A Whitepaper on

Containerization

VERITIS

INDEX

1.Introduction

We are in a digital world, where business can happen over smart devices, cloud platforms, shareable platforms and more.

IT industry, in specific, has seen a wide range of transformation in business functioning that has gradually moved from being ‘organization-specific to user-specific’.

Thanks to the digital trend that brought about the application-centric business transformation, making the user the decision maker.

As a result, every firm in the race is considering early customer reach-out as the means to achieving success.

The trend demands high-speed delivery, but quality and security can’t be compromised! How do you achieve that?

The answer is **‘Containerization’!**

This whitepaper speaks in detail about Containerization and its significance to today’s IT industry in achieving business continuity.

2. An Overview of Containerization

2.1 Defining Containerization

Definition 1: Containerization refers to the process of application packaging into different container packages to run multiple isolated applications at every instance.

Definition 2: Containerization is also defined as the OS-level virtualization method that runs distributed applications with their own set of configurations and resources, without launching the entire Virtual Machine (VM).

Definition 3: Containerization is also a method of encapsulating apps within a container with dedicated resources, dependencies, configuration files and libraries i.e., referred to as 'Application Containerization'.

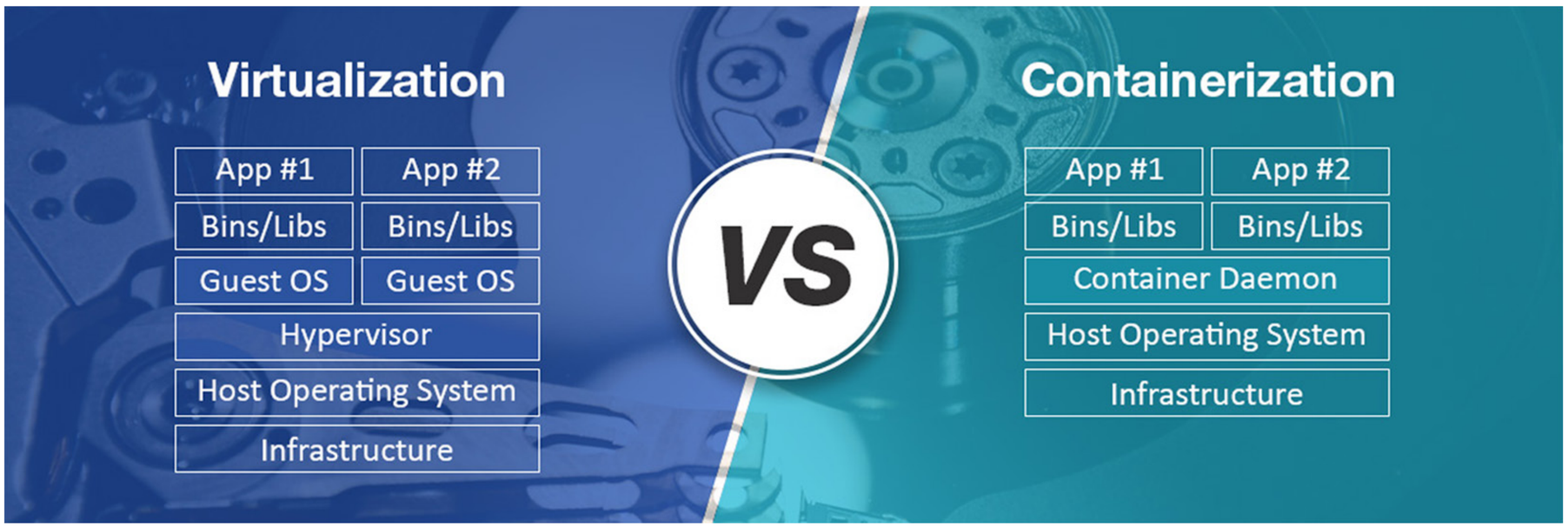
2.2 Containerization vs Virtualization

With the underlying concept of easing application delivery, both Containerization and Virtualization aim to facilitate delivery of multiple applications with minimum resources.

Virtualization refers to the process of ‘Server Consolidation’, that deals with the top of the server creating separate Virtual Machines (VM) and OS resources for every single application.

Whereas, Containerization works at the hypervisor level enabling ‘OS Consolidation’ to make applications run independently in separate containers with their OS and other own set of configurations and resources, without the need to launch an entire VM.

Because of this reason, Containerization is often termed as ‘a replacement to Virtualization’ or ‘advanced virtualization’.



2.3 Containers over Legacy Infrastructure

Executing applications in legacy IT infrastructure was mostly about moving around bare metal servers and further on VMs.

That's where Container infrastructure made a difference through effective utilization of hardware and resources.

Container environment operates at the hypervisor level, between host server (VM or a bare metal server) and application.

Abstracting applications from the host server and reducing dependencies on the same, Containerization enables high runtime for individual applicationsthrough simplified configuration.

Thus, applications run faster in their own environment without depending on the main server, with minimum configuration variables and maximum scalability.

2.4 Three Phases of Containers Evolution

The evolution of container technology services can be explained in three different phases namely:

1 **Staging and PaaS**

Way back in 2013-15, the use cases for container services were limited to hosting production applications under Platform-as-a-Service (PaaS) system and building agile staging environments for automated release pipelines. Containerization frameworks were new to the IT industry and containers services were popular for facilitating orchestration procedures. This period is often termed as the period of container runtimes.

2 **Deployment at Scale**

Orchestrators started expanding in the container ecosystem during 2015-18. As the firms began orchestrating containers at scale, ‘containers for application deployment’ emerged as the new trend taking containers beyond staging and tightly controlled PaaS limitations. Kubernetes and Swarm were the popular orchestrators.

3 **Takeover**

Containers started witnessing demand as the most-sought technology solution, from 2018 and beyond. Kubernetes and Docker emerged as the leading orchestration tools; Azure, AWS and Amazon ECS became popular for container deployment platforms; and more what we see today made their entry.

2.5 Types of Containers

By use-cases, Containers are basically categorized into two types namely, System Containers and Application Containers.

1 System Containers

Any container that runs an OS is called a System Container or OS Container. Similar to VMs, System Container shares kernel of the host OS and facilitates user space isolation. Alike typical OS, one can install, configure and run different applications, libraries, etc., on OS Containers. They allow multiple processes at a time, on a single OS and no guest OS. Here, services running in containers can only use resources limited to that specific container.

2 Application Containers

Application Container is the other container type that undertakes ‘application packaging’ without launching VM for each app or its services. Allowing separate container for each app component, these containers allow distribution, greater control, security and process restriction. Application packaging method is known to reduce compatibility, inconsistency and unreliability issues, significantly. Because of this reason, Application Containers work well with Microservices architecture.

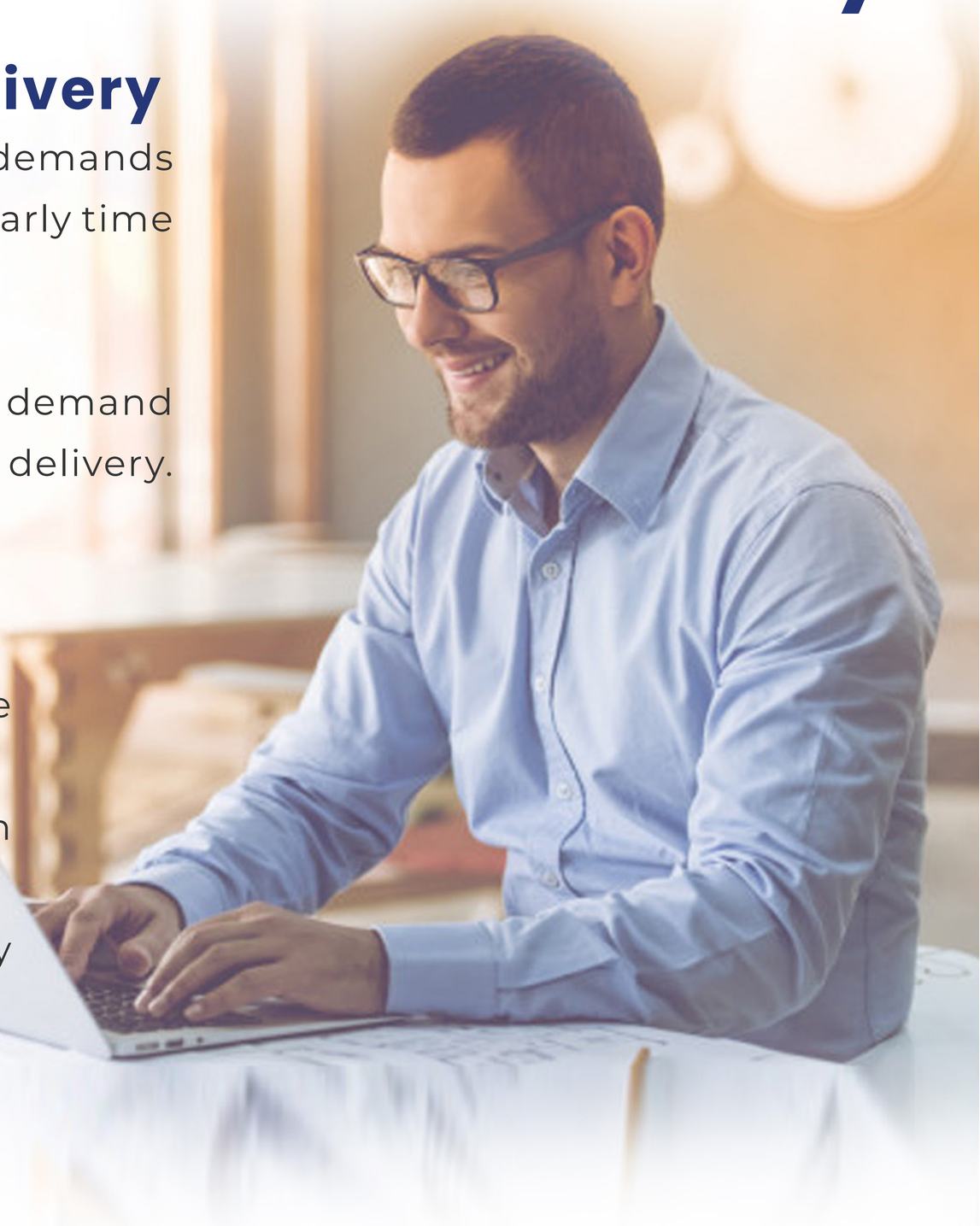
3. Containerization for Business Continuity

3.1 A Way to Enhance Application Delivery

Winning the race in the fast-paced competitive IT market demands high-speed product/application delivery mechanism and early time to market.

This increasing need to speed-up delivery cycles led to the demand for Containerization, as a means to enhancing application delivery.

- Not able to delivery applications on time?
- Facing challenges with application delivery cycle?
- Struggling to develop platform-independent portable applications?
- Looking for maximum application productivity with minimum resources?
- Not able to meet Continous Delivery cycles and early time-to-market demands?



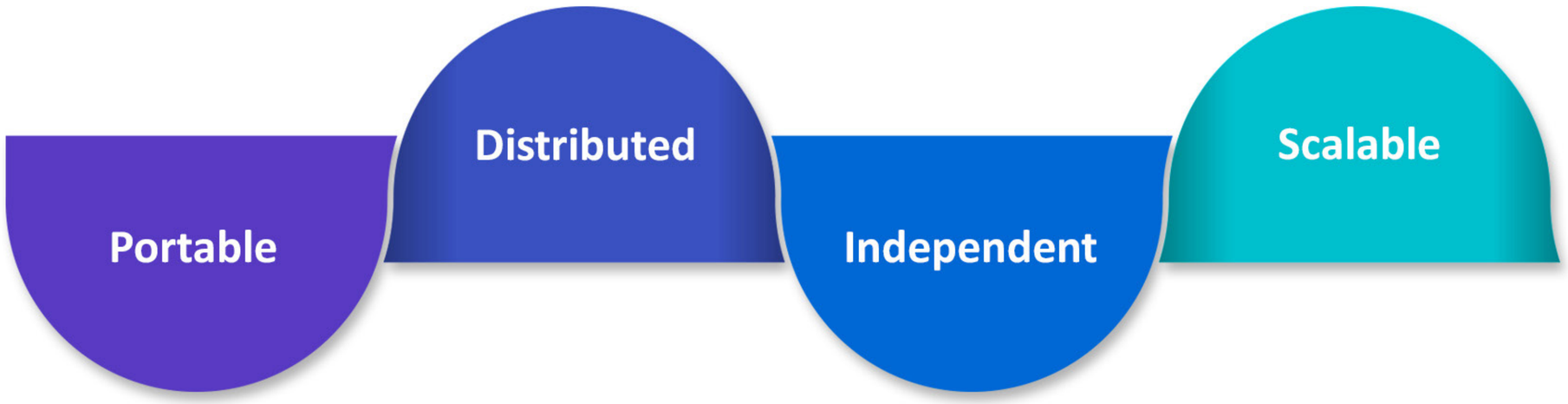
Containerization has answers for all these concerns!

- **Less Release Overhead**
  A Container environment creates scope for breaking down a single large application into multiple components, which are further packaged into separate containers. Convenient application packaging makes apps isolated. While, dividing them into multiple components enables multiple app instances, cutting down the ‘release overhead’.

- **Continuity**
  Updates and any changes or additions to a specific application will not interrupt the functioning of other container applications in the chain, thus ensuring continuity.

- **Less Security Risks**
  Unlike VMs, container applications live shorter, which minimizes the risk of outdated packages. Owing to their shorter timeframes, container apps can easily be rebuilt and re-deployed to production. Any security update or changes to a specific application will not have impact on others, as containers work isolated.

Thus, Containerization makes your applications:



3.2 Container Tools That Revolutionize

There are a wide variety of container tools and resources that revolutionize application delivery cycle. These include Container Tools by Runtime, Cluster Management and Deployment, Storage, Security and Operating System.

Container Tools and Resources

3.3 Containers vs VMs: A Security Perspective

Containerization is termed as ‘advanced virtualization’, which clearly says that containers enjoy some dominance over traditional VMs.

A container environment is known for its abilities of high delivery speeds, effective resource utilization and production efficiency in driving applications.

However, ‘Security’ continues to be the most common debate between the two.

Security in VM

A Virtual Machine (VM) is known for its adaptability to operating environments that require strong segmentation boundary, for example, a multi-tenant cloud.

In VMs, boundary segmentation lies between ‘virtual hosts’, and between ‘VMs and hypervisor’. This level of segmentation offers scope to make changes in underlying configurations through micro segmentation.

This is also of high advantage while migrating from a physical to virtual environment.

However, creation of undesired images on a larger scale and outdated images on a disk are some serious challenges with traditional VMs.

Security in Containers

Application isolation is one of the key differences that a container environment makes over traditional VMs.

Isolating apps as different container packages limits the chances of deeper penetration of malicious programs and risk vulnerability to apps on board.

Moreover, every container package possesses their own level of security, individually.

The short lifespan of container apps makes them less vulnerable to outdated scenarios. Container engine's host OS gets updated automatically.

While workloads in VMs need manual intervention, Containers perform on CI/CD cycles where they remain updated all the time with regular releases and timely patches.

3.4 Containers in DevOps Ecosystem

Looking at the true nature of DevOps, it's needless to say that 'better the system design and operating environment, better will be the DevOps implementation'.

DevOps practices were found to generate maximum efficiency while operating on flexible systems that support continuous integration and continuous delivery cycles.

Containers need a special mention here owing to their abilities in facilitating 'Continous Delivery (CD)' mechanism.

Container Technology's core concept of 'packaging and shipping software along with dependencies' is the major supporting factor for DevOps cycles. This style of packaging makes applications faster, effective and platform independent.

App packaging also makes it easier for developers in making changes or updates to any application in the production chain. Addition to one application will not disturb the others on board.

Another significant factor is ‘Consistency’ that is easily achieved in a Container environment, as the container apps live short and take less shift from the actual state. Writing, testing or deployment of an application will not affect the entire environment of delivery chain. This also bridges collaboration gaps among teams.

Since the container apps are scalable, another key feature of DevOps culture i.e. ‘Agility’ is also achieved.

Thus, the core objectives of DevOps Continuous Integration, Continous Delivery, enhance Collaboration and increased Agility can be achieved easily through Containers.

3.5 Containers and Microservices

Containers and Microservices are different in approach but often used in combination and share some similarities in making developers work easier.

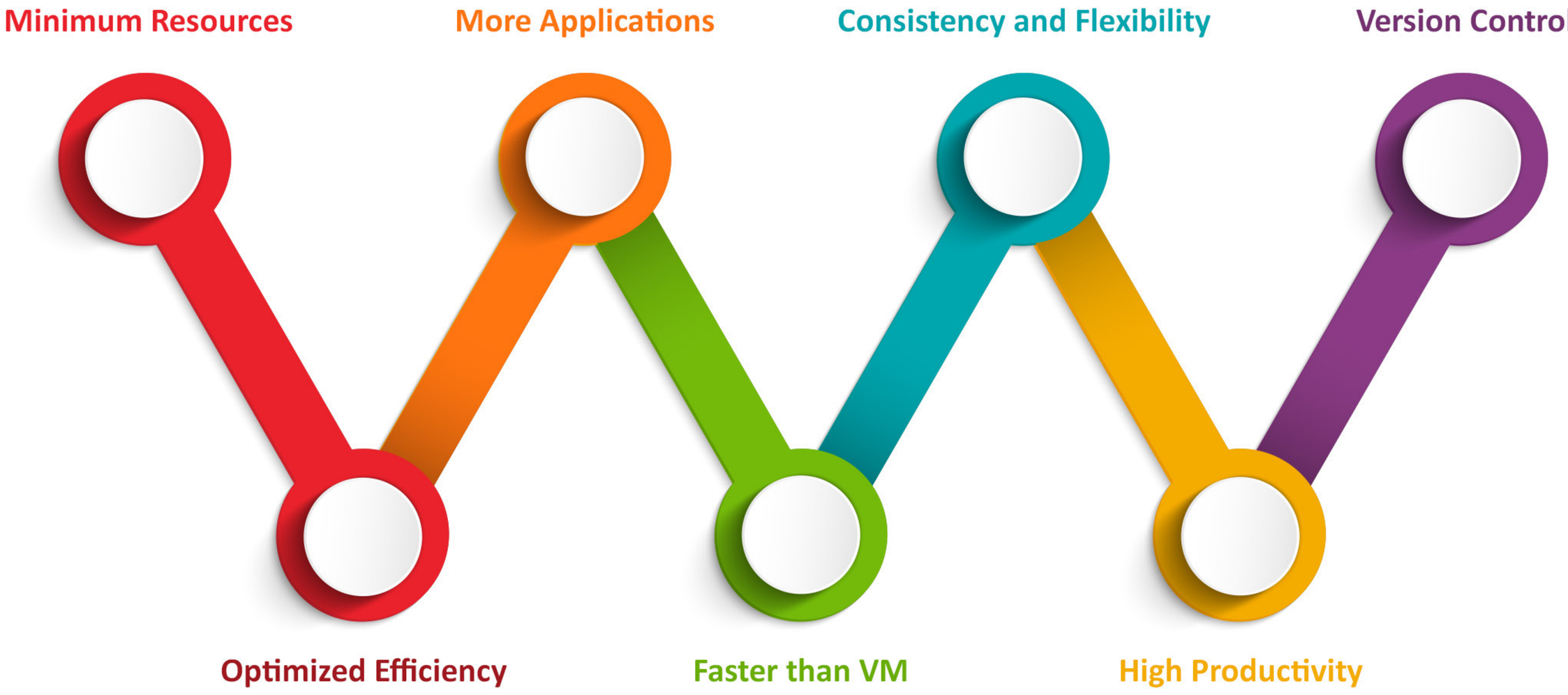
Here are some salient features of Containers and Microservices, individually and together:

Containers	Microservices
Lightweight application packaging units designed to run apps anywhere	Simplified approach to app development, where a large app is treated as a set of modular components or services
Containers manage application packaging in a way that all apps contain their own configurations, dependencies, etc, thus making the apps isolated	A microservice architecture structures an application as a collection of loosely coupled services that are independently deployable and contain lightweight protocols
Containers encapsulate individual components of the application logic and contain only the minimal resources specific to that particular task	A microservice architecture includes everything such as OS, platform, framework, runtime and dependencies
Containers are mostly opensource, downloadable	Autonomous, independent and easy to monitor

Containers	Microservices
Pre-built components can be used to build up application images	Launching multiple instances of application server can be replaced with scaling out a specific microservice on-demand
Allows overwriting and addition of separate container layer even while applications are running	Faulty services can be modified without impacting other components
Minimized downtime and better business continuity	Better value from underlying infrastructure
A container can be the best convenient way to develop and deploy a microservice	A microservice can be run in a container but might not necessarily need a container to run

3.6 Key Benefits of Containerization

Overall, Containerization has a wide range of benefits for your IT business, summarized as:



A technology Container exactly does the same that a general industry container does by segregating loads into dedicated boxes/packages, and is best suitable for today's business scenario that demands continuous delivery and quicker time-to-market.

4. Containerization Market Overview

This section gives a brief about the Containers Technology market, by Growing Adoption and by Performance Insights.

By Growing Adoption

Because of its abilities in boosting app delivery mechanism, the container technology services have been gaining some notable adoption since the recent past and is now skyrocketing towards the future.

The 2019 Container Adoption Survey indicates a ‘growing confidence among IT leaders’ on container services usage for business-critical applications.

Here are some excerpts from the survey:

- A significant rise in percentage of enterprises running Container technologies from 55 percent in 2017 to 87 percent to current
- Container usage in production increased to 90 percent, a 23 percent rise over 2017
- 24 percent of the respondents spent USD 500k on container technology annually, while 17% spent USD 1 million
- The application container market is expected to progress with a CAGR of 31.8 percent towards 2025
- The container orchestration market is expected to witness a CAGR of 17.9 percent towards 2023
- The containers technology market is expected to touch a USD 3 billion revenue by 2020

By Performance Insights

Here are the performance insights reported by various surveys and organizations, after applying Container Technology to their service portfolio:

Besides high-performance benefits, Security & Compliance, Performance, Complexity, Personnel skillset and Availability (9%) continue to be among key challenges in containers implementation.

5. The Conclusion

After witnessing the extensive benefits that it offers, Containerization becomes imperative for any organization looking for the competitive market edge.

If you are a large enterprise with well-equipped cloud infrastructure, succesful DevOps culture and leading digital capabilities, addition of containers to your product development cycle can get you faster results.

If you are a small business aiming to gain some market presence, then containers work for you as well because of their features such as minimum resource usage and maximum productivity saving costs.

Wait no more! Get Containerization and witness the new app speeds!

info@veritis.com

1-877-VERITIS (283-7484)

972-753-0033

US Corporate Headquarters

1231 Greenway Drive

Suite 1040

Irving, TX 75038

India Headquarters

#607, 6th Floor,

Ashoka Bhoopal Chambers,

S P Road, Begumpet

Hyderabad – 500003, India.

Phone no: 040 42211730

For more information, contact info@veritis.com