters when the stack is configured.

With CloudFormation templates, you can work with a broad set of AWS services, such as Amazon S3, Auto Scaling, Amazon CloudFront, Amazon DynamoDB, Amazon EC2, Amazon ElastiCache, AWS Elastic Beanstalk, Elastic Load Balancing, IAM, AWS OpsWorks, and Amazon VPC. For the most recent list of supported resources, refer to AWS resource and property types reference.

AWS Serverless Application Model

The AWS Serverless Application Model (AWS SAM) is an open-source framework that you can use to build serverless applications on AWS.

AWS SAM integrates with other AWS services, so creating serverless applications with AWS SAM provides the following benefits:

- **Single-deployment configuration** — AWS SAM makes it easy to organize related components and resources, and operate on a single stack. You can use AWS SAM to share configuration (such as memory and timeouts) between resources, and deploy all related resources together as a single, versioned entity.
- **Extension of AWS CloudFormation** — Because AWS SAM is an extension of AWS CloudFormation, you get the reliable deployment capabilities of AWS CloudFormation. You can define resources by using AWS CloudFormation in your AWS SAM template.
- **Built-in best practices** — You can use AWS SAM to define and deploy your IaC. This makes it possible for you to use and enforce best practices such as code reviews.

AWS Cloud Development Kit (AWS CDK)

The AWS Cloud Development Kit (AWS CDK) is an open source software development framework to model and provision your cloud application resources using familiar programming languages. AWS CDK enables you to model application infrastructure using TypeScript, Python, Java, and .NET.

Developers can leverage their existing Integrated Development Environment (IDE), using tools such as autocomplete and in-line documentation to accelerate development of infrastructure. AWS CDK utilizes AWS CloudFormation in the background to provision resources in a safe, repeatable manner. Constructs are the basic building blocks of CDK code. A construct represents a cloud component and encapsulates everything AWS CloudFormation needs to create the component. The AWS CDK includes the AWS Construct Library, containing constructs representing many AWS services. By combining constructs together, you can quickly and easily create complex architectures for deployment in AWS.

AWS Cloud Development Kit for Kubernetes

AWS Cloud Development Kit for Kubernetes is an open-source software development framework for defining Kubernetes applications using general-purpose programming languages.

Once you have defined your application in a programming language (as of the date of this publication, only Python and TypeScript are supported), cdk8s will convert your application description into pre-Kubernetes YAML. This YAML file can then be consumed by any Kubernetes cluster running anywhere. Because the structure is defined in a programming language, you can use the rich features provided by the programming language. You can use the abstraction feature of the programming language to create your own boilerplate code and reuse it across all of the deployments.

AWS Cloud Development Kit for Terraform

Built on top of the open source JSII library, CDK for Terraform (CDKTF) allows you to write Terraform configurations in your choice of C#, Python, TypeScript, Java, or Go and still benefit from the full ecosystem of Terraform providers and modules. You can import any existing provider or module from the Terraform Registry into your application, and CDKTF will generate resource classes for you to interact with in your target programming language.

With CDKTF, developers can set up their IaC without context switching from their familiar programming language, using the same tooling and syntax to provision infrastructure resources similar to the application business logic. Teams can collaborate in familiar syntax, while still using the power of the Terraform ecosystem and deploying their infrastructure configurations via established Terraform deployment pipelines.

AWS Cloud Control API

AWS Cloud Control API is a new AWS capability that introduces a common set of Create, Read, Update, Delete, and List (CRUDL) APIs to help developers manage their cloud infrastructure in an easy and consistent way. The Cloud Control API common APIs allow developers to uniformly manage the lifecycle of AWS and third-party services.

As a developer, you might prefer to simplify the way you manage the lifecycle of all your resources. You can use Cloud Control API's uniform resource configuration model with a pre-defined format to standardize your cloud resource configuration. In addition, you will benefit from uniform API behavior (response elements and errors) while managing your resources.

For example, you will find it simple to debug errors during CRUDL operations through uniform error codes surfaced by Cloud Control API that are independent of the resources you operate on. Using Cloud Control API, you will also find it simple to configure cross-resource dependencies. You will also no longer require to author and maintain custom code across multiple vendor tools and APIs to use AWS and third-party resources together.

Automation and tooling

Modern operating environments commonly rely on full automation to eliminate manual intervention or access to production environments. This includes all software releasing, machine configuration, operating system patching, troubleshooting, or bug fixing. Many levels of automation practices can be used together to provide a higher level end-to-end automated process.

Automation has the following key benefits:

- Rapid changes
- Improved productivity
- Repeatable configurations
- Reproducible environments
- Elasticity

- Automatic scaling
- Automated testing

Automation is a cornerstone with AWS services and is internally supported in all services, features, and offerings.

AWS OpsWorks

AWS Opsorks takes the principles of DevOps even further than AWS Elastic Beanstalk. It can be considered an application management service rather than simply an application container. AWS OpsWorks provides even more levels of automation, with additional features such as integration with configuration management software (Chef) and application lifecycle management. You can use application lifecycle management to define when resources are set up, configured, deployed, un-deployed, or ended.

For added flexibility AWS OpsWorks has you define your application in configurable stacks. You can also select predefined application stacks. Application stacks contain all the provisioning for AWS resources that your application requires, including application servers, web servers, databases, and load balancers.

Application stacks are organized into architectural layers so that stacks can be maintained independently. Example layers could include web tier, application tier, and database tier. Out of the box, AWS OpsWorks also simplifies setting up AWS Auto Scaling groups and Elastic Load Balancing (ELB) load balancers, further illustrating the DevOps principle of automation. Just like AWS Elastic Beanstalk, AWS OpsWorks supports application versioning, continuous deployment, and infrastructure configuration management.

AWS OpsWorks showing DevOps features and architecture

AWS OpsWorks also supports the DevOps practices of monitoring and logging (covered in the next section). Monitoring support is provided by Amazon CloudWatch. All lifecycle events are logged, and a separate Chef log documents any Chef recipes that are run, along with any exceptions.

AWS Elastic Beanstalk

AWS Elastic Beanstalk is a service to rapidly deploy and scale web applications developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, NGINX, Passenger, and IIS.

Elastic Beanstalk is an abstraction on top of Amazon EC2, Auto Scaling, and simplifies the deployment by giving additional features such as cloning, blue/green deployments, Elastic Beanstalk Command Line Interface (EB CLI) and integration with AWS Toolkit for Visual Studio, Visual Studio Code, Eclipse, and IntelliJ for increase developer productivity.

AWS Proton

AWS Proton enables platform teams to connect and coordinate all the different tools your development teams need for infrastructure provisioning, code deployments, monitoring, and updates. AWS Proton enables automated infrastructure as code provisioning and deployment of serverless and container-based applications.

AWS Proton enables platform teams to define their infrastructure and deployment tools, while providing developers with a self-service experience to get infrastructure and deploy code. Through AWS Proton, platform teams provision shared resources and define application stacks, including CI/ CD pipelines and observability tools. You can then manage which infrastructure and deployment features are available for developers.

AWS Cloud9

AWS Cloud9 is a cloud-based IDE that lets you write, run, and debug your code with just a browser. It includes a code editor, debugger, and terminal. AWS Cloud9 comes prepackaged with essential tools for popular programming languages, including JavaScript, Python, PHP, and more, so you don't need to install files or configure your development machine to start new projects. Because your AWS Cloud9 IDE is cloud-based, you can work on your projects from your office, home, or anywhere using an internet-connected machine.

AWS CloudShell

AWS CloudShell is a browser-based shell that makes it easier to securely manage, explore, and interact with your AWS resources. AWS CloudShell is pre-authenticated with your console

credentials. Common development and operations tools are pre-installed, so there's no need to install or configure software on your local machine.

Amazon CodeGuru

Amazon CodeGuru is a developer tool that provides intelligent recommendations to improve code quality and identify an application's most expensive lines of code. Integrate CodeGuru into your existing software development workflow to automate code reviews during application development and continuously monitor application's performance in production and provide

recommendations and visual clues on how to improve code quality, application performance, and reduce overall cost. CodeGuru has two components:

- **Amazon CodeGuru Reviewer** — Amazon CodeGuru Reviewer is an automated code review service that identifies critical defects and deviation from coding best practices for Java and Python code. It scans the lines of code within a pull request and provides intelligent recommendations based on standards learned from major open-source projects as well as Amazon codebase.
- **Amazon CodeGuru Profiler** — Amazon CodeGuru Profiler analyzes the application runtime profile and provides intelligent recommendations and visualizations that guide developers on how to improve the performance of the most relevant parts of their code.

Credit to: AWS Documentation