Container First Approach for Micro, Small and Medium Enterprises - An Overview

Nishi Tiku[1], Sharan Iyengar[2], Kapil Kumar Trivedi[3]

Mumbai University, Vivekanand Education Society's Institute of Technology, Collector's Colony, Chembur, Mumbai, Maharashtra, India

Abstract: *Distributed architecture no longer means having multiple servers for different services, the world had moved on to micro services. Containerization has given the power in the hands of the developers to spin up images for micro services as and when required. The architecture of containers divides applications into distributed containers, which offer the flexibility to place them on different physical and virtual machines. System admin now just have the responsibility of keeping the server running with any container orchestrator. Cloud providers have realized these needs and have started offering similar services, which take away the mundane burden and emancipating time for development.*

Keywords: Container, Cloud, DevOps, Virtual Machine, Server, Cloud Service Provider, Containers as Services

1. Introduction

With the advancement in the cloud services, we have seen a spike in the use of containers from all the IT Professionals. By 2018 it was seen that nearly one quarter of companies have adopted one or other form of containerization tool. Developers and Administrators have more inclination toward containers as they are self-contained and resource isolated applications which could be deployed without the interference of the Admin and requiring individual servers for each project. Given the benefits container technology brings to the table, it is quite impossible for the companies to not fairly investigate it and adopt it. It is assumed that more than 25% of the companies have accepted container technology. With the rise in them, various cloud service provides have started offering containers as service to customers with various functionalities and price points.

2. Literature Survey

Ravi Kumar in his paper "An Introduction to Dockers: From Monolithic toContainerized Environment"[4] explained that how the IT industry was following a monolithic based application approach in 1990's and then moved on to virtualization to scale down expenses and shorten the production life cycle. Now IT industry is moving towards its next milestone Containerization and continues to explain about the features and benefits of Containers.

You would find that Priyanka P. Kukade and Prof. Geetanjali Kale in their article, "Auto-Scaling of Micro-Services Using Containerization" [5] have expressed about the changing style of development and deployment. Applications are now being designed in a loosely coupled manner, so that there can be a plug and play type of architecture and how containers or container-based applications can help in achieving this with optimized usage of resources in cloud. Orchestration tool will help us improve the efficiency and simplify the management process.

When we talk about the IT Industry you would come across many small organizations trying to make it big. For these organizations to survive, finance plays an important role.

These organization use their personal savings or borrow from financial institutions to start and survive. Keeping the expenses minimal while trying to run business becomes difficult with this kind of capital. In the article "Why Do Startups Fail? A Case Study Based Empirical Analysis in Bangalore."[11], Kalyanasundaramand Ganesaraman has talked about the various reasons of why a startup fails. For these organizations cutting cost can help them survive in the market from an IT stand point container can help them cut down on infrastructure cost.
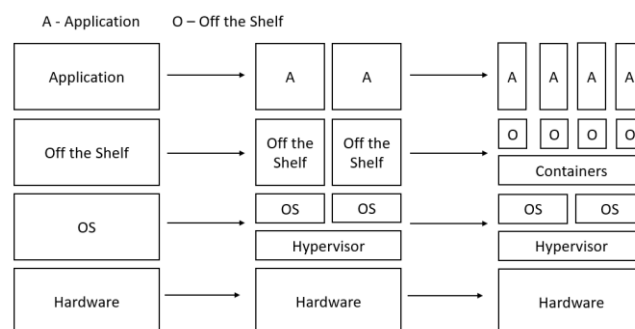

Figure 1: Evolution of Infrastructure

3. Container

When kernel allows the existence of multiple isolated user-environment to co-exist at operating system level, such isolated user-environments are called containers. The origin of this technology can be traced from '*chroot*', '*cgroups*' and '*namespaces*', that enables us to create containers. '*cgroup*' control groups, a Linux kernel feature that limits, accounts for, and isolates the resource usage like CPU, memory, disk I/O, network, etc.of a collection of processes [1].

All flavors of Unix OS have root directory (/) as the top directory.All file system entries branch out of this root. This is the system's actual root.We can havedifferent roots so that we can have separate environments to run, which will make it easier to run and debug the processes. '*chroot*' changes apparent root directory for current running process and its children [2].

Figure 3: Transformation of Infrastructure

In Linux, Namespaces isolates kernel resources such that processes cannot interfere with each other [3]. All these technologies together provide Linux developers means to isolate processes into their own 'containers'. This allows developers to move from monolithic applications to distributed micro services, which helps develop agile, flexible and scalable cloud-ready applications. This gave rise to a new MVC paradigm called Plug and play architecture, where components are loosely coupled and work as a collection work flow. Advantage of such modular architectures is that you can add, remove or replace flows without disturbing the rest of the application. This gave developers the power to spin-up their own container with consistent environments without the help of sys-admins, thus the term "power of the developer".

Figure 2: Container Architecture

It was predicted by many organizations that there would be great shift from on premises computing to cloud computing, we can now see that prediction coming true with organizations collaborating or having mergers and acquisitions with cloud providers. It is now possible because it is quite affordable to have your own infrastructure in cloud than on premise. With the benefits that containers provide to developers, the cloud providers are providing containers as a service.

4. MSME the Current Scenario

For any Small or Medium Enterprise (SME), proper utilization of resources is a major concern. Given the trends and transformation in all the industries due to digitization, having an IT infrastructure has become a must for survival, whether you outsource it or maintain it inhouse.

Setting up an IT infrastructure is comparatively less expensive now as compared to a decade ago, but it is quite an investment for any small organization. Its not just the cost of equipment but also the cost of various other amenities like physical space, maintenance and people with the recursive cost of managing them.

Being said that one would opt for either a cloud-based solution where the infrastructure is not the organization's concern but of the cloud vendor or setting up an inhouse infrastructure.

Cloud based solution are of various types and can be used as per need. Various cloud services offered by cloud vendors are SaaS, PaaS, and IaaS.

Benefits of Cloud Based Solution

- Reduced IT Cost
- Scalability
- Efficiency
- Pay per usage
- Increased Collaboration

When an origination spends its resources to establish an IT infrastructurecontrolled and managed by them then this type of IT infrastructure can be referred as Inhouse IT infrastructure.

Benefits of Inhouse IT infrastructure

- Total Control on physical infrastructure
- Sensitive data stays inside the organization
- Availability

You can run your application server in your own premises or use cloud service to run them. It doesn't matter where you are running the server, it is a tricky job to scale and maintain them as per your needs, even if you use virtualization software. Given the trends of development cycle moving towards the agile methodology which requires continuous integration of new features and continuous deployment (CI/CD) with near to zero down time. To better manage this situation Containerization of applications can help.

5. MSME Container First Approach

Containerization of applications brings a lot of benefits to the table:

- Portability on cloud as well as various platforms.
- Efficient in term of resource usage than VMs.
- Better Utilization of compute resources.
- Integration with existing DevOps environment is easy.
- Better throughput or delivery rate of the development team.
- Helps create containerized monolithic applications using microservices which helps the development teams to create modules with its own life cycle.
- Better security provided by separating applications from their host systems and from each other.
- Faster app start-up and easier scaling.
- Easier maintenance with the help of various opensource orchestration tools.

These benefits make containerization the go-to option. Let's say, we were to host an application with 5 instances. In case of bare metal hardware situation, we would require 5

different physical machines with its own copy of operating system and the overhead of setting up each of the machines and maintaining them. The other overhead we would face would be power consumption. This scenario is better when we replace bare metal hardware with VM.

If we host these in VM, we do not have the overhead of managing multiple hardware devices, since we run all the 5 applications virtually on a single physical machine. But we still need 5 OS copies and must make sure that the single machine is powerful enough to manage the load of all the five application.

Now, imagine that if we could run the same application only using a single hardware device and single OS with 5 application containers running. The cost that we are saving is quite large and we haven't considered the possibilities of clustering.

Containerization does a better job of consolidation than hardware virtualization, because of less duplication and therefore less resource consumption. We have lesser operating systems to manage and worry about the frequent security patches.

Major providers of Cloud Container are:

1) AWS Cloud Container Services
2) Azure Cloud Container Services
3) Google Cloud Container Services

6. Conclusion

We can see that there are a lot of benefits of going with containers, be it mid-stage or from the get-go. Having said so, we feel that the container technology is still in its nascent stages and there are lot more changes to come. So maybe the largest of applications may not be able to go on containers yet, but if we start adapting now then we will be ready for the upcoming digital revolution.

Reference

[1] Pooja Jaywant Patil, "System for Secure Storage and Auditing of Cloud Data", International Journal of Science and Research (IJSR), Volume 4 Issue 1, January 2015, 1387 – 1390.
[2] Yirgalem Tadele Gerba; P. Viswanadham, "Small Scale Enterprise Finance Sources and Constraints in Ethiopia, Case Study of Addis Ababa", International Journal of Science and Research (IJSR), Volume 5 Issue 4, April 2016, 1426 – 1430.
[3] Nishigandha Shendkar, "Container Based Intrusion Detection System in Multitier Web Applications", International Journal of Science and Research (IJSR), Volume 5 Issue 12, December 2016, 440 – 445.
[4] Ravi Kumar, "An Introduction to Dockers: From Monolithic to Containerized Environment", International Journal of Science and Research (IJSR), Volume 5 Issue 12, December 2016, 457 – 458.
[5] Priyanka P. Kukade; Prof. Geetanjali Kale, "Auto-Scaling of Micro-Services Using Containerization", International Journal of Science and Research (IJSR), Volume 4 Issue 9, September 2015, 1960 – 196.
[6] Carlos Arango; Remy Dernat; John Sanabria, "Performance Evaluation of Container-based Virtualization for High Performance Computing Environments", arXiv:1709.10140v1 [cs.OS] 28 Sep 2017
[7] "Cgroup" Wikipedia. [Online].https://en.wikipedia.org/wiki/Cgroups
[8] "ChrootCgroupsand NamespacesAn Overview". [Online]https://itnext.io/chroot-cgroups-and-namespaces-an-overview-37124d995e3d
[9] "Monolithic_application" Wikipedia. [Online].https://en.wikipedia.org/wiki/Monolithic_application
[10] "The Benefitsof Containerization" IBM. [Online] https://www.ibm.com/cloud/blog/the-benefits-of-containerization-and-what-it-means-for-you
[11] Kalyanasundaram, Ganesaraman. (2018). Why Do Startups Fail? A Case Study Based Empirical Analysis in Bangalore. 10.7545/ajip.2018.7.1.079.