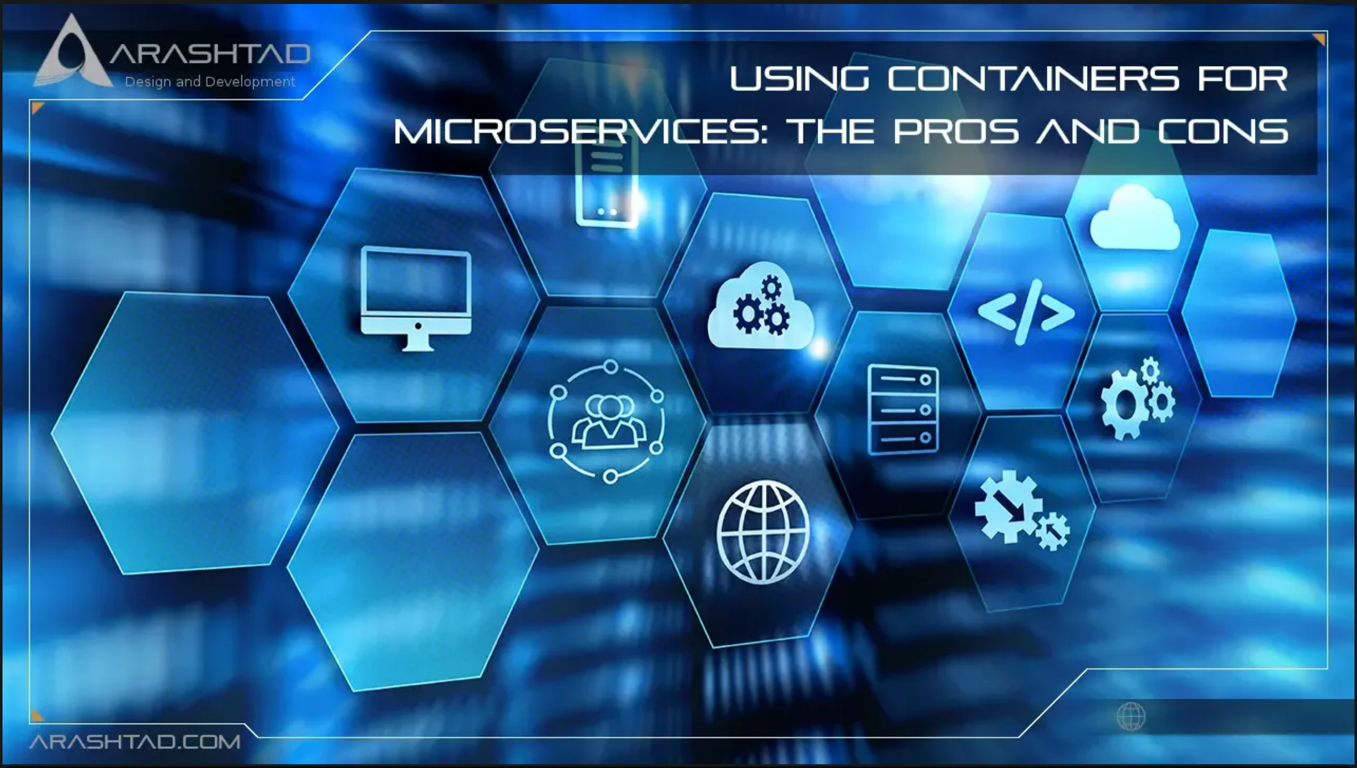
Using Containers for Microservices: The Pros and Cons

No comments



The use of containers for microservices has gained a lot of popularity over the last decade. By developing an application using microservices in multiple containers, you can combine the best of both worlds. Through scaling and improvements, it provides resilience as well as agility. Let's start by getting to know microservices and containers first before we discuss how they make an ideal combination.

What Are the Microservices and Containers?

An architectural paradigm known as microservices focuses on constructing single-function modules with well-defined interfaces and operations to develop software systems. Containerization is the process of packaging software code, including libraries, runtimes, frameworks, and other dependencies, and is isolated in their own "container". We will discuss in this article the advantages of using containers along with microservices and some of the challenges associated with using them.

Various Options for Microservices Deployment

Among the options available for hosting your microservices' computing resources, containers are not the only option. Here are some other options:

1. Virtual Machines (VM)

In general, it's not recommended to host microservices in virtual machines. If you deploy to one VM, it will be a single point of failure. If you deploy to multiple VMs, you will need to connect them. In place of microservices, VMs are a better choice for deploying monolithic applications.

2. Serverless Functions

An isolated environment runs code that responds to triggers, such as user login requests.

Containers for Microservices: The Best Option

VMs are known to be slower and heavier than containers. However, that's not the only reason why containers for microservices are best suited. Here are some of the awesome benefits of using containers.

Better Performance and Low Infrastructure Footprint

When microservices are deployed in containers instead of virtual machines, it's easier to take advantage of their agility. Virtual machines can take a while to start, but containers are lightweight and can start immediately.

Higher Security

The container provides better isolation for each containerized microservice. Each container has its own attack surface and is isolated from the other microservices. Containers ensure that security vulnerabilities in one container do not leak into another. As opposed to microservices deployed directly on host OSes or virtual machines, which are less secure.

Ease of Use

In addition to making developers' lives easier, containerized microservices allow them to work on their specific tasks rather than having to get involved in the overall application's complexity. Microservices are relatively small and self-contained components, so they allow them to concentrate on their own specific tasks. Moreover, containerized applications allow them to develop each service in the language that suits its needs.

Service Discovery

Service discovery is a crucial aspect of any SOA-based architecture. Microservices can be localized and interconnected much more easily if they are hosted in containers. Each host may have a different networking configuration if you deploy microservices on virtual machines. This makes designing a network architecture that supports reliable service discovery challenging.

Better Orchestration

Using containers on a shared platform, microservices can be orchestrated, scheduled, started, stopped, and restarted more easily.

A Greater variety of Tools

Container-based tools for supporting microservices have matured significantly in the past few years. Orchestration platforms for containers, such as Amazon ECS and Amazon EKS (Kubernetes), have gained popularity and are well supported by the community today. On the other hand, there are few tools to orchestrate microservices hosted in unikernels or virtual machines.

Image not found or type unknown

Using Containers for Microservices: The Pros and Cons

The Challenges of Using Containers for Microservices?

In spite of the many benefits of using containers for microservices, there are still some challenges. Below is a summary of some of those challenges:

Higher Complexity for Your Workforce

Containers introduce a layer of abstraction, adding complexity to management, monitoring, debugging, etc. Many developers struggle to deal with these abstractions, especially when implementing microservices in large applications. A container runs on a dynamic infrastructure, which means it boots up and shuts down continuously based on its load. This can pose a challenge for deployment, management, and monitoring. The complexity will also increase if microservices are written in different languages.

Learning Curve

Dockers, Kubernetes, or other similar tools must be familiar to developers if container orchestration is to be achieved. Your servers must also be able to handle different container runtimes, as well as the network and storage resources that each requires. Using managed container services such as AWS ECS and AWS EKS can help reduce this learning curve.

Persistent Data Storagec

Due to the ephemeral nature of containers, you will need a way to store data outside of them.

Tips of Using Containers for Microservices

In order to containerize your microservices successfully, it is essential to plan and evaluate the following factors:

Container runtime

In addition to the container runtime, you should also consider the full set of configuration management tools for containerized microservices. Instead of simply deploying the container runtime on its own, containerized microservices are easier to manage. Among common containers, runtimes are run, Docker, Windows Containers, etc. Regardless of the container runtime you use, ensure it adheres to the specifications set by the Open Container Initiative (OCI).

Plan for External Storage

Your application must implement an external storage mechanism since container data usually disappears once the instance is closed. Many orchestration tools come with data storage capabilities. Consider the features and attributes of each data storage tool when comparing data storage solutions. AWS ECS supports many types of persistent volumes, such as EFS, FSX for Windows, and Docker volumes.

Service Orchestration

If you're working with a huge number of containers, you'll need orchestration tools to automate operational tasks, such as load balancing, shared storage, etc. The Kubernetes container orchestrator is the de facto choice if your application runs on Dockers, but container management platforms are also available for specialized applications. AWS ECS and EKS feature enterprise-level features such as workflow automation and integrated continuous integration/continuous delivery pipelines.

Networking and Communication

The application design should consider networking and communication issues that may arise when microservices are deployed independently in their own containers. microservices will still need to communicate with each other. As part of this process, several tools and services are used, including load balancers (ALB and NLB), API gateways, and VPCs.

Security

A containerized microservice is generally more secure than a monolithic VM-based application due to its reduced attack surface. However, microservices often require access to back-end resources. In privileged mode, containers can access the host's root capabilities, exposing the kernel and other sensitive system components.

Furthermore, it is essential to implement audit tools that verify that container configurations meet security requirements as well as container image scanners that automatically detect potential security threats in addition to solid IAM practices, security groups, good network policies, and security context definitions.

Conclusion

Our article discussed the pros and cons of using containers for microservices. Microservices are a great option for growing organizations. Containers are the best way to host them. As a result of our discussion, we also discussed the challenges associated with containerized microservices. These challenges are faced by every company implementing microservices for complex and multi-component systems. With a modern solution like Query, containerizing microservices offers the benefits mentioned above without the complexities.

Join Arashtad Community

Follow Arashtad on Social Media

We provide variety of content, products, services, tools, tutorials, etc. Each social profile according to its features and purpose can cover only one or few parts of our updates. We can not upload our videos on SoundCloud or provide our eBooks on Youtube. So, for not missing any high quality original content that we provide on various social networks, make sure you follow us on as many social networks as you're active in. You can find out Arashtad's profiles on different social media services.

Get Even Closer!

Did you know that only one universal Arashtad account makes you able to log into all Arashtad network at once? Creating an Arashtad account is free. Why not to try it? Also, we have regular updates on our newsletter and feed entries. Use all these benefitial free features to get more involved with the community and enjoy the many products, services, tools, tutorials, etc. that we provide frequently.

SIGN UP NEWSLETTER RSS FEED