Modernize today with containers on AWS

Companies worldwide are undergoing digital transformations. By modernizing their applications, they can deliver better service to customers, and keep pace in a competitive landscape. AWS has helped companies modernize by implementing containers—and initiating cultural shifts—to streamline development. In this eBook, we discuss best practices in containerization and how you can get started today with containers on AWS.

It's about survival

Over the past few years, worldwide digital transformation has caught fire, with overall spending estimated to hit $1.7 trillion by 2019, according to the IDC. Companies of all sizes are finding new ways to leverage technology to boost their agility and better respond to demands from customers. Fueling this fire is the need to survive in a changing environment. These days it's digital or bust. You don't have to look too far to find examples of cloud-native companies disrupting industries while leaving legacy businesses in the dust. For many companies, an initial step toward digital transformation is modernizing their applications and taking advantage of automated environments in the cloud. Modernization empowers companies with:

Elasticity: The ability to respond to spikes in customer demand

Availability: The ability to serve customers' requests wherever and whenever

Agility: The ability to quickly fix a problem or deploy new functionality that customers want

Containers can help get you there

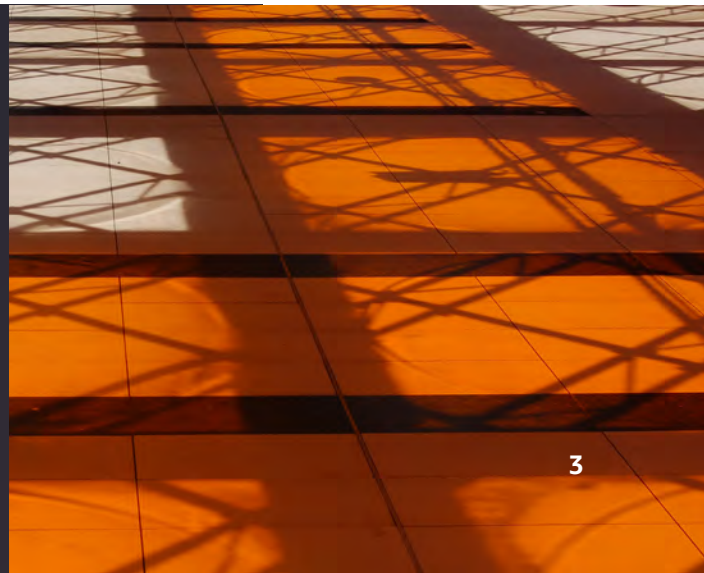
A digital transformation takes time, but in the end, the gains in productivity make the business case. While there are many tools out there to help companies modernize, containers are gaining steam as the go-to solution for developers to more efficiently package and deploy applications. This is evident from the adoption of two different container technologies broadly used today—Docker and Kubernetes. Amazon Elastic Container Service (ECS), which manages Docker containers, has seen more than a 450% increase in active users since 2016. More than 50% of Kubernetes workloads run on AWS. We are now managing containers across millions of instances each month.

Containers streamline development

Containers provide a portable, consistent, and lightweight software environment for applications to easily run and scale anywhere. Throughout its lifecycle, an application will operate in many different environments, whether it's moving from test and production early on, or from on-premises virtual machines to the cloud during a migration. Before containers, IT teams had to consider the compatibility restrictions of each new environment and write additional code to ensure the application would function. Then containers were developed to package the application with its dependencies, configuration files, and interfaces—allowing developers to use a single image that moves seamlessly between different hosts.

For developers, containers allow them to focus on building the application—whether they're adding new features or the latest security—instead of spending time managing the compatibility requirements of different environments.

Containers are also integral to breaking down traditional monolithic application architectures, and they enable a transition to microservices for easier scale. With microservices, each application component runs as its own service, allowing developers to work independently on different aspects.



Containers drive sustainable culture shifts in development practices

Containers are not only a tool to modernize your applications, but they're also the instigators of improvements to your development practices. Containers disrupt the traditional development cycle by driving developers to assume ownership of quality control for the apps and code developed. Where developers used to only be focused on building the application, with containers, now the success of packaging and deploying shifts to them. This phenomenon, known as shift left, ensures quality control is handled upfront, rather than farther down in the development process.

With developers empowered to own the implications of their work, the next step is to create a culture that fosters failing fast and learning from mistakes. Containerization enables this by automating code integration/deployment, thus improving the company's overall agility.

Legacy companies can modernize

A large corporation that's been around for more than 100 years seized an opportunity to modernize its applications in one of its international markets. The company utilized containers and implemented cultural changes to bring its business platform to the cloud. Containers and the automated environment in AWS allowed the company's development team to iterate quickly, pushing the platform from concept to prototype to deployment in just six weeks.

Get started today with containers on AWS

AWS provides all the solutions needed to make containerization seamless. Proven tools for infrastructure provisioning, orchestration, monitoring, security, and automation are readily available today for you to get started with containers.

Provisioning

Seamlessly provision the underlying infrastructure and resources

To run containers, the underlying infrastructure has to be provisioned. AWS offers two different solutions based on the extent of management or automation desired.

- Use Amazon Fargate to automate the provisioning of the underlying infrastructure.
- Use Amazon EC2 instances to define compute, storage, and network capabilities of the infrastructure.

Orchestration

Scale and manage Docker or Kubernetes containers

- Leverage Amazon Elastic Container Registry to store and manage Docker Images.
- Adopt Amazon Elastic Kubernetes Service (EKS) to orchestrate Kubernetes containers.

Security

Secure, scan, and detect vulnerabilities in containers

- Amazon IAM and tagging, security groups for EC2 instances, and Virtual Private Cloud (VPC) enable securing the containers.
- Image scanning solutions detect vulnerabilities of Docker container images.

Automation

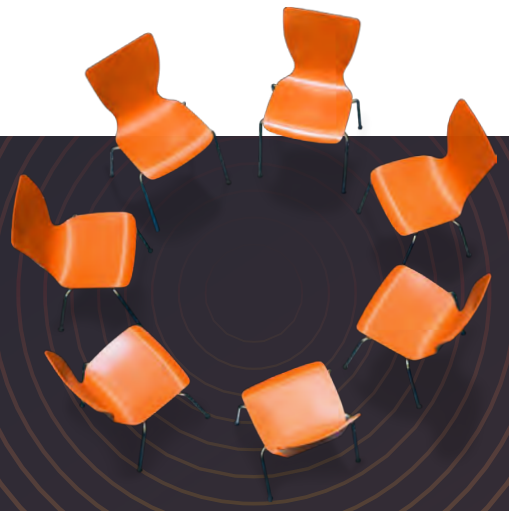
Deploy code automatically with CI/CD

- Create a source code repository using AWS CodeCommit.
- Configure a CI/CD pipeline using AWS CodePipeline.
- Deploy AWS CodeBuild to build your container image.

Observation and monitoring

Ensure services running on containers are healthy and communicate with each other as expected

- Deploy App Mesh to provide visibility into logging, metrics, and tracing and to enable load balancing and traffic shaping.
- Run Healthcheck of Docker container images to confirm your containers are running and your app is working.



Provisioning

AWS provides two tools to provision the infrastructure: EC2 and Fargate. The main difference between the two is the amount of management and control you want over the underlying infrastructure that runs your container applications.

- AWS Fargate: With Fargate, you can run your containers without having to manage servers or clusters. All you have to do is package your application in containers, specify the CPU and memory requirements, define networking and IAM policies, and launch the application.

- Amazon Elastic Compute Cloud (EC2): The more traditional EC2 launch type allows you to bring instances to run containers, providing server-level, more granular control over the infrastructure that runs your container applications.

Orchestration

Once your application has been containerized, the next step is to run the containers in production. To scale out your architecture, you'll want an orchestration tool. AWS offers two orchestration platforms to fit your needs: Amazon Elastic Container Service and Amazon Elastic Kubernetes Service.

Amazon Elastic Container Service

Amazon Elastic Container Service (Amazon ECS) supports Docker containers and allows you to easily run and scale containerized applications on AWS. Amazon ECS eliminates the need for you to install and operate your own container orchestration software, manage and scale a cluster of virtual machines, or schedule containers on those virtual machines.

Amazon ECS provides an infinitely scalable control panel that's managed for you. This kind of orchestration tool works well for companies that use a proprietary operating system or want the ability to control their own infrastructure.

Amazon Elastic Kubernetes Service

Kubernetes is a rapidly growing open-source container management platform built to help you run your containers at scale. When deployed in the cloud, the speed, stability, and scalability of Kubernetes are still affected by the underlying cloud platform. With Amazon Elastic Kubernetes Service (Amazon EKS), the Kubernetes management infrastructure is run for you, across multiple AWS availability zones.

Our customers believe there are tremendous advantages to running Kubernetes on AWS. Based on a 2018 survey from the Cloud Native Computing Foundation, 51% of Kubernetes workloads run on AWS today.

Amazon Elastic Container Registry:

For using Docker containers, first, a Docker image is required. This image acts as a blueprint for creating instances of containers. Amazon Elastic Container Registry (ECR) is a fully managed Docker container registry that makes it easy for you to store, manage, and deploy Docker container images. Amazon ECR is integrated with Amazon Elastic Container Service (ECS), simplifying your development-to-production workflow. Amazon ECR eliminates the need to operate your own container repositories or worry about scaling the underlying infrastructure.

Security

Like quality control, security considerations have also shifted into the earlier stages of the development cycle. With more autonomy over quality control, developers can more readily adapt their code to address the latest security threats. However, for this to occur, security must first be prioritized across an organization. One way to support this cultural change is to make security efforts as transparent as possible and define upfront an architecture that takes security best practices and tools into account.

Amazon Web Services provides multiple tools to protect who has access to your containers: You can use Amazon Identity and Access Management (Amazon IAM) to control who is authenticated (signed in) and authorized (has permissions) to use resources. Amazon Virtual Private Cloud (Amazon VPC) lets you to logically isolate Container Tasks (Docker) or Pods (Kubernetes) in a virtual network that you define. You can define security groups to create a virtual firewall between EC2 instances.

Using an image scanning solution, you can detect vulnerabilities of container images or image dependencies. Security teams can perform scans on these container images or image dependencies, then publish pre-approved resources that developers can consume with confidence.

Automation

The automated environment removes the task of deploying code manually. When your infrastructure includes hundreds or thousands of containers, automating with continuous integration/continuous delivery (CI/CD) allows you to scale and react faster while minimizing the risk of human error. AWS CodeCommit, CodePipeline, and CodeBuild enable you to create a source code repository, configure a CI/CD pipeline, and build your container image, respectively.

Observation and monitoring

Once the microservices are deployed using containers, you need to be able to monitor the health of the containers and ensure that the services are communicating with each other as expected.

- AWS App Mesh makes it easy to run services by providing consistent visibility and network traffic controls for services built across multiple types of compute infrastructure. App Mesh removes the need to update application code to change how monitoring data is collected or traffic is routed between services. App Mesh configures each service to export monitoring data and implements consistent communications control logic across your application. This makes it easy to quickly pinpoint the exact location of errors and automatically reroute network traffic when there are failures or when code changes need to be deployed. App Mesh uses the open-source Envoy proxy, making it compatible with a wide range of AWS partner and open-source tools.

- Docker container images allow you to verify if the container and application are healthy. With a simple health-check command, a Docker file will check a container to confirm that it's still working. This process can detect when a web server is stuck in an infinite loop and unable to handle new connections, even though the server process is still running.

Ready to get the most out of containers?

AWS and our vast partner ecosystem will provide you with the tools you need to get started—no matter what stage you're at.

You can always reach out to an AWS sales associate or talk with your preferred APN partner.

To learn more,
visit: aws.amazon.com/containers