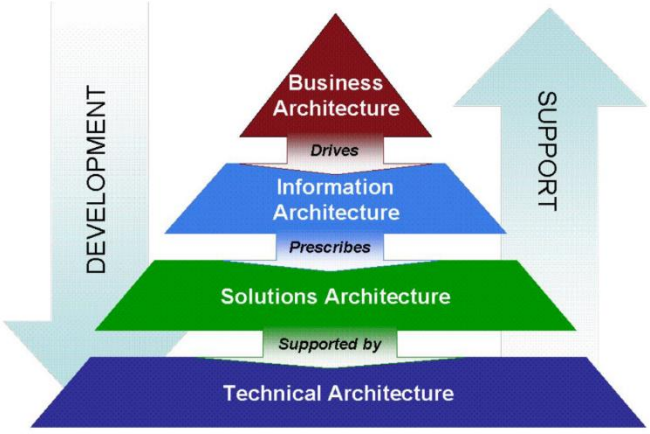
Virginia Information Technologies Agency

Enterprise Architecture Technical Brief

Virtualization and Containers

Robert Kowalke
January 2019



Summary

Virtualization and containerization represent strategic, cloud enabling technologies that increase infrastructure efficiencies, while enabling positive IT experiences for agency staff, customers, and residents.

This technical brief defines the meaning of virtualization and containerization technologies; identifies the pro's and con's of these technologies, and in some measure identifying how they may be exploited by various agency enterprises.

Guidance from this technical brief is intended to help commonwealth agencies determine which, or both technologies will be useful for them as they transition to cloud environments. This document also provides VITA guidance on the cooperative use of these technologies.

A recent catalyst towards achieving higher consumption of virtualization technology throughout the commonwealth, is Governor Ralph S. Northam's September 17, 2018 executive order nineteen, which directs use of cloud technologies for Virginia IT services. [1]

Commonwealth of Virginia
Office of Governor Ralph S. Northam

For any comments, questions, and/or concerns with this technical brief, please contact VITA EA: ea@vita.virginia.gov

[1] Executive Order Nineteen (19) – Cloud Service Utilization and Readiness. Commonwealth of Virginia Office of the Governor on September 17, 2018.

Contents

VITA Virtualization and Containerization Recommendations

1. VITA recommends and historically supports virtualization.
    i. 50% of the current VITA enterprise is virtualized, with plans to reach higher percentages where appropriate.
    ii. Virtualization is good for:
        1. Running a large number of services, or users because each service is variable in terms of CPU requirements
        2. Where there are many changes such as adding new applications or when sizing is adjusted often.
        3. If you have multiple applications with varying characteristics requiring a secure environment, remain on VMs.
        4. Web servers that serve thousands of users.

2. Further, VITA recommends use of containers, which are a form of virtualization, for use cases such as:
    a. Faster startup and spin-downs are important.
    b. Continuous integration (agile programming).
    c. Services being migrated from on-premise to cloud technologies. [2]
    d. For heavy development, test, or integration environment.
    e. Offering services in the cloud through which container standardization is helpful.

    Note that existing IT infrastructure optimized over a period of many years for virtualized business applications may not efficiently support containers.

[2] Commonwealth of Virginia (COV), Information Technology Resource Management (ITRM) Enterprise Architecture Standard EA225 of 2017.

Virtualization and Containerization Research

Virtualization Research

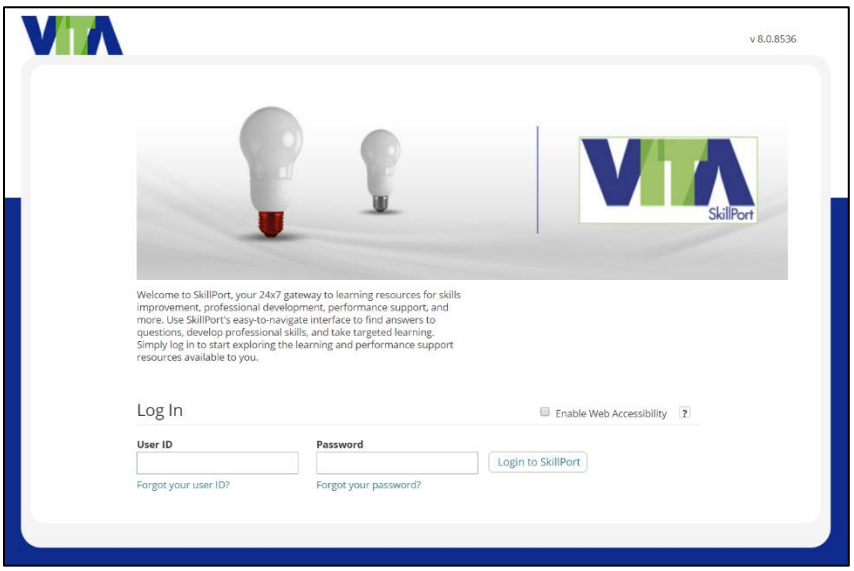
CompTIA Cloud+ CV0-001: Virtualization Course. [3]

- Cloud computing depends on virtualization.
- Virtualization technology has been around for decades, and now is used with cloud computing to allow for the rapid elasticity, or the provisioning and de-provisioning of cloud resources.
- Without virtualization, large-scale and dynamic cloud computing data centers would not exist to the extent that they do.
- Virtualization allows multiple virtual machines (VMs) to run at the same time on one physical computer.
    - Each VM has its own operating system (OS).
    - Each VM behaves as if it is running its own separate computer physically; instead, it is running on top of a hypervisor.
- A hypervisor manages access to the physical computer hardware for each VM.
    - Each VM running an OS can also be running a series of applications just like on a real physical computer.
    - Type I hypervisors run directly on the hardware.
        - Sometimes called bare metal hypervisors or Virtual Machine Managers (VMMs).
        - Examples of Type I hypervisors include products such as VMware's vSphere hypervisor, Microsoft Hyper-V, and IBM PowerVM.
        - Type I hypervisors used for mission critical systems.
    - Type II hypervisors do not have direct access to the underlying physical hardware.
        - They run on a host operating system.
        - They cause an extra layer of software – a host OS sitting between the hypervisor and the underlying physical hardware meaning more can go wrong.
        - Examples of Type II hypervisors include products like VMware workstation, Microsoft Virtual PC, and Oracle VirtualBox.

[3] CompTIA Cloud+ CV0-001: Virtualization course taken by Mr. Robert Kowalke (VITA Enterprise Architect) through VITA's Skillsoft Learning Center in April 2017.

    - Type II hypervisors are used most often at the consumer level such as testing or when learning about products.
- VMs are emulators that execute programs like a physical machine.
- Elasticity implies we can rapidly provision and de-provision compute resources.
- Cloud service portability means we should be able to switch to different cloud providers and any application we've been using with our previous cloud provider will be available.
- With cloud computing scalability, there is less capital expenditure on the cloud customer side because one can simply provision additional compute resources right away.
    - No upfront cost for all of this computing power.
    - Less waste of compute resources when scaling down.
    - Cloud resource usage is metered and billed accordingly.
- When agency executives are considering whether to maintain the on-premise virtualization solution, or move the virtualization to the cloud, then it is prudent to draw up a business case for virtualizing certain services in the cloud.
    - What are the benefits of cloud virtualization?
        Options:
        - Computer resources are used more efficiently.
        - Resources can be rapidly scaled.
        - Network traffic can be isolated.
        - The computing infrastructure can be spread out.
    - Answer
        Option 1: This is correct.
        Computer resources are pooled among all the cloud tenants, so there is less waste of computer resources.
        Option 2: This is correct.
        Resources can be provisioned and de-provisioned as business needs change. The business pays for only the resources it provisions.
        Option 3: This is correct.
        Network traffic between cloud tenants is isolated, so they do not affect each other. Applications can also be isolated to allow for custom configuration and security settings.
        Option 4: This is incorrect.

Having a dispersed computing infrastructure will likely cost more than a consolidated infrastructure. Being able to consolidate a computing infrastructure is a benefit of cloud virtualization.



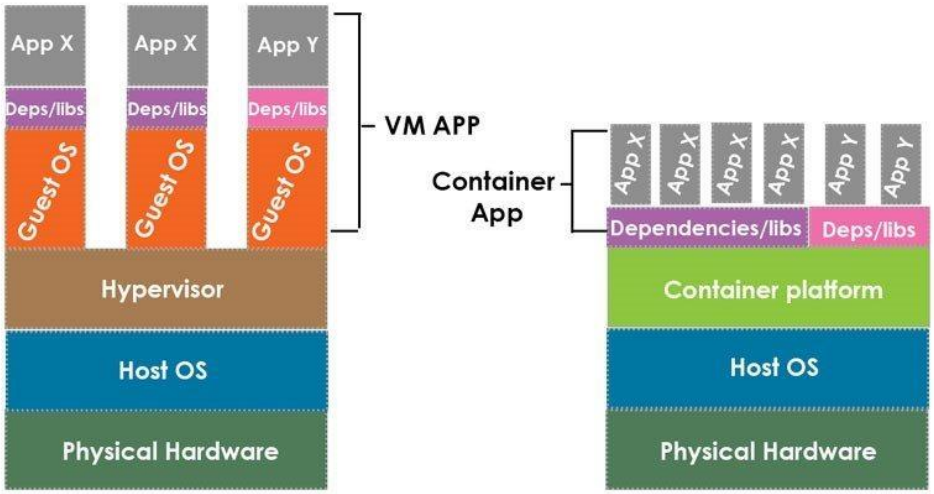
Application Virtualization Smackdown. [4]

- There is server virtualization, desktop virtualization, and application virtualization.
- Application virtualization relates to isolated containers for applications.
  - OS, registry, and file system contained within.
  - Decreased application packaging time.
  - Decreased testing time.
  - Decreased help desk calls.
  - Improved security due to isolation from the underlying OS.
  - Can run incompatible applications side-by-side.
  - Simplifies OS migrations.

[4] Application Virtualization Smackdown by CDH in 2011. Obtained from the internet in December 2018.

Bare Metal Servers vs. Virtualization: What Performs Better? [5]

- Dollar for dollar, bare metal servers can process more data than any other solution.
  - If you are running a single-threaded application, it does not matter how much cores you throw at it, you just will not see any benefit.
- When it comes to security, dedicated bare metal servers are as safe as it gets.
  - They are a single-tenant environment in which each server is under the control of a single client.
  - The only way bare metal can be compromised is if somebody breaks into the data center with the intention of damaging or stealing data.
- Compared to VMs, bare metal is time-consuming to provision
  - Plan wisely as deploying a physical server is not as fast as powering up a VM.



- Virtualized environments are more easily managed than bare metal.
- Besides scalability, reduced costs are the main reason why everything is going cloud, which makes it so easy to manage and scale cloud resources.

[5] Bare Metal Servers vs. Virtualization: What Performs Better? Article by phoenixNAP Global IT Services in 2018.

    - And easier to scale your costs as well.
- A virtual environment is ideal for:
    - E-Commerce.
    - SaaS.
    - Testing new features.
    - Enterprise resource planning (ERP) solutions.
- If you are running data-crunching apps, which can significantly benefit from direct access to physical hardware, a bare metal server should be your first choice.
    - Bare metal comes out as the winner with its lower latency and lower CPU utilization, consequently providing faster result times and more data output.
- We claim that cloud workloads can be moved around freely, are more flexible and scalable, tend to cost less, and are more easily maintained than bare-metal, but, they also tend to offer less performance and safety.

Virtual Machine or Physical Server – How to choose. [6]

- There are numerous advantages of VMs over physical servers.
- In some cases, VMs are simply not cost effective.
- VMs exist when a pool of computer resources are divided among a number of systems (operating systems, or specialized software) that "act" as if each was a complete computer server.
- One physical server can in theory host dozens of hypervisors/VMs.
    - Each of these VMs will run a different application.
        - Replaces the need for dozens of separate, dedicated, and underutilized PCs.
        - Such a system will work perfectly on the condition that not all users/applications require the full resources at the same time.
- Another obvious advantage of a VM is easier management.
    - In case of failure or even re-configuration:
        - Rebuilding a physical (hardware) server is a difficult task.
        - Reconfiguring a VM is a simple software configuration.

[6] Virtual Machine or Physical Server – How to choose. Actus Digital article on January 8, 2018. Obtained from the internet in May 2018.

- VM scalability is a breeze.

To summarize, the 3 main advantages of VM are:

	Virtual Machine	Physical Server
CAPEX Investment	Low. One server can serve dozens of users	High. Each user requires a physical machine.
System Management	Easy. One dashboard manages everything.	Medium. Each physical server is managed.
Scalability and Flexibility	High. Can add and modify services at ease.	Medium. Each physical server must be configured.

CAPEX = Capital Expenditures

- Based on these, we can recommend VMs for applications that are:
  - Running a large number of services, or users.
  - Each service is variable in terms of CPU requirements
    - Sometimes high.
    - Often low.
  - There are many changes.
    - New applications added.
    - Sizing is often modified.
  - Web servers that serve thousands of users, are very appropriate for VMs as an example.
- On the other hand, physical servers give benefits when:
  - The number of services is constant.
  - Each of the services requires constant resources.
  - It is possible to size a server to use close to 100% of its capacity, constantly.
  - There are very few changes over time.
    - In the context of Media encoding servers, where we have customers encoding video from a number of specific TV channels, the amount of CPU used is constant, as are the number of channels, as well as the retention period and the video quality.
      - This means it is possible to size the servers at almost full capacity.
      - In this context, a physical server is a better choice.

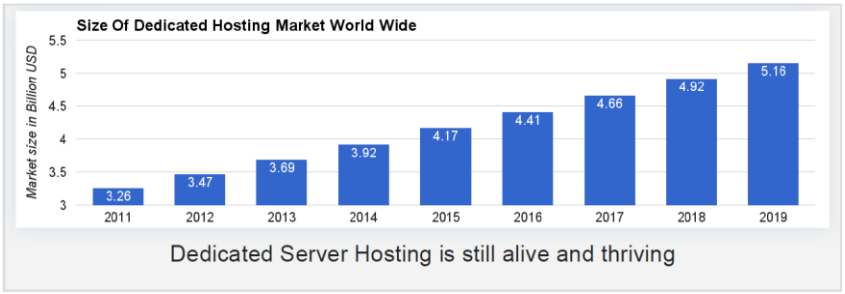
- A physical server will deliver 10% to 20% more than VMs and will cost 10% to 20% less.

So the VM disadvantages are:

	Virtual Machine	Physical Server
CPU Power	low. We noticed 10% to 20% loss compared to physical.	Good. We can run at CPU almost full capacity for months.
O/S cost	More expensive. Can reach US$ 5,000 and more.	Medium. A Windows Server O/S costs around US$ 1,000 per server.

Physical vs. Virtual Server: Which one should you choose? [7]

- Dedicated servers are still a favorite choice for many.
  - Research shows the dedicated server market grows by $237 million each year. [8]

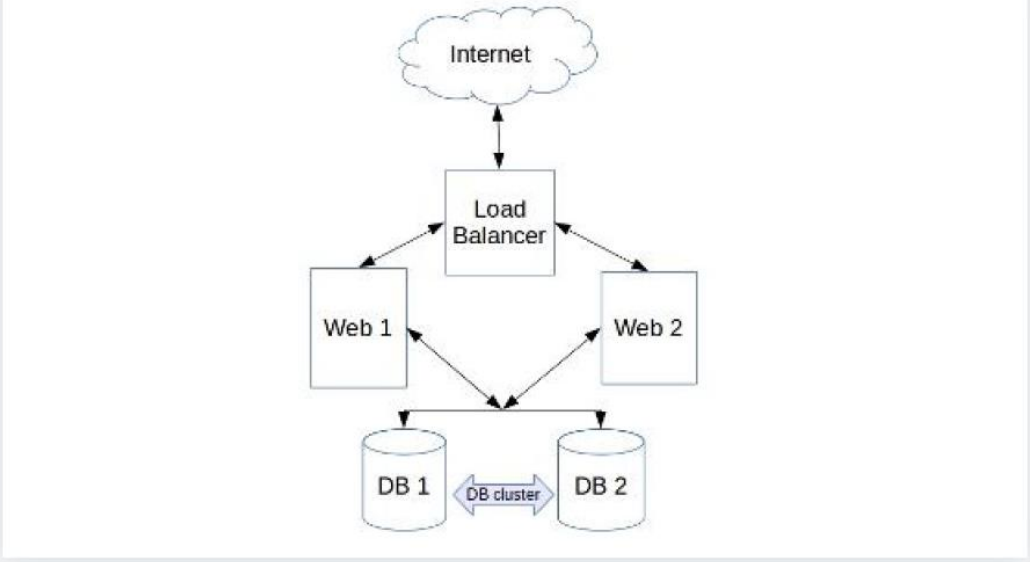


Dedicated Server Hosting is still alive and thriving

- If you take two dedicated servers of identical capabilities, virtualize one of them, and run the same application in both servers, the dedicated server will always show better performance.

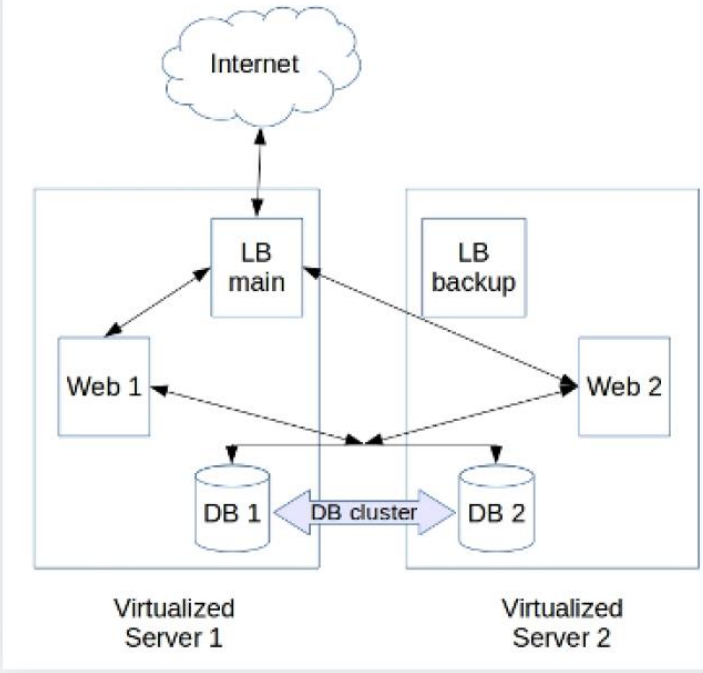
[7] Physical vs. Virtual Server: Which one should you choose? Article by Sajan Sebastian on BobCares – April 16, 2018. Obtained from the internet in May 2018.
[8] Worldwide Dedicated Hosting Market Size statistics at www.statista.com.

  - The performance difference is noticeable only in applications that run thousands of transactions per minute (like busy eCommerce sites for example).
  - It's because virtualization always inflicts a performance penalty.
  - If performance improvement is your sole concern for a server upgrade, go ahead and choose a physical server.
- The most popular reason to virtualize is to save money.
- Virtualization is beneficial:
  - if you have a large number of servers.
  - if you're using an open source virtualization solution such as oVirt.
    - Software licensing costs can be too heavy for smaller installations.



A mid-size business infrastructure with load-balanced web and database services.

To implement a 5-server infrastructure using a virtualization solution, we'll only need 2 powerful physical servers, as shown below:



The 5-server infrastructure could be consolidated into 2 higher capacity servers with a virtualization system.

The Advantages and Disadvantages of Virtualization. [9]

- Virtualization advantages:
  - Reduced spending – requires fewer servers and extends lifespan of existing hardware.
  - Easier backup and disaster recovery – .Less manpower and a fraction of the equipment.
  - Better business continuity.
  - More efficient IT operations.
- Virtualization disadvantages:
  - Upfront costs.

[9] The Advantages-and Disadvantages of Virtualization by Milner Technology Services on July 14, 2015. Obtained from the internet in May 2018.

  - Software licensing considerations.
    - Becoming less of a problem, but be sure you understanding how they relate to a virtualized environment.
  - Possible learning curve for IT staff.
- For many businesses comparing the advantages to the disadvantages; moving to a virtual environment is typically the clear winner.

Docker vs. VMs. [10]

- VMs are used extensively in cloud computing.
  - Isolation and resource control have continually been achieved using VMs.
  - VMs load a full OS with its own memory management and enable applications to be more efficient and secure, while ensuring their high availability.
- Docker containers are executed with the Docker engine rather than the hypervisor.
  - Containers are:
    - Smaller than VMs.
    - Enable faster start up with better performance, less isolation, and greater compatibility possible due to sharing of the host's kernel.
  - Containers present lower system overhead than VMs.
    - Performance of an application inside a container is generally the same or better when compared to the same application running within a VM.

[10] Docker vs. VMs article by Sudhi Seshachala on DevOps-dot-com in November 2014. Obtained from the internet in December 2018.

Will Containers Replace VMs? [11]

- Architecturally speaking, VMs and containers have enough architectural similarities that some question the long-term survival of VMs, especially since VMs are already a couple of decades old.
- There are also serverless cloud options available now that dynamically manage the allocation of machine resources instead of staff.
- Most enterprises have been using VMs for quite some time and that will continue to be true for the near future.
- Most container infrastructure deployments are going to be on VMs.
    - Looking at where container runtimes are deployed, as well as container orchestration systems such as Kubernetes, more often than not, it is on virtual infrastructure and there is a very important reason for that.
    - In most cases, the basic provisioning processes for infrastructure are going to be based on VMs, so even if you wanted to switch to provisioning on bare metal, it is quite possible that you wouldn't have any processes in place to do that.
- The main benefit of a VM is deploying the entire stack fully configured and ready-to-roll.
    - Biggest issue with VMs is that if you want to deploy a stack that has unique scaling needs, you are going to be in a world of hurt.
    - Containers have a huge upside, especially in the enterprise space because they can scale different parts of a platform as needed, independent of each other.
- One of the most important benefits containers provide is once you have a containerized application, it runs in exactly the same environment at every stage of the lifecycle, from initial development through testing and deployment – you get workload mobility at every stage of its lifecycle.
- Containers are helping to eliminate application-level friction for end users and IT.
- VMs and containers will continue to coexist for some time, mainly because businesses require the benefits of both.

[11] Will Containers Replace VMs? InformationWeek article on June 21, 2018. Obtained from the internet in December 2018.

Containers and Kubernetes vs. VMs vs. Config Management. [12]

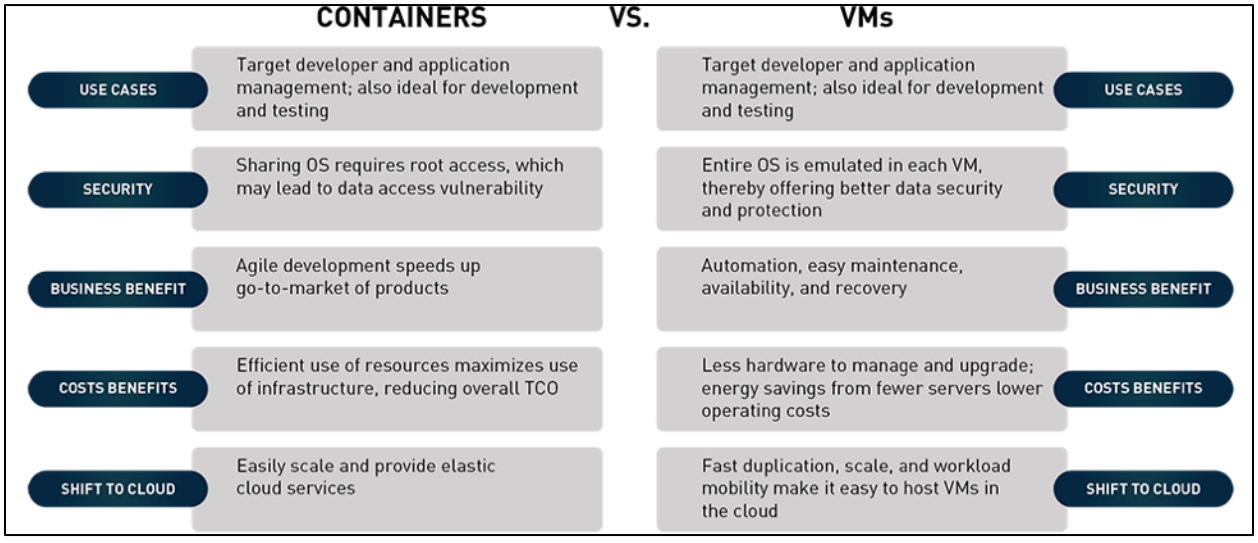
- Problem solved by VMs.
  - VMs allow you to run different applications on the same hardware, but in an isolated fashion.
    - You don't wonder about what was already using ports.
    - You're not worried about what particular OS versions or patches the other applications required.
    - If you have different applications that have different operating system needs, VMs will deal with that.
- Problem solved by containers/Kubernetes.
  - Containers allow running a single application in a lightweight fashion and still stay separate from other containers and applications on that server.
    - Removes requirement for having a separate VM for each application.
    - All containers use the same kernel as the host.
  - Containers are simpler for developers to spin up, as someone else already made the OS decisions.

Containers vs. VMs: A 5-Minute Guide to Understanding Their Differences. [13]

- Containers do not package system resources as much as VMs.
  - Can run at least twice or more the number of applications on the same server with containers than if you were to run them with VMs.
  - Maximizes resource usage and brings down operating costs.
  - Agile development and testing speed up time-to-market with containers.
- If you are looking to develop applications, or run a single or a handful of applications in multiple instances, and resource footprint is a concern, consider containers.
- If you are looking to run multiple applications and your resource footprint can be fluid, consider VMs.

[12] Containers and Kubernetes vs. VMs vs. Config Management by Scalyr.com on June 26, 2018. Obtained from the internet in December 2018.
[13] Containers vs. VMs: A 5-Minute Guide to Understanding Their Differences by MapR (Dataware for data-driven transformation.) on May 16, 2018. Obtained from the internet in December 2018.

- Security has been the single biggest problem around containers.
    - Containers, by sharing the OS, require root access, which makes the data vulnerable and at risk for unauthorized access.
    - VMs have a very robust, rich set of security services that make them attractive for sensitive data and for production environments.
        - VMs also have a very mature ecosystem in terms of network, storage, data protection, and recovery that can make them better for production environments.
- Container elasticity allows organizations to create containers on demand and tear them down when done.
    - Scaling up and down your services on cloud by spawning new containers is easier and more cost-effective than it is with VMs.
    - Containers in cloud also allow you to use only the minimum cloud resources you need for your service, thereby keeping your subscription costs down.
- If you have a heavy development, test, or integration environment, switch to containers.
- If you have multiple applications with varying characteristics requiring a secure environment, remain on VMs.
- If you are looking to deploy in the cloud or offer services in the cloud, standardizing the deployment in containers will be a good idea.



- Eventually, it will be prudent to envision and plan for both containers and VMs to coexist in both data centers and in the cloud, since each has its own benefits and challenges.

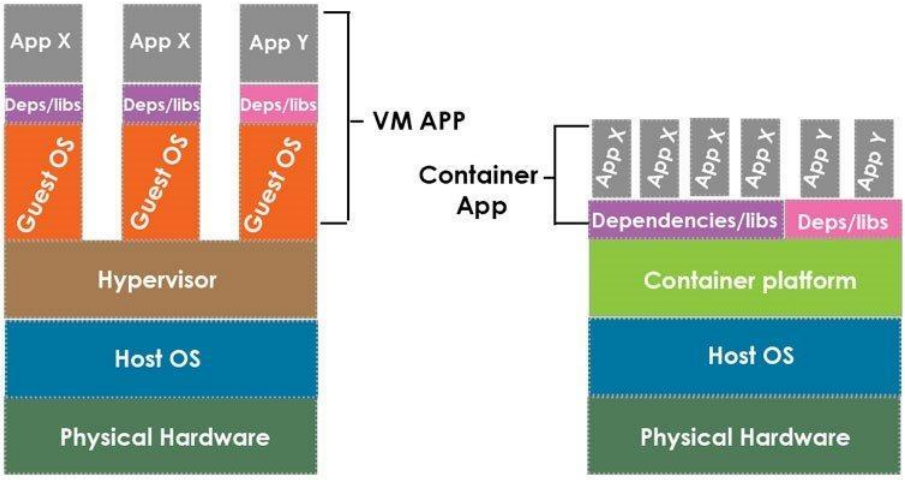
Containers vs. Virtualization: Which is Superior? [14]

- Virtualization has been the foundation for many modern technologies, and that CAN'T go without saying for the next- generation network implementation: Network functions virtualization and software-defined networking (NFV/SDN).

- The VM approach is much more involved in scope.
  - Relies on a Hypervisor (e.g. KVM, XEN) which emulates an entire physical machine, assigns a desired amount of system memory, processor cores and other resources such as disk storage, networking, PCI addons and so forth.
  - Most x86 processors manufactured from 2013 onwards include virtualization-specific optimizations (Intel VT-x, AMD-V) which bring virtualization overhead penalties on the processor to around 2%, a more than fair trade-off for the functionality virtualization brings.
    - System memory usage might end up being the most important difference between virtualization and containers.
  - One of the true advantages of VM's over containers is their portability.
    - Although docker containers offer a certain degree of portability between the host operating system by packaging dependencies with the application, there's no guarantee the underlying host OS is compatible with XYZ container application.
    - Another advantage is the maturity of VM management solutions, though Kubernetes is steadily closing this gap.
- Container-like technologies have existed for a long time, though under different names: jails, sandboxes, etc.

[14] Containers vs. Virtualization: Which is Superior? Article by Lanner of LEI Technology on July 5, 2017. Obtained from the internet in December 2018.

- Containerization has recently matured enough and made headway into production environments.
- Containers essentially isolate an application from the host through various techniques, but utilize the same host systems kernel, processes (e.g. network stack) to run applications.
- The real efficiency in containers comes from reduced memory usage through the elimination of the guest OS, the subsequent de-duplication of processes which consume additional resources and the reduction in application size from aforementioned reductions.
  - Combine that with the ability to manage resources like system memory on-the-fly and dynamically, it could make for a much more efficient deployment option.
- On paper, containers fit more in line with NFV/SDN initiatives and the industry has taken notice, as Kubernetes is one of the fastest growing open source projects to date.



Containers vs. VMs: Where Should IT Pros Put Their Money? [15]

- Containers are not yet the VM killer that many make them out to be.
- The primary battle cry for containers lies in the ability to significantly reduce cloud-computing resources.

[15] Containers vs. VMs: Where Should IT Pros Put Their Money? Article by Light Reading, an Informa Business trading within KNect365 US, Inc. on April 26, 2017. Obtained from the internet in December 2018.

- Regardless of the potential for a small cost savings early on, don't count on containers being a great way to save money in the long term.
- The portability and share-ability of containers blows away anything that VM architectures can provide.
- The true advantages of containers are gained in enterprises that require continuous integration (CI) on a large scale.
- Containers restrict the OS type and version that can be run.
    - VMs will run whatever OS you throw at it.
    - This OS restriction is a deal breaker for companies that must support applications running on legacy operating systems.

LightReading Networking the Communications Industry

Containers Research

VITA Platform Domain Topic Report on IT Solutions Hosting Services. [16]

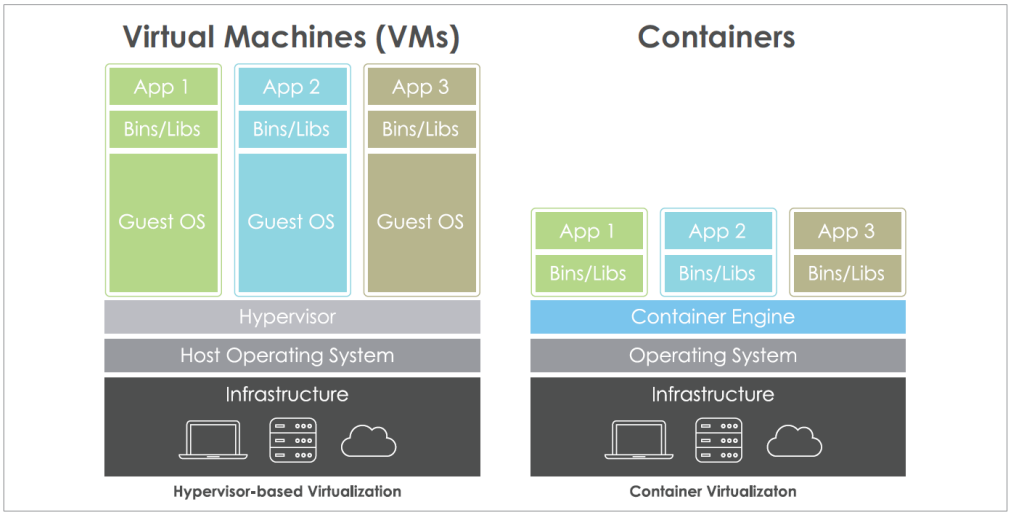
What is a container?

A container is a packaging format that encapsulates a set of software with its dependencies and runs in a virtual server environment with a minimal operating system (OS). Therefore, it is a form of virtualization.

The difference between VM's and containers is that each VM has its own full sized OS, while containers have a minimal OS.

Containerization is the encapsulation of an application in a container.

A physical server running three VMs has a hypervisor and three operating systems running on top of it. A container is a server running three containerized applications using a single operating system – shares the operating system kernel using a software tool to cluster the CPU's into a single virtual host.

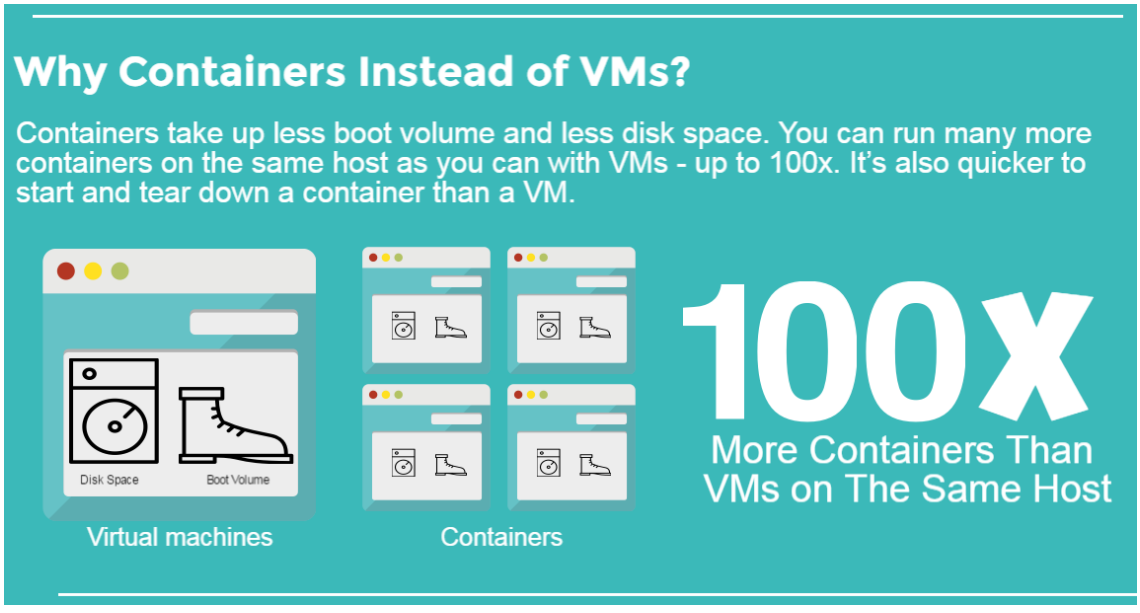


[16] Commonwealth of Virginia (COV), Enterprise Technical Architecture (ETA), Platform Domain Topic Report, IT Solutions Hosting Services by VITA in 2018. Draft copy obtained in December 2018 with anticipated completion date of Dec-31-2018.

Containers solve the problem of how to get *"software"* to run reliably when moved from one computing environment to another.

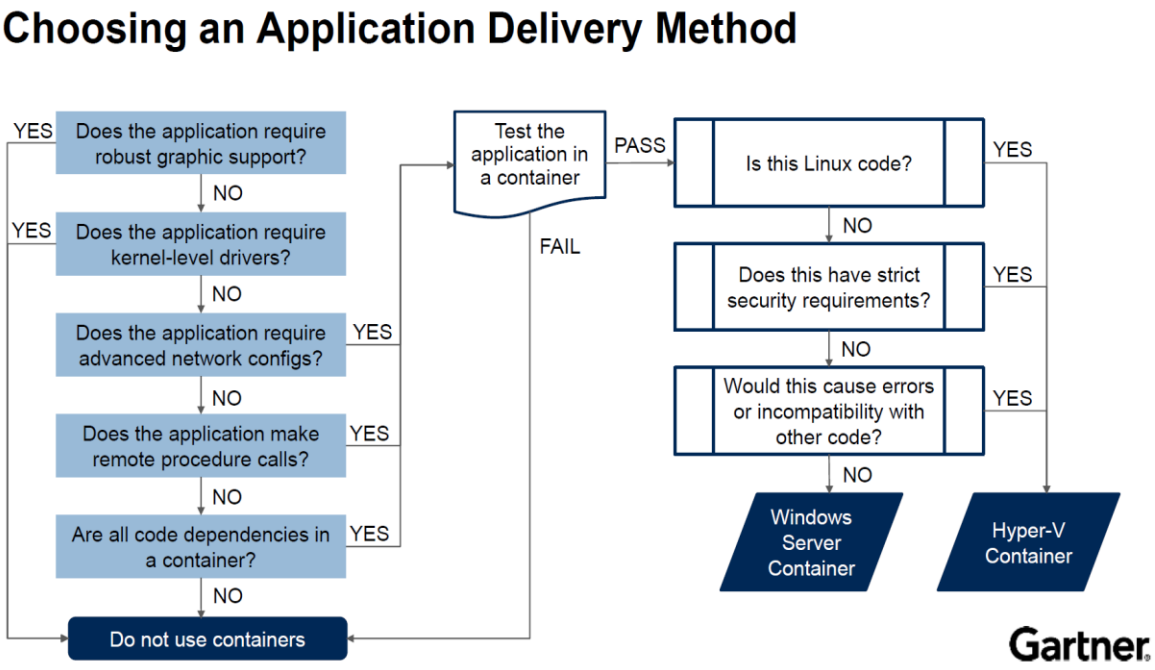
PRO's

- Containers are lightweight and use far fewer resources than VMs.
  - A single server can host far more containers than VMs.
  - Containers generally spin up much faster than VMs.
- Containerization allows for modularity.
  - A developer normally running a complex application inside a single container could split the application into modules. Thus, the application becomes easier to manage, because each module becomes relatively simple to manage. Plus application changes would apply directly to the modules instead of having to rebuild the application.
- Application Development of (net-new) applications is the primary adopted use for containerization.
- Containerization supports portability.
  - Applications can be migrated between platforms with relative ease.



CONS

- Not all applications fit the container model.
- Security is a major concern, because containers share a common OS kernel.
- Lose visibility and control over what is running in your infrastructure.
- Management tools to orchestrate large numbers of containers are not currently as comprehensive as the more mature VM environments.
- The networking complexity of container applications' coexistence with VM applications, along with public or private cloud applications can require a major effort.

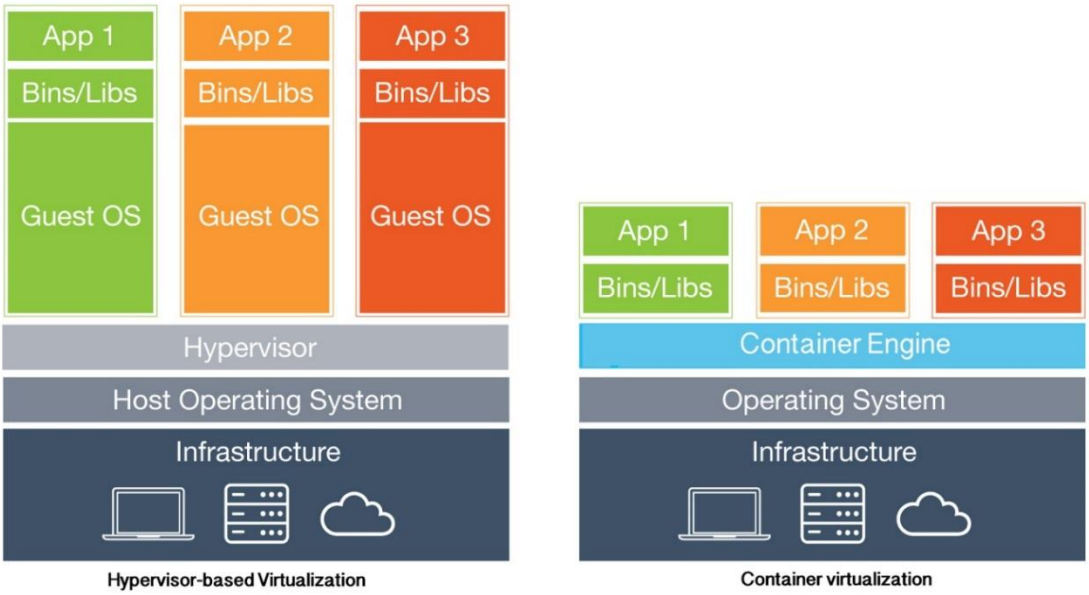


Container Basics Whitepaper – Chapter 1. [17]

- Virtual containers have roots in FreeBSD jails and Solaris Zones.
- Linux Containers (LXC) helped establish containers as a virtualization technology suitable for cloud data centers.
    - Docker came along later.

[17] Container Basics Whitepaper – Chapter 1 by Twistlock Cybersecurity Platform. Obtained from the internet in December 2018.

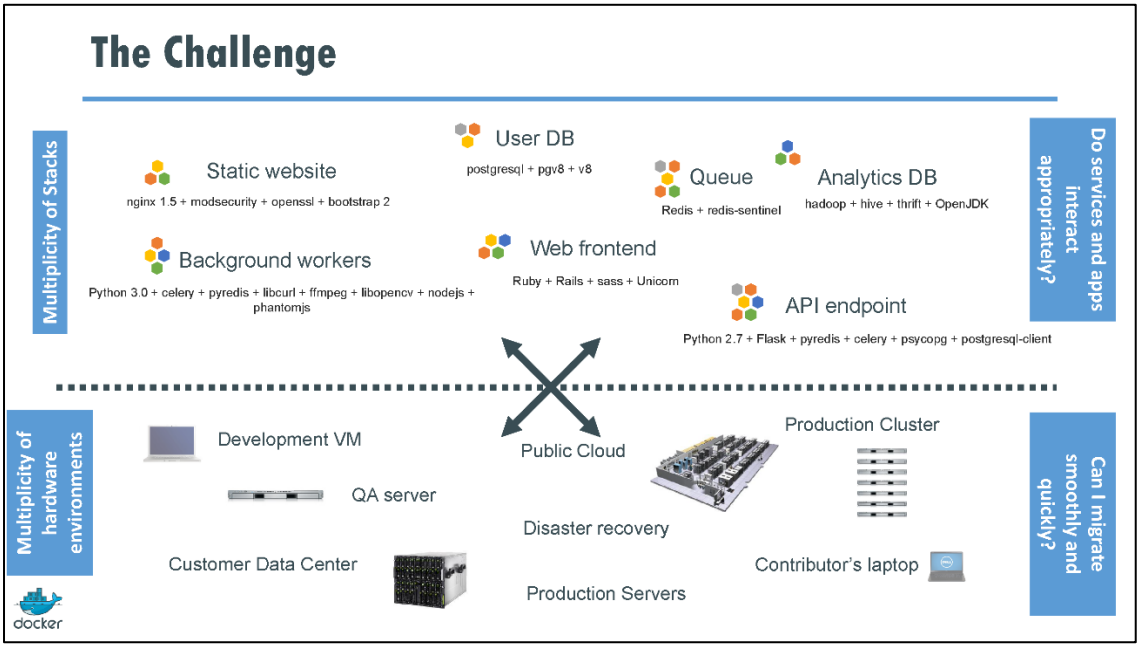
- Originally, Docker was a project to build single-application LXC containers.
- Docker has made several significant advances to the container concept, including moving away from LXC as the container format.
- Docker containers let users easily deploy, replicate, move, and back up a workload, thus giving cloud-like flexibility to any infrastructure capable of running Docker.
- For these reasons, Docker is often credited as the development that led to the modern-day popularity of virtual containers.



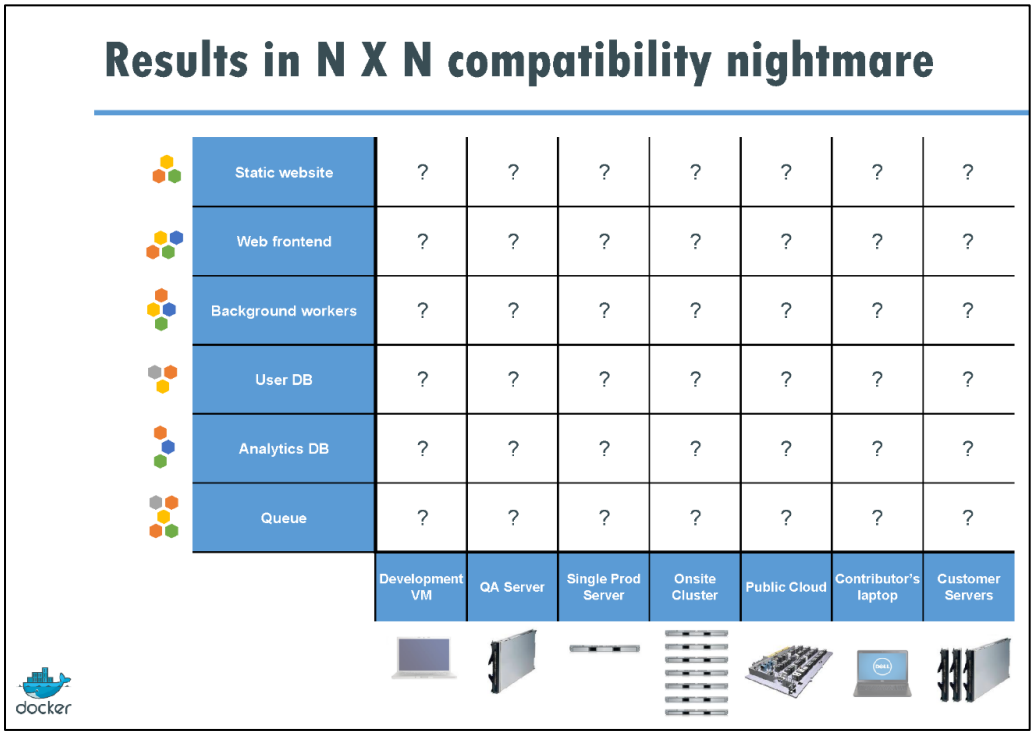
Docker, Containers, and the Future of Application Delivery. [18]

- The challenge that Docker seeks to address with containerization:

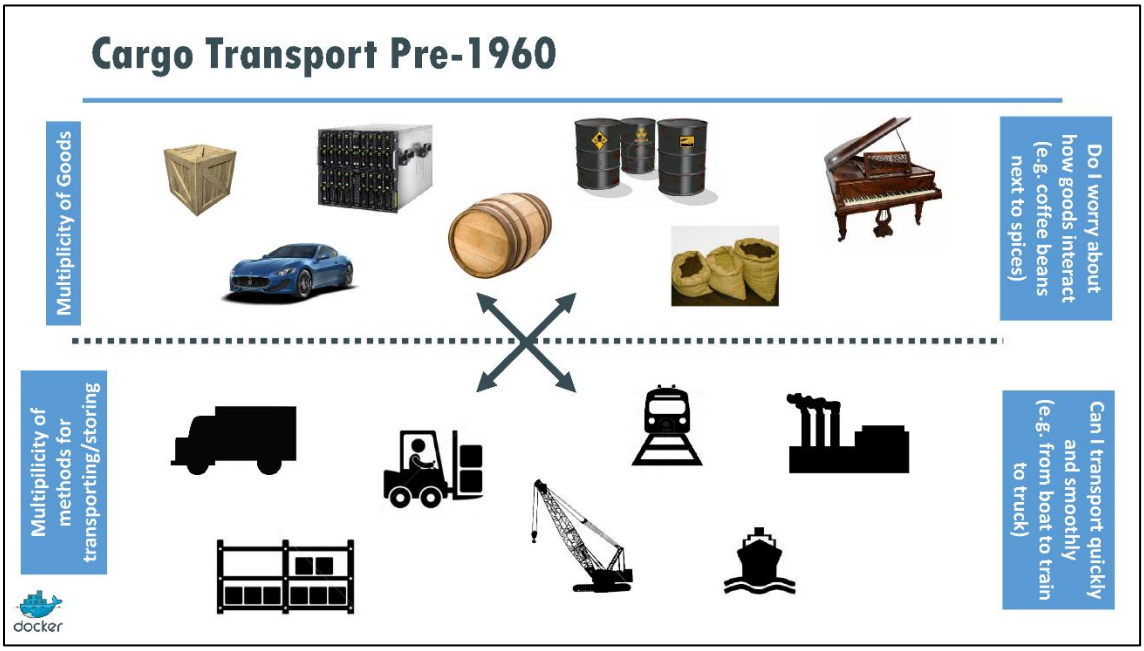
[18] Docker, Containers, and the Future of Application Delivery by Docker in 2013. Obtained from the internet in December 2016.

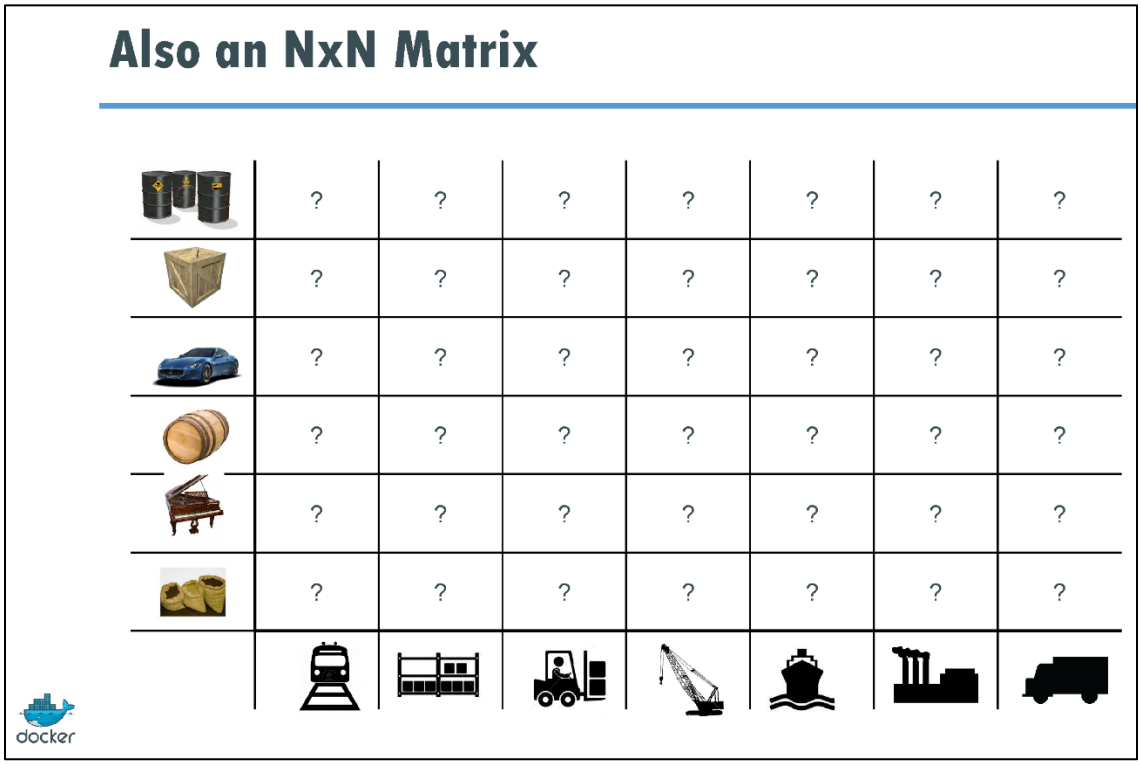


- The challenge results in a compatibility nightmare that containerization seeks to favorably mitigate:

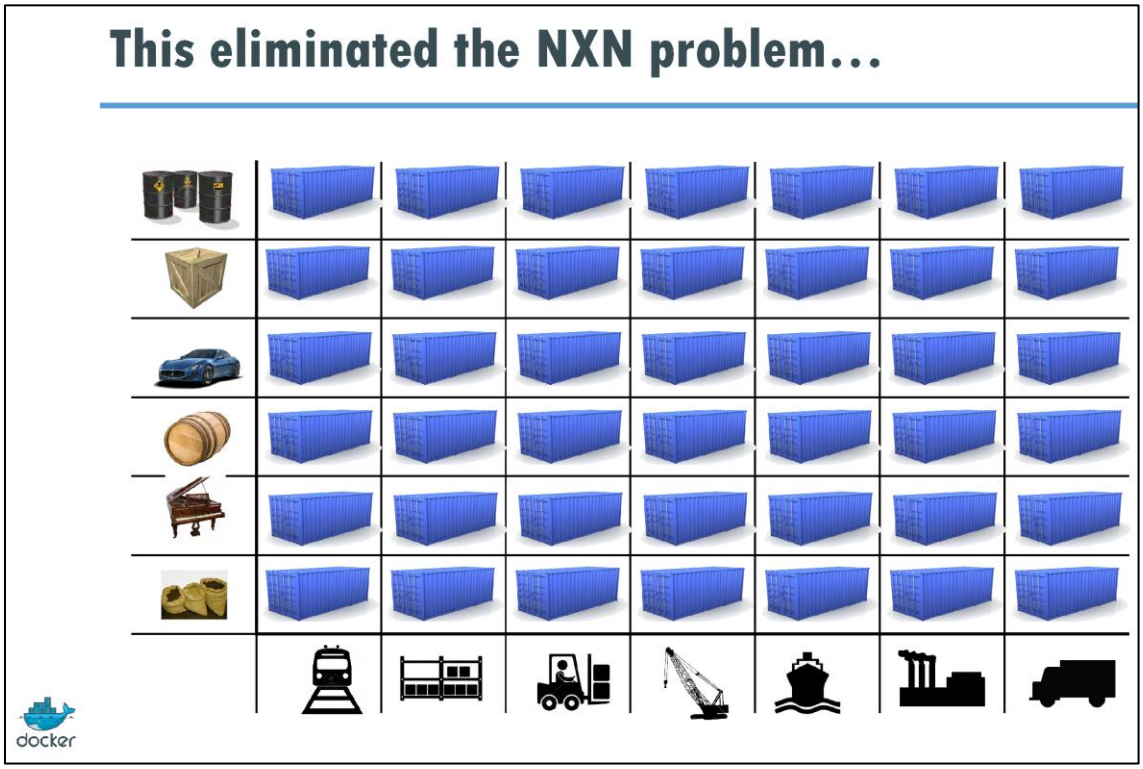


- Used the analogy of shipping containers solving the problem inherent in the high expense of cargo transport prior to the year 1960.

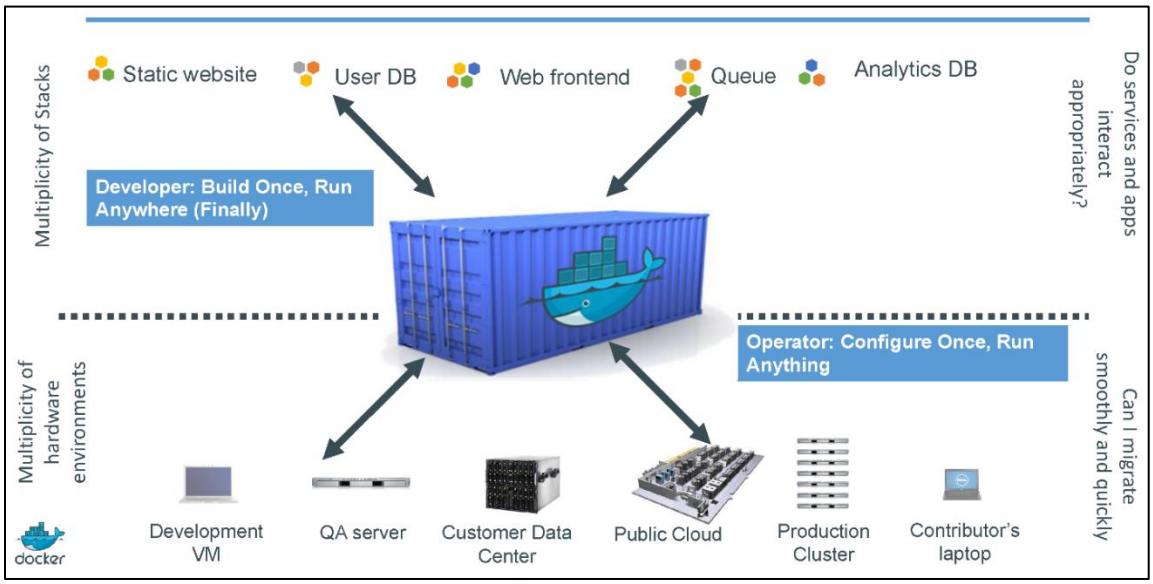




- Shipping containers solved the myriad of complex transport problems that various types and sizes of cargo caused.



- Related how Docker is a shipping "container" system for code.

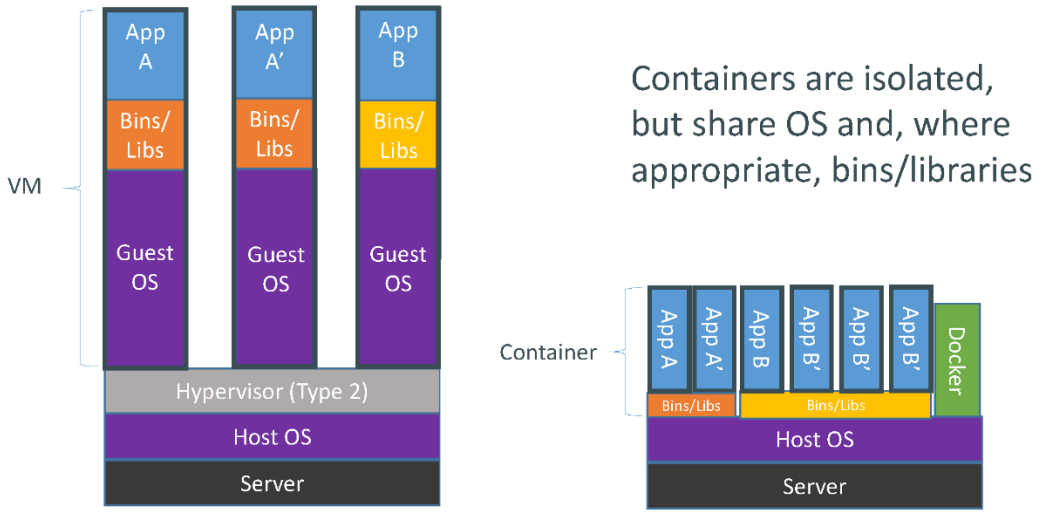


- Why containers matter?

	Physical Containers	Docker
Content Agnostic	The same container can hold almost any type of cargo	Can encapsulate any payload and its dependencies
Hardware Agnostic	Standard shape and interface allow same container to move from ship to train to semi-truck to warehouse to crane without being modified or opened	Using operating system primitives (e.g. LXC) can run consistently on virtually any hardware—VMs, bare metal, openstack, public IAAS, etc.—without modification
Content Isolation and Interaction	No worry about anvils crushing bananas. Containers can be stacked and shipped together	Resource, network, and content isolation. Avoids dependency hell
Automation	Standard interfaces make it easy to automate loading, unloading, moving, etc.	Standard operations to run, start, stop, commit, search, etc. Perfect for devops: CI, CD, autoscaling, hybrid clouds
Highly efficient	No opening or modification, quick to move between waypoints	Lightweight, virtually no perf or start-up penalty, quick to move and manipulate
Separation of duties	Shipper worries about inside of box, carrier worries about outside of box	Developer worries about code. Ops worries about infrastructure.

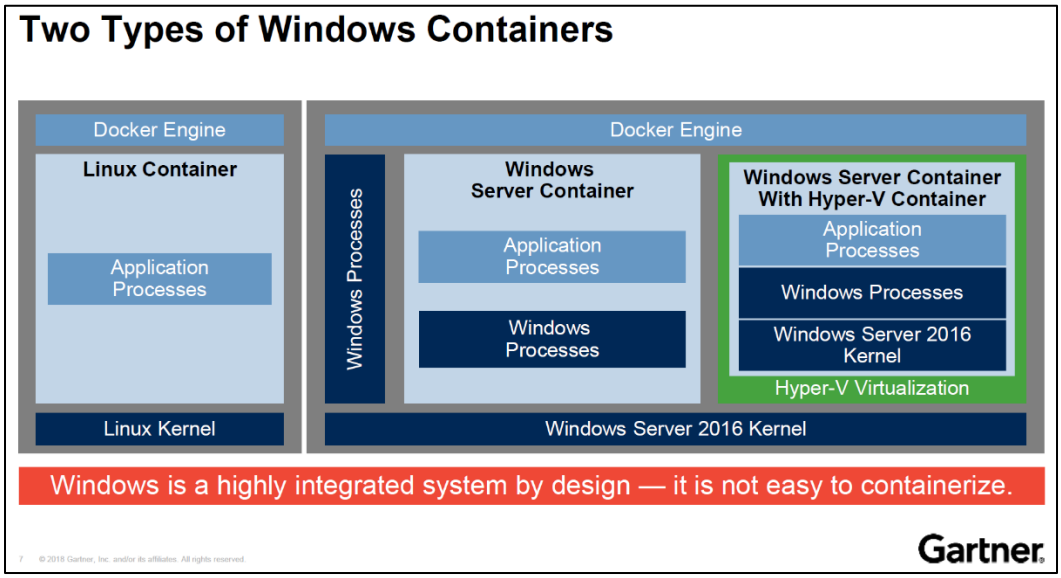
- With containers one can:
  - Run each application in its own isolated container without worrying about various versions of libraries.
  - Automate testing, integration, and packaging – basically anything you can script.
  - Reduce/eliminate platform compatibility concerns.

- Containers vs VMs



Assessing Enterprise Deployment Windows Containers. [19]

- Containers are for Windows too.



[19] Assessing Windows Containers for Enterprise Deployment by Anna Belak of Gartner - August 2018.

- Windows container support is still lacking as of 2018.

Ecosystem Support

Vendor	Windows Containers	With Kubernetes
Docker	✓	✗
Mesosphere	✗	✗
Pivotal/VMware	✓	✗
Rancher Labs	✗	✗
Red Hat	✗	✗
Amazon Web Services	✓	✗
Microsoft Azure	✓	✗
Google Cloud Platform	✗	✗

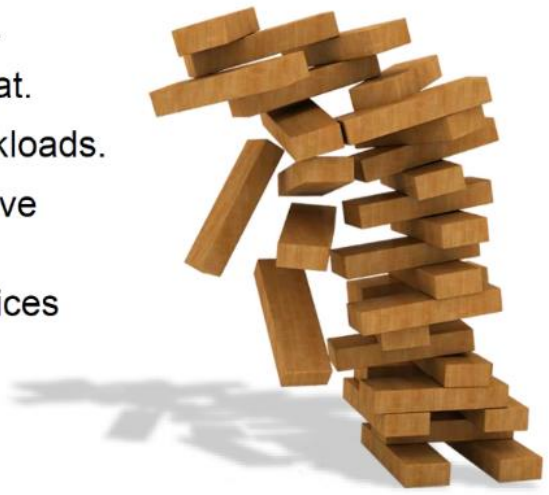
… is still lacking.

Microsoft, Docker, and other major contributors expect generally available Kubernetes for Windows containers by the end of 2018.

Gartner

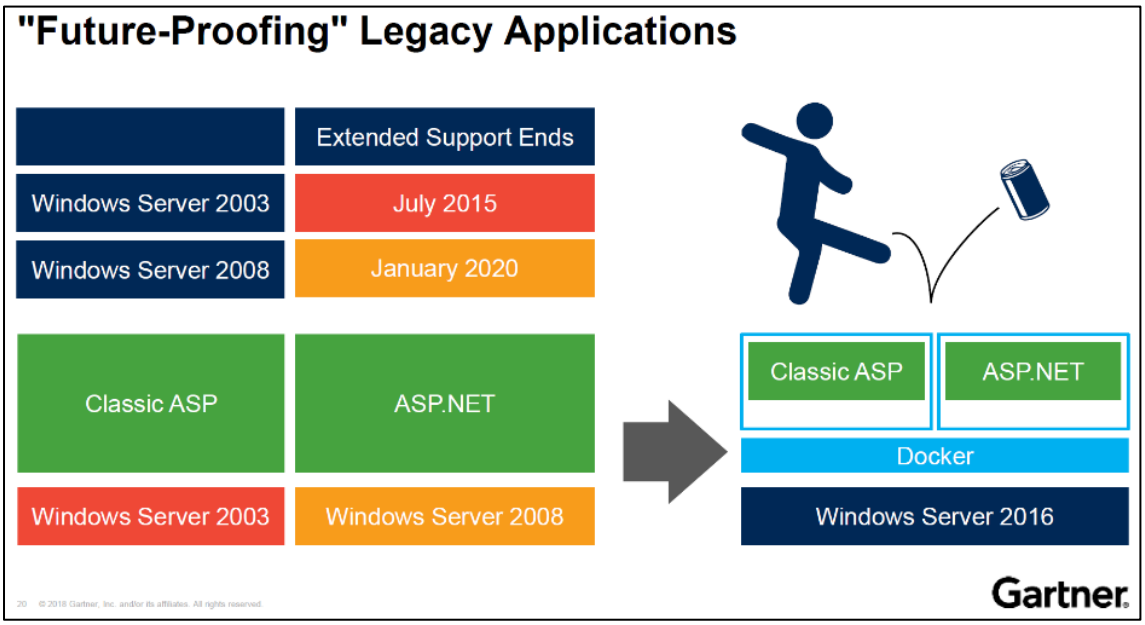
- Windows containers:

- Make Windows a little more modular.
- Provide a consistent packaging format.
- Enable future-proofing of legacy workloads.
- Increase workload density and improve resource utilization.
- Facilitate future hybrid OS microservices development.
- Ease cloud migration efforts.



Gartner

- Windows containers help to future-proof your legacy applications.



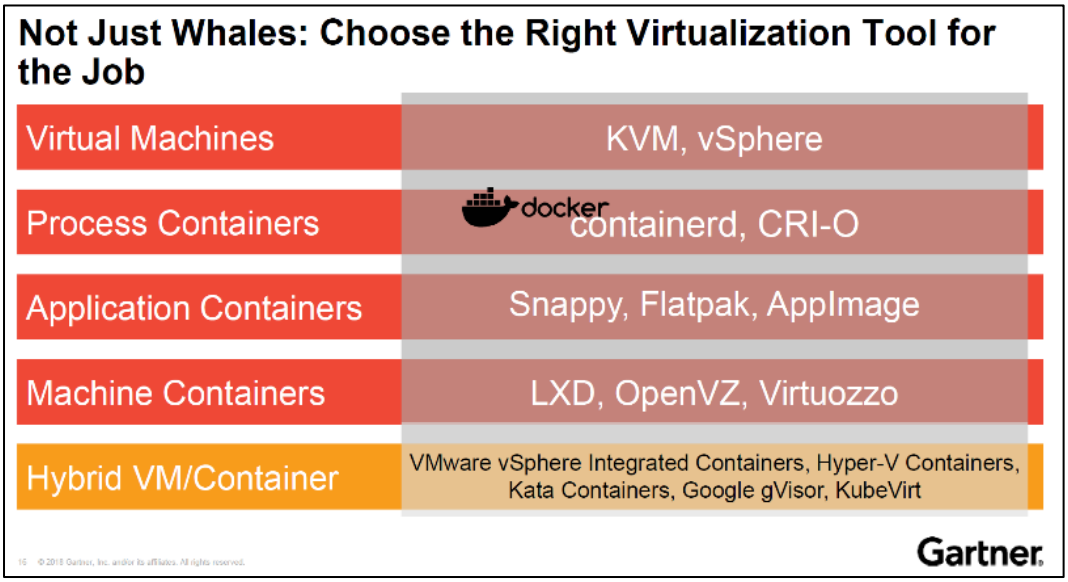
- Prove the agility benefits of Windows containers by piloting adoption with select in-house .NET applications:
  - Wait for broader integration with orchestration tools and more commercial application support before widespread use.
- Minimize container size by preferring Nano Server for all containerized Windows applications:
  - Use Windows Server Core for refactored legacy applications that have OS dependencies not yet supported by Nano Server.
- Selecting container orchestration and management tools that support both Windows and Linux containers.

Top Ten Container Myths. [20]

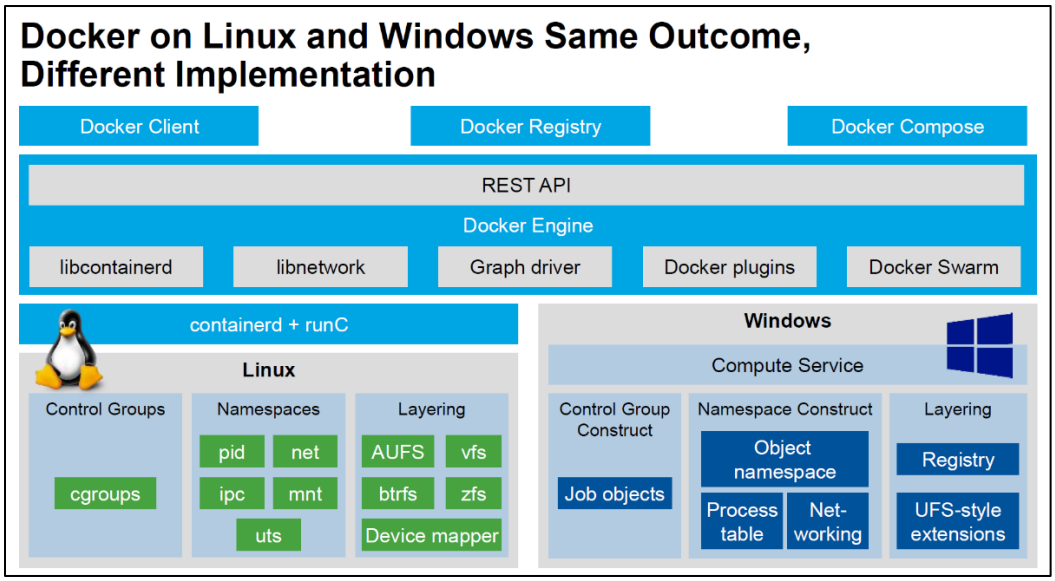
- Myth No-1: Containers are new technology.
  - Leverage existing Linux system admin experience regarding containers.

[20] TechDemo: Top Ten Container Myths by Richard Watson from the Gartner Catalyst Conference – August 2018 in San Diego, CA.

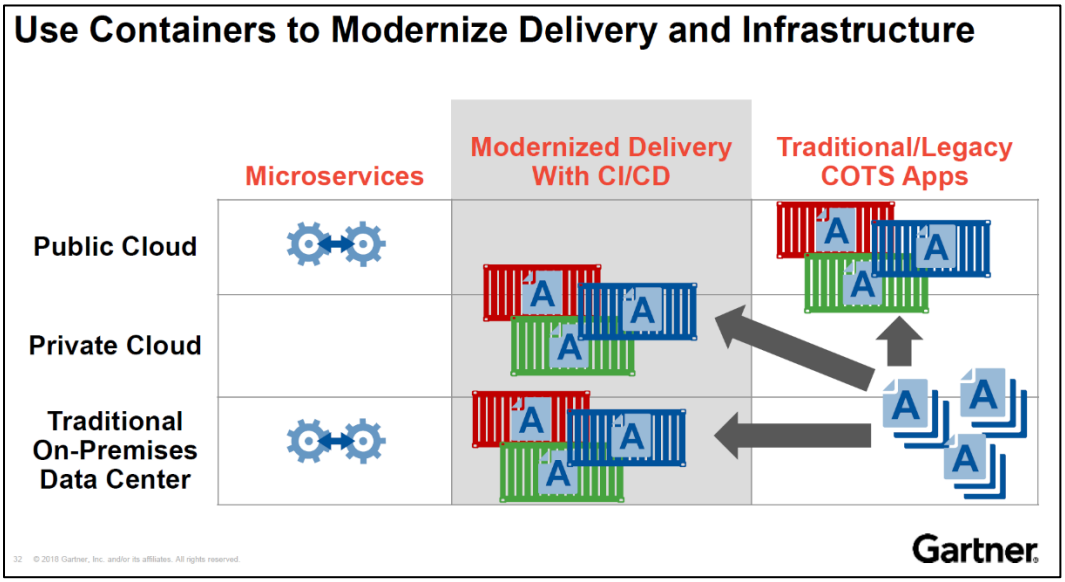
- Myth No-2: Containers mean Docker.
  - Use the right virtualization technology for the job.



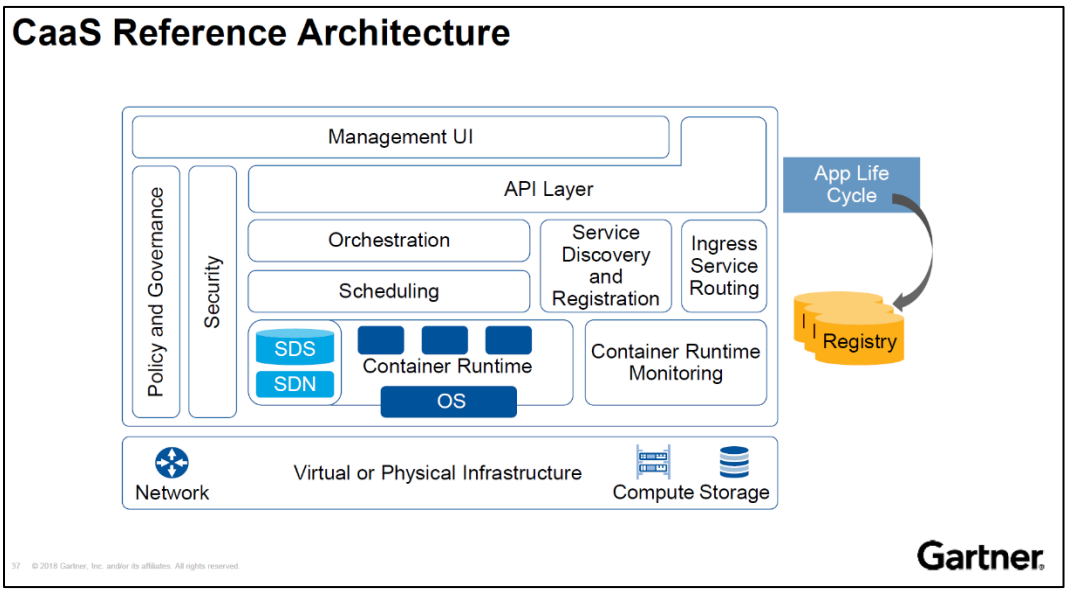
- Myth No-3: Containers are just for Linux.
  - Plan for containers across Linux and Windows servers.

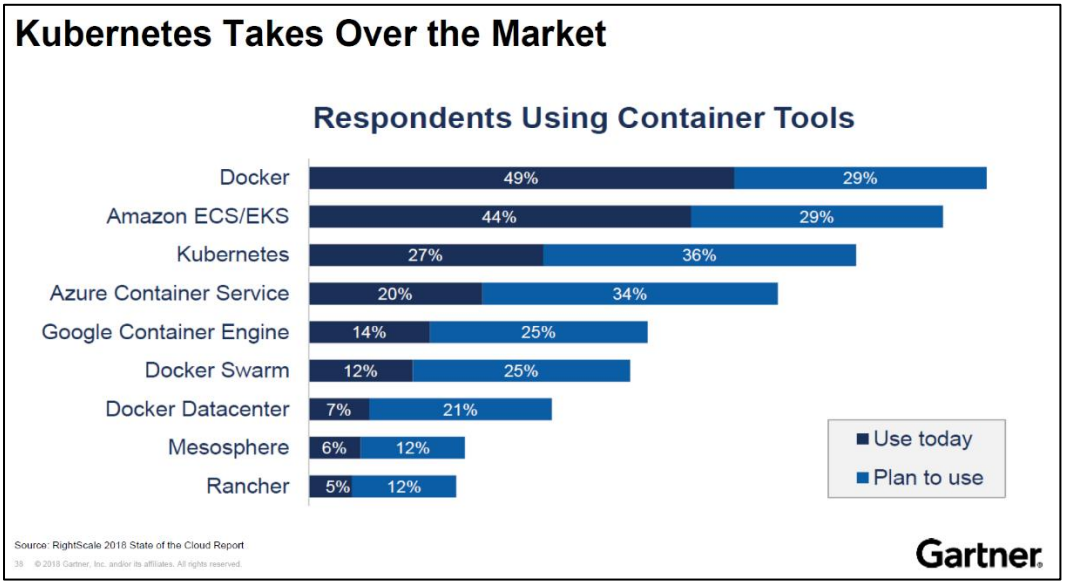


- Myth No-4: Containers are just for microservices.
  - Also, modernize delivery of existing applications with Docker.

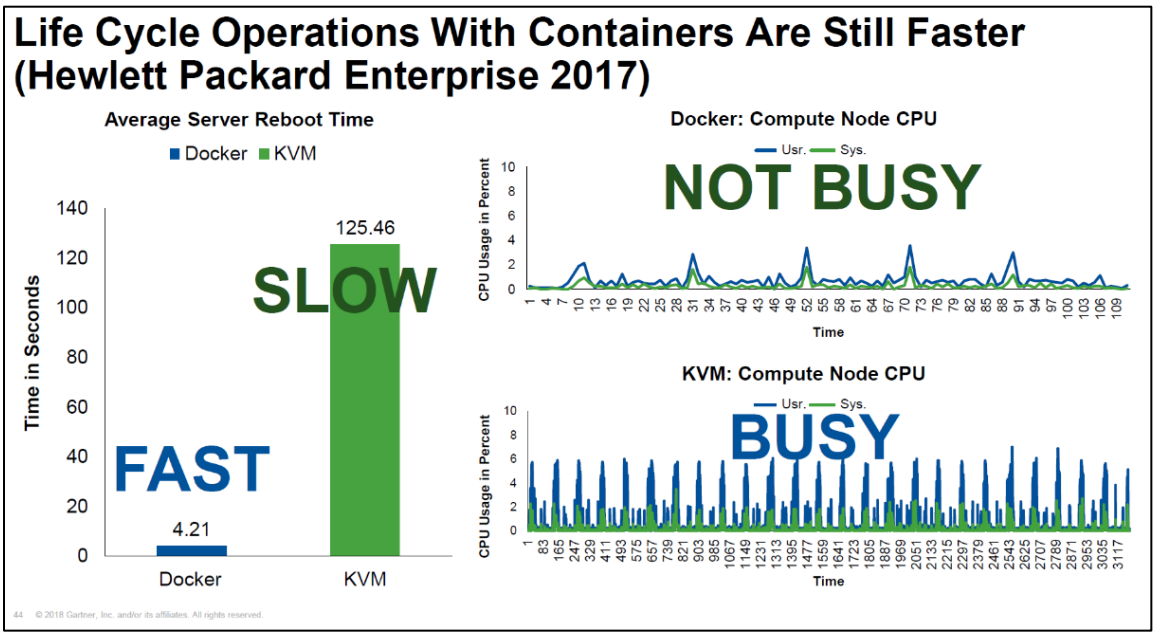


- Myth No-5: Docker (standard edition) is enough to build a complete container platform.
  - Buy or rent a complete container-as-a-service (CaaS).

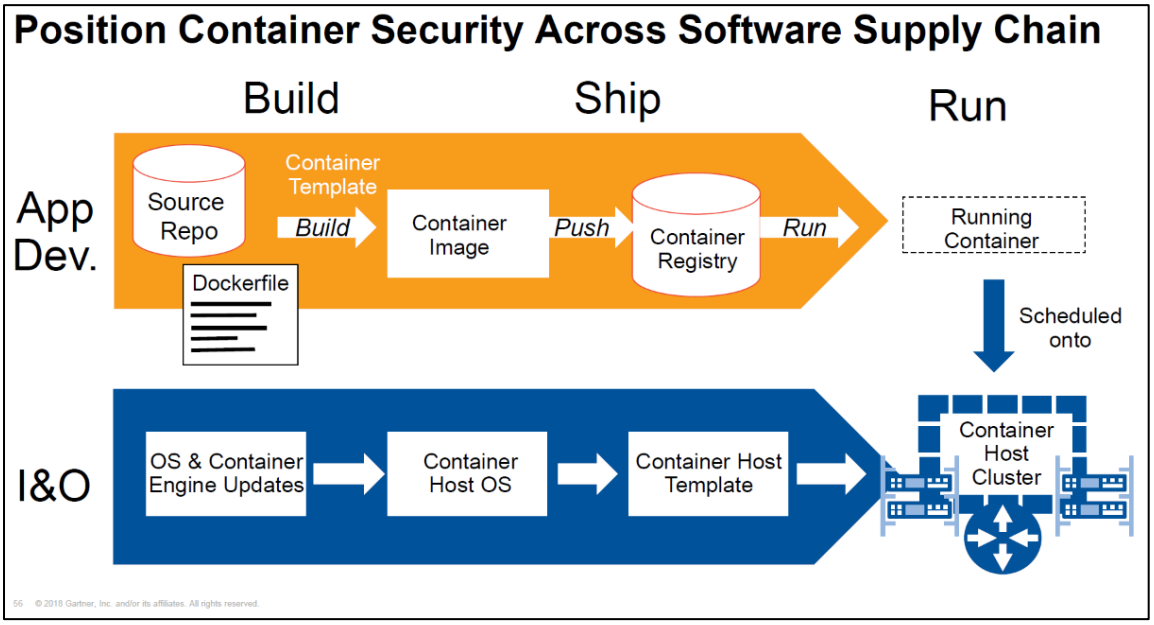




- Myth No-6: Containers are faster than VMs.
  - Choose containers for use cases where faster startup and spin-down matter.
    - Containers are 49x faster to reboot than VMs per IBM in 2014.
    - Guest apps (bare-metal, docker, and kvm) in containers perform near identically in CPU-bound testing per IBM in 2014 and RHEL benchmark data in 2016.



- Myth No-8: Containers are not secure.
  - Plausible – apply security controls in build, ship, and run phases.
    - ADP uses Docker because of their security requirements not despite them.



Choosing the Right Container Infrastructure for your Organization. [21]

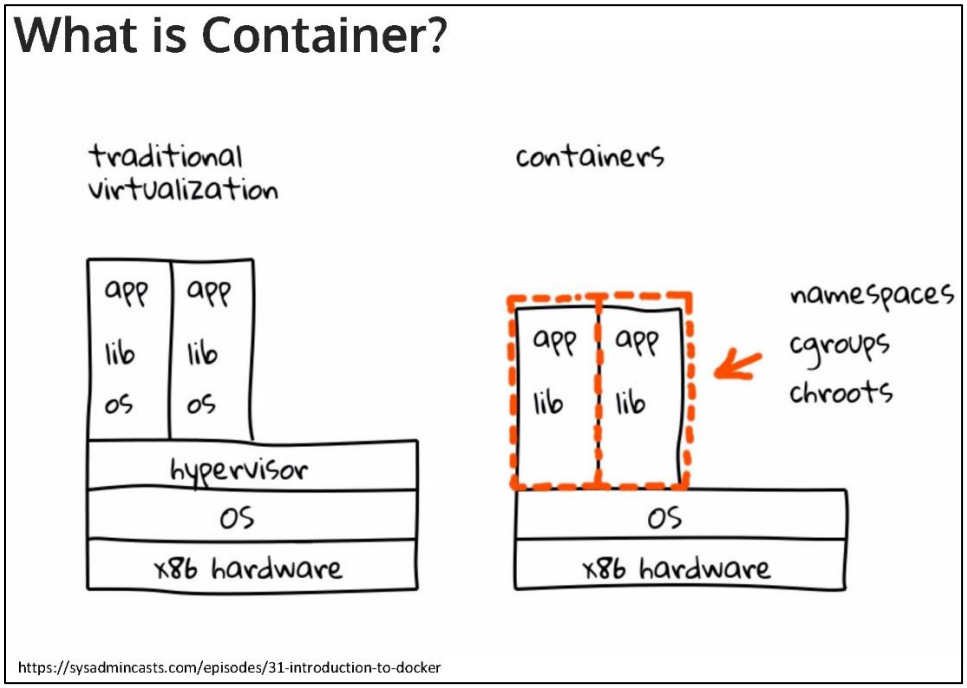
- Container adoption is accelerating rapidly with Gartner predicting more than 50% of new workloads will be deployed into containers in 2018.
- Docker expects 30x growth of containerized apps in two years.
- Bare metal is the gold standard for production containers.
  - Why?
    - Bare-metal containers provide optimal performance, allowing applications to access hardware without the need for pass-through, or hardware emulation.
    - Bare-metal delivers many of the perceived advantages of virtualization including application portability and isolation.

[21] Choosing the Right Container Infrastructure for your Organization by Diamanti on April 12, 2018. Obtained from the internet in December 2018.

    - Running containers inside VMs is like doing virtualization on top of virtualization – totally unnecessary.
- Interestingly, existing IT infrastructure optimized over a period of many years for virtualized business applications may not efficiently support containers.
    - As you navigate the transition from virtualization, you'll need infrastructure that addresses the unique needs of a container environment.
    - A variety of vendors have created converged and hyperconverged infrastructure solutions to reduce the complexity of IT infrastructure deployment:
        - Converged infrastructure (CI) pre-packages several servers with a separate storage array.
        - Hyperconverged infrastructure (HCI) combines servers with internal storage, software to virtualize that storage, and virtualization software such as VMware vSphere.
    - These solutions can be deployed for use in container environments, however, as a class, they are designed for virtualization rather than containers, making bare-metal container deployment impossible in almost all cases.
        - The vendors themselves remain largely focused on virtualization.
        - The level of support you'll get from a vendor for everything in the infrastructure stack above virtualization could be minimal.
        - You may have to rely on the open-source community for container and orchestration support.
- Container solutions must provide a mechanism for persistent storage as containers come-and-go.
- Getting-networking-right remains one of the most difficult aspects of configuring container environments.
    - Networking documentation for technologies like Docker and Kubernetes run to hundreds of pages, suggesting much to think and plan for in this area.
- Dynamic container environments require orchestration tools to coordinate activities and automate operations – currently Kubernetes has emerged as the clear leader.
- Shameless white-paper marketing – the Diamanti platform integrates everything—hardware and software—out of the box, so it can be fully deployed and operational in minutes.

- Open-source software, including Docker, CentOS, and Kubernetes, is pre-installed and ready to run containers, so there's no vendor lock-in.
- Scaling occurs through the addition of nodes to a cluster.
  - Because containers run on bare-metal, container density is extremely high; hardware utilization approaches 90%.

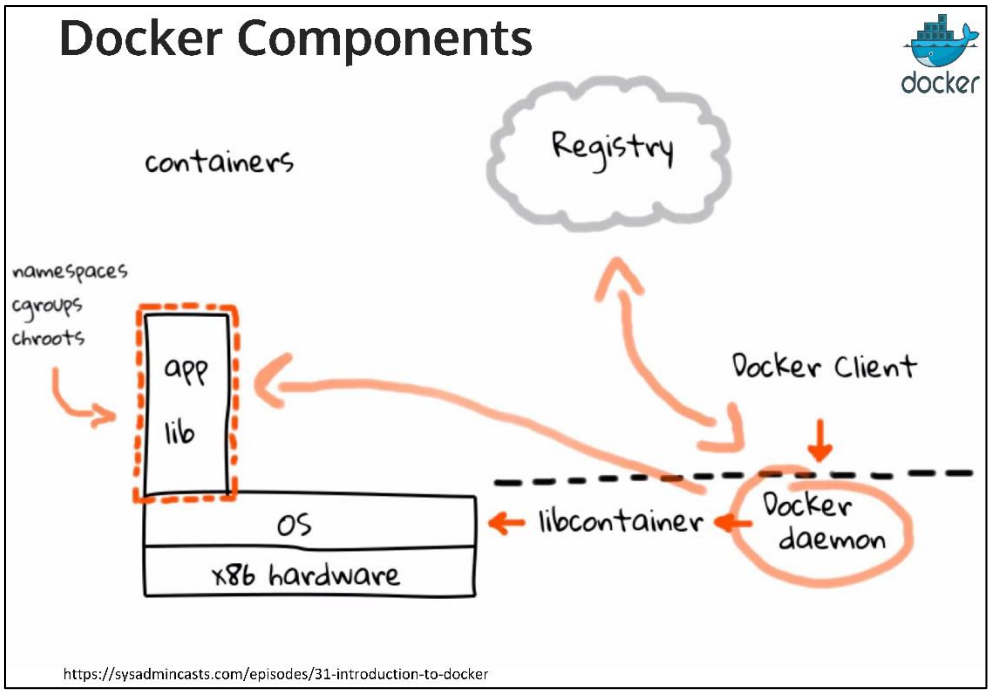
Future of Cloud Computing with Containers. [22]



- Containers can change, and so they have state.
  - A container may be running or exited.
  - When running, the idea of a "container" also includes a tree of processes running on the CPU, isolated from the other processes running on the host.
- When you exit a container, the state of the file system and its exit value is preserved.

[22] Future of Cloud Computing with Containers by WS02 Inc in September 16, 2017. Obtained from the internet in December 2018.

  - You can start, stop, and restart a container and the processes restart from scratch (their memory state is not preserved in a container).
  - The file system is just as it was when the container was stopped.
- You can promote a container to an Image with docker commit.
  - Then the image becomes a parent for new containers.



- Containers can start within milliseconds.
  - Full service could start within seconds.
  - Do we need up and running servers anymore?
    - Serverless Architecture: Amazon Lambda and Google Function.
- Containers scale fast.
- When running an application inside a container then can set the maximum memory and CPU the container is allowed to use.
  - When the container hits max, it will be destroyed and a fresh container will start – system auto healing.
- Never have to upgrade your container or your code.
  - Instead create a new container and throw away the old one!
  - Rollback is simply bringing back the old container.
  - No need for incremental updates.

Pictorial Insight of Virtualization and Containerization Technologies [23]

[23] Obtained through various sources as found on the World Wide Web / Internet. December 2018.

Overview

- The following graphics are provided to assist with understanding virtualization and containerization.
    - It is understood that some graphics may not help one's understanding at all.
    - Viewer discretion is advised.

Virtualization Insights

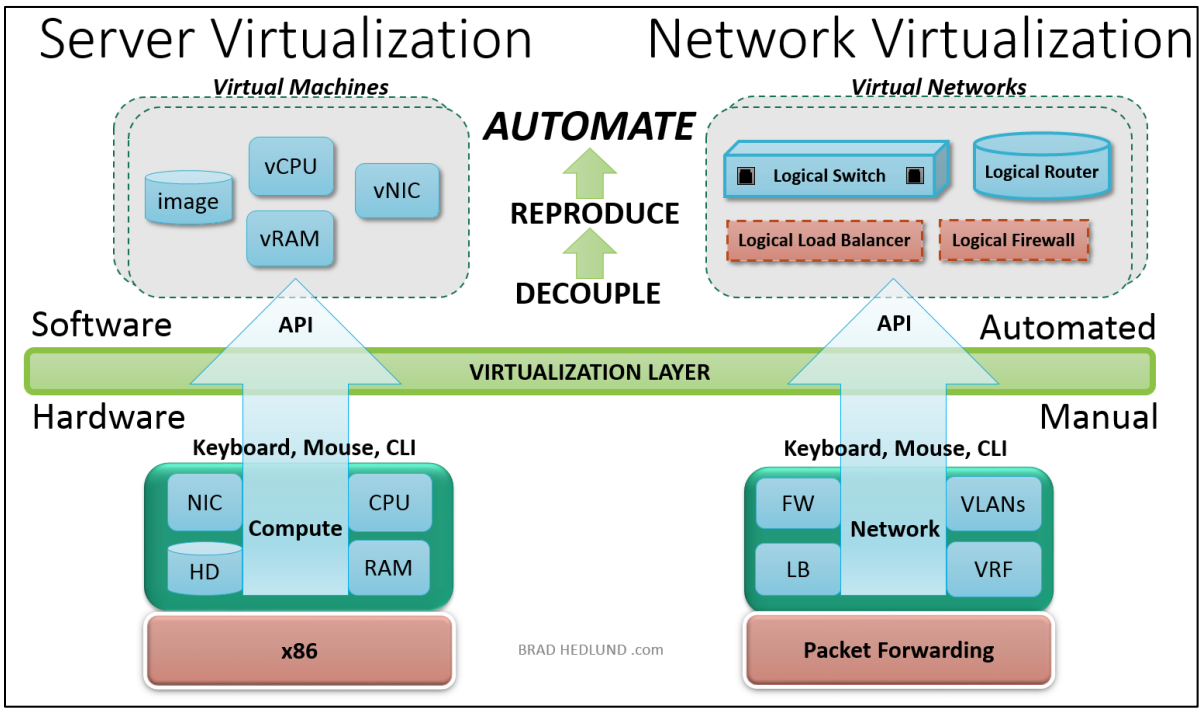
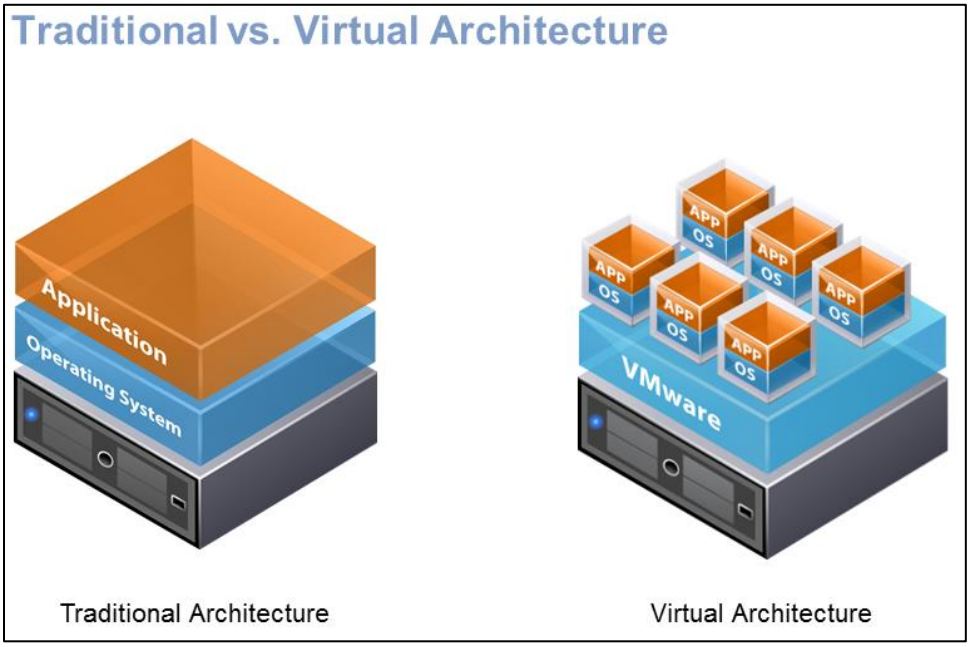


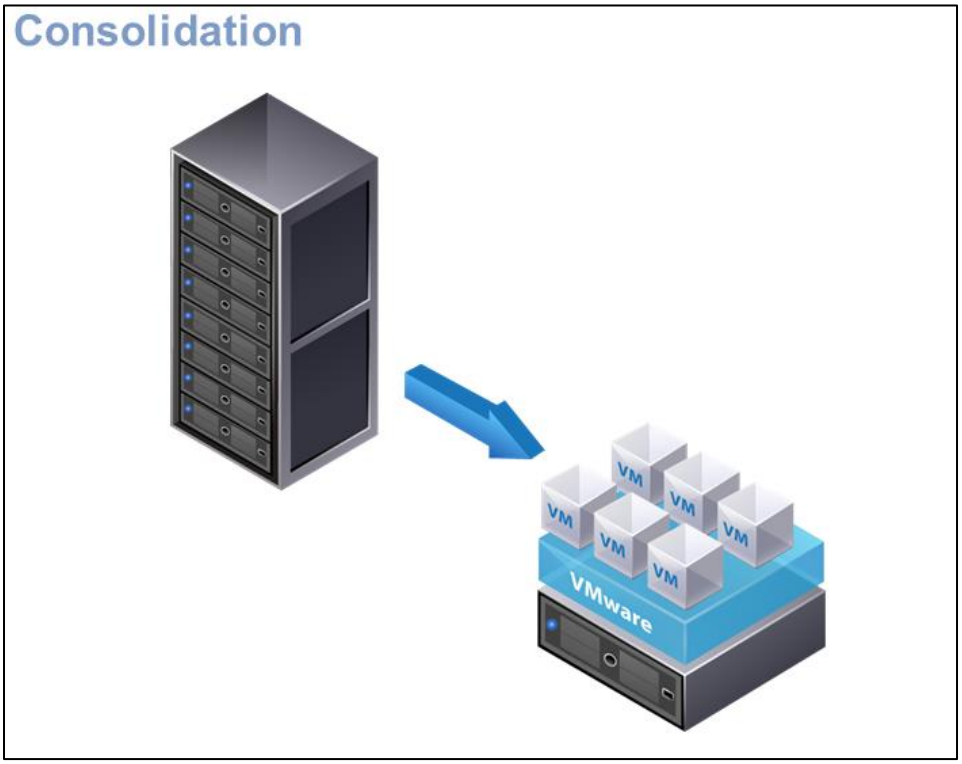
Figure 5 – Physical to Virtual Operations Transformation

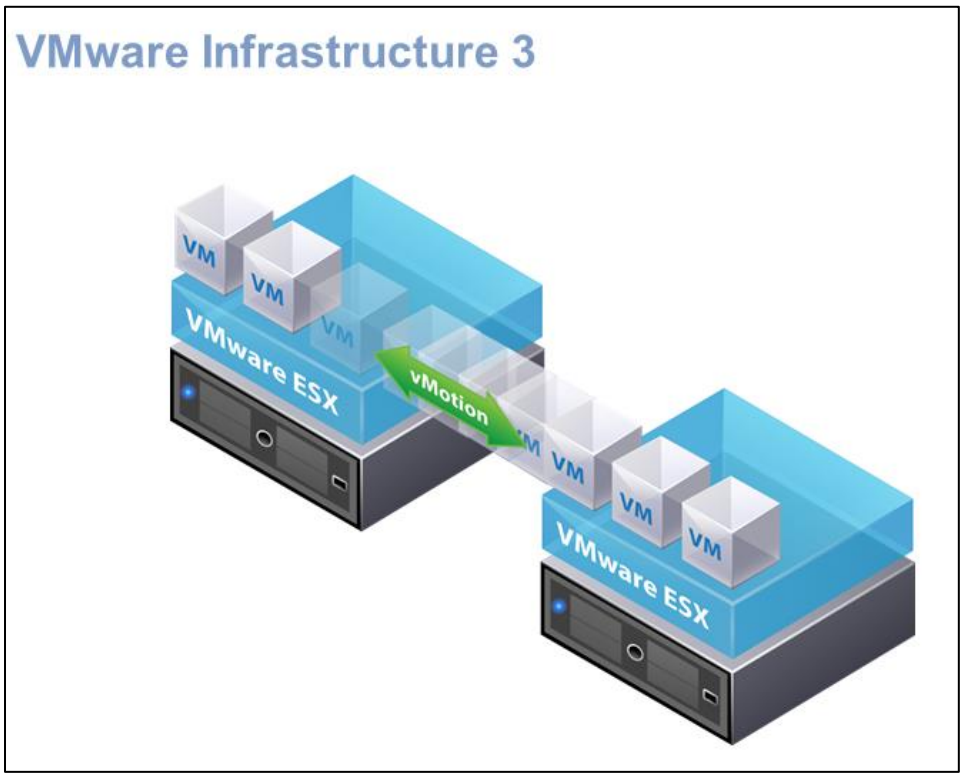
Figure 1. Magic Quadrant for x86 Server Virtualization Infrastructure

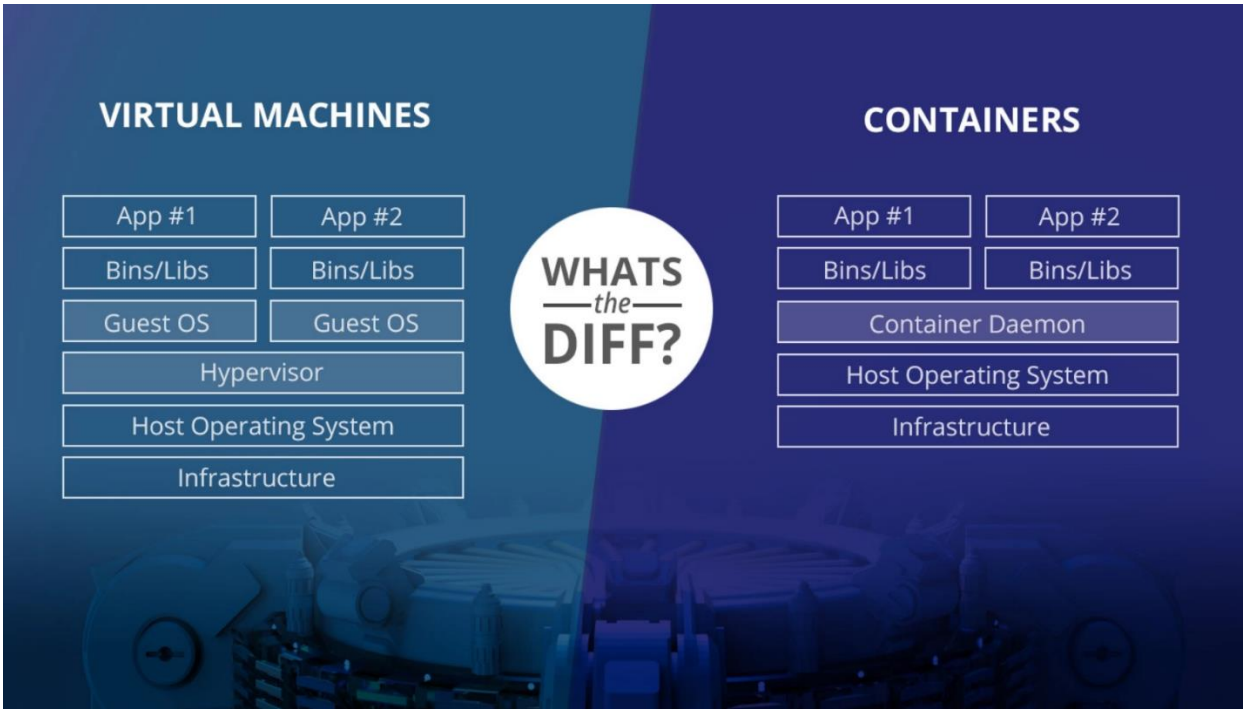
Source: Gartner (August 2016)

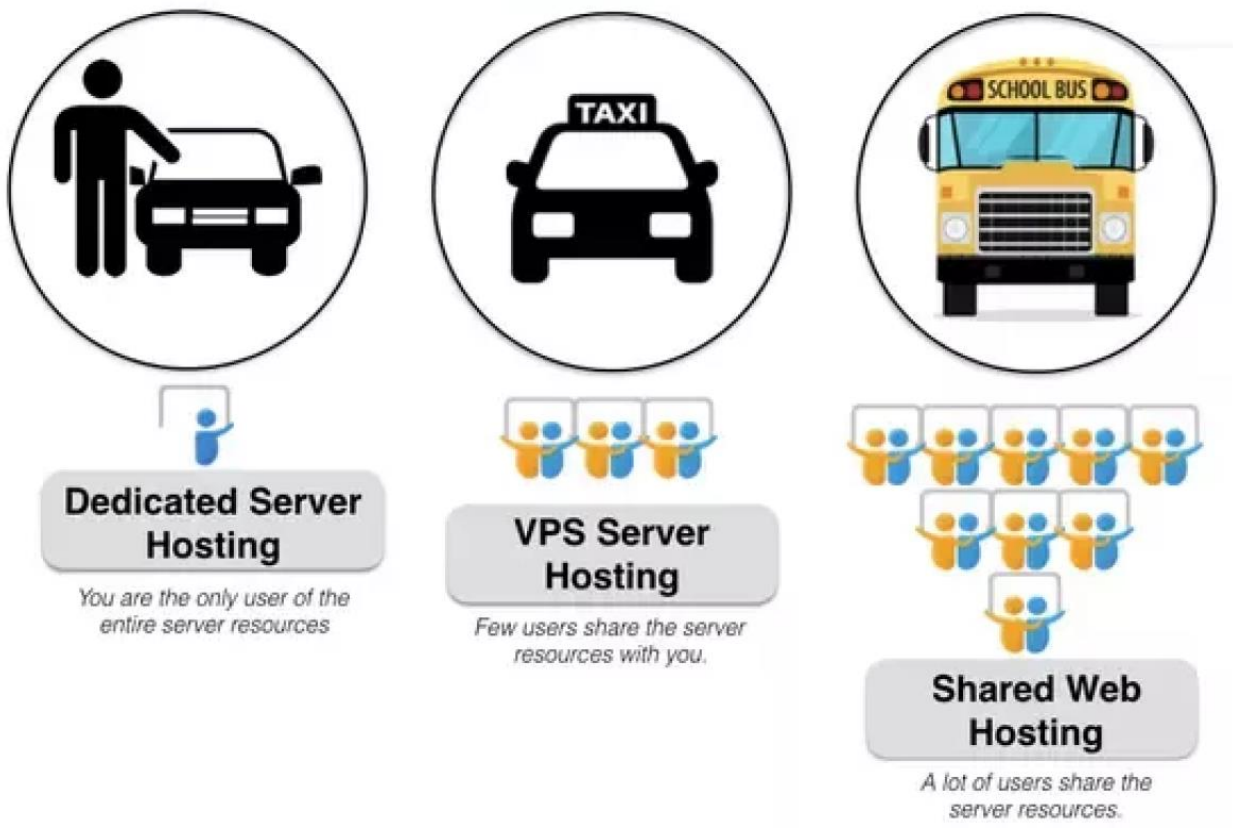
Source: Magic Quadrant for x86 Server Virtualization Infrastructure, Authors: Thomas J. Bittman, Philip Dawson, Michael Warrilow, Aug. 3, 2016

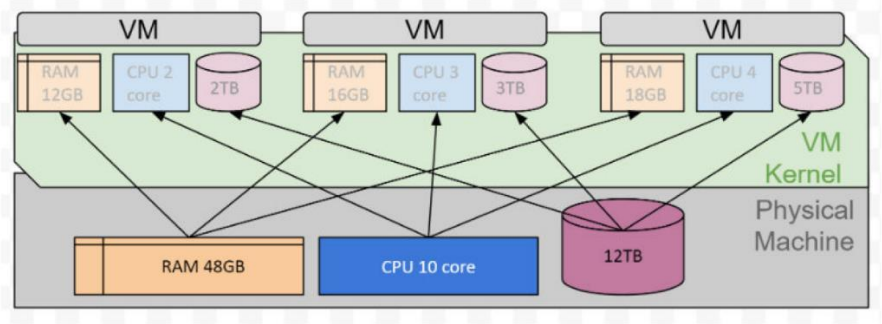

Traditional vs. Virtual Architecture
Application
Operating System
APP
OS
VMware
Traditional Architecture
Virtual Architecture


Consolidation
VM
VMware

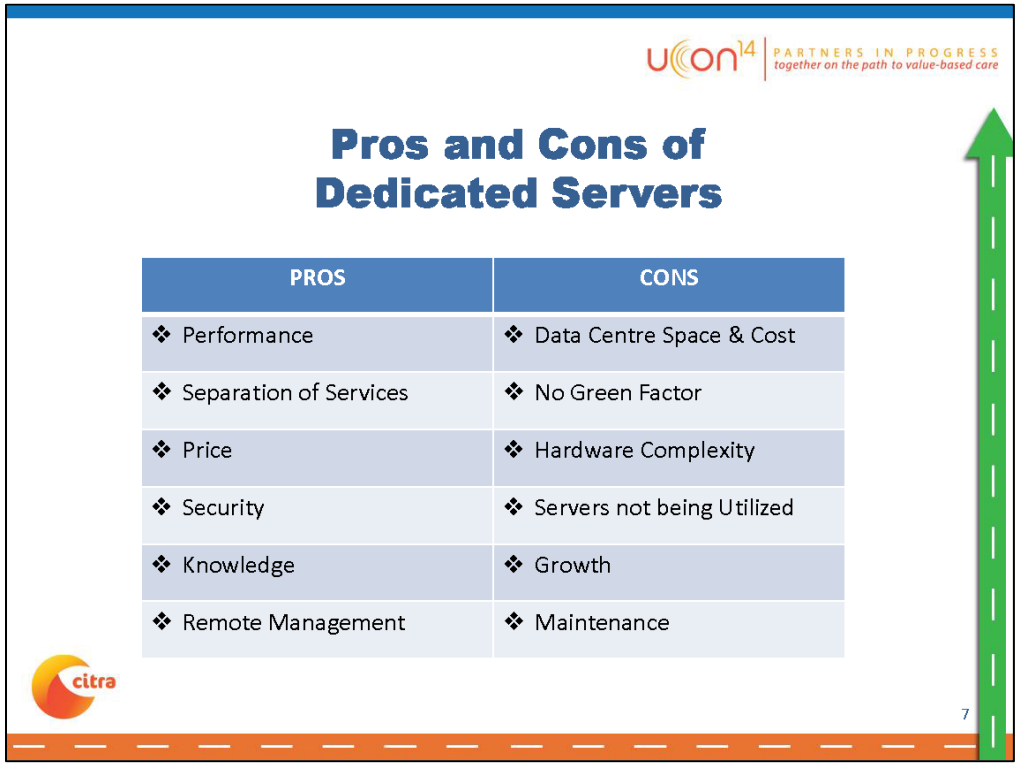

VMware Infrastructure 3
VM
VM
VM
VMware ESX
vMotion
VM
VM
VM
VM
VMware ESX

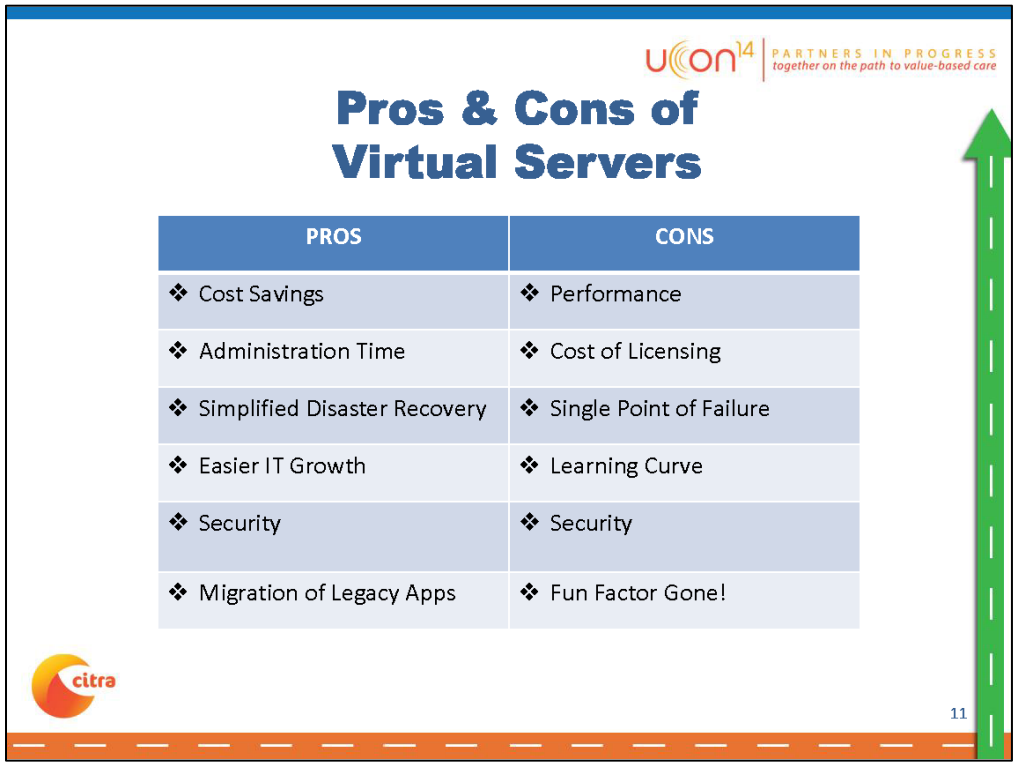

VIRTUAL MACHINES
App #1
App #2
Bins/Libs
Bins/Libs
Guest OS
Guest OS
Hypervisor
Host Operating System
Infrastructure
WHATS the DIFF?
CONTAINERS
App #1
App #2
Bins/Libs
Bins/Libs
Container Daemon
Host Operating System
Infrastructure





Virtual machines exist when a pool of computer resources are divided among a number of systems (operating systems, or specialized software) that "act" as if each of them was a complete computer server.

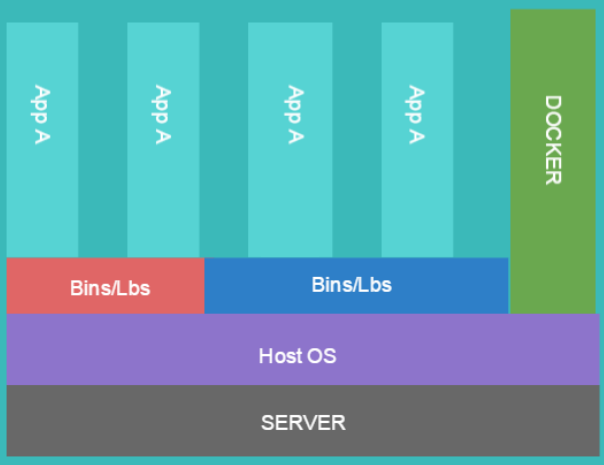

UCON14 PARTNERS IN PROGRESS
together on the path to value-based care
Pros and Cons of Dedicated Servers
PROS
Performance
Separation of Services
Price
Security
Knowledge
Remote Management
CONS
Data Centre Space & Cost
No Green Factor
Hardware Complexity
Servers not being Utilized
Growth
Maintenance
citra
7


UCON14 PARTNERS IN PROGRESS
together on the path to value-based care
Pros & Cons of Virtual Servers
PROS
Cost Savings
Administration Time
Simplified Disaster Recovery
Easier IT Growth
Security
Migration of Legacy Apps
CONS
Performance
Cost of Licensing
Single Point of Failure
Learning Curve
Security
Fun Factor Gone!
citra
11

How Are Containers Different From VMs?

Virtual Machines (VMs) virtualize the hardware; every VM carries its own OS on top of the hypervisor and host OS. Containers virtualize the OS – every container has its own CPU, memory, block I/O, network stack, but shares the same kernel as other containers on the same host.

Virtual Machines



Containers & Docker

What's the Diff: VMs vs Containers

VMs	Containers
Heavyweight	Lightweight
Limited performance	Native performance
Each VM runs in its own OS	All containers share the host OS
Hardware-level virtualization	OS virtualization
Startup time in minutes	Startup time in milliseconds
Allocates required memory	Requires less memory space
Fully isolated and hence more secure	Process-level isolation, possibly less secure

For most, the ideal setup is likely to include both. With the current state of virtualization technology, the flexibility of VMs and the minimal resource requirements of containers work together to provide environments with maximum functionality.

Explaining the deployment of apps within containers is made easier by comparing it with the deployment of apps within virtual machines (VMs) from hardware virtualization technologies, which many readers are already familiar with. Figure 2 shows the VM deployment on the left, a container deployment without VMs (installed on "bare metal") in the middle, and a container deployment that runs within a VM on the right.

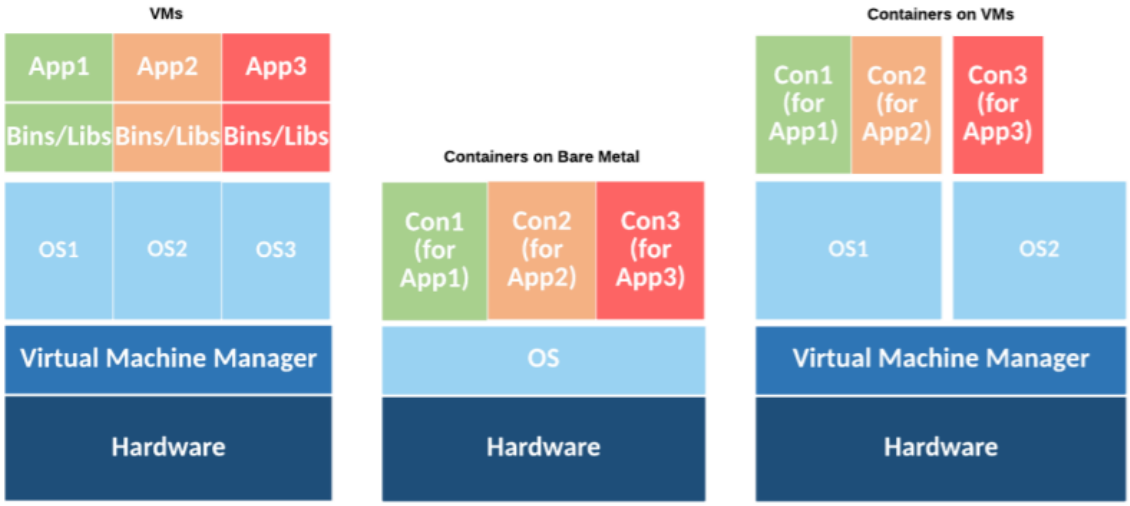
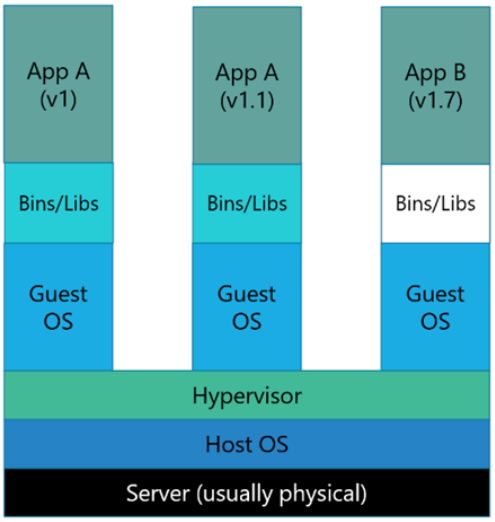


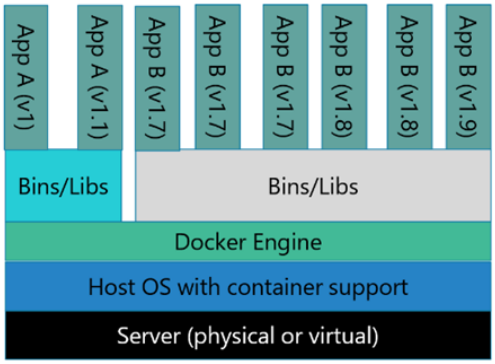
Figure 2: Virtual Machine and Container Deployments

NIST on Containerization

Server Virtualisation: Each app and each version of an app has dedicated OS



Containers: All containers share host OS kernel and appropriate bins/libraries



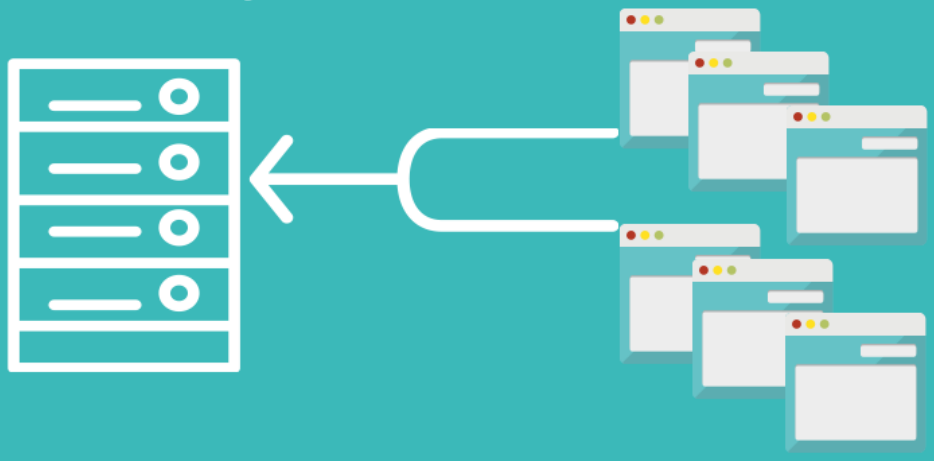
Containerization Insights

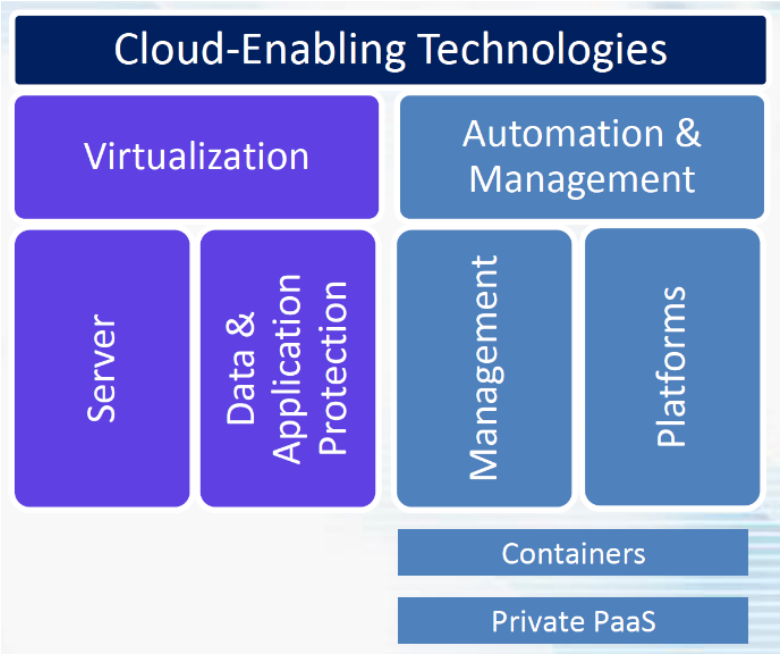
What Are Containers?

Virtual containers have roots in FreeBSD jails and Solaris Zones. Linux Containers – LXC – established containers as a cloud virtualization technology. Docker added unprecedented portability and propelled container to the forefront of cloud computing.

What Is the Purpose of Containers?

Virtualization: so you can run many different workloads on the same host and move them around as much as you wish.





Virtualization automation containers

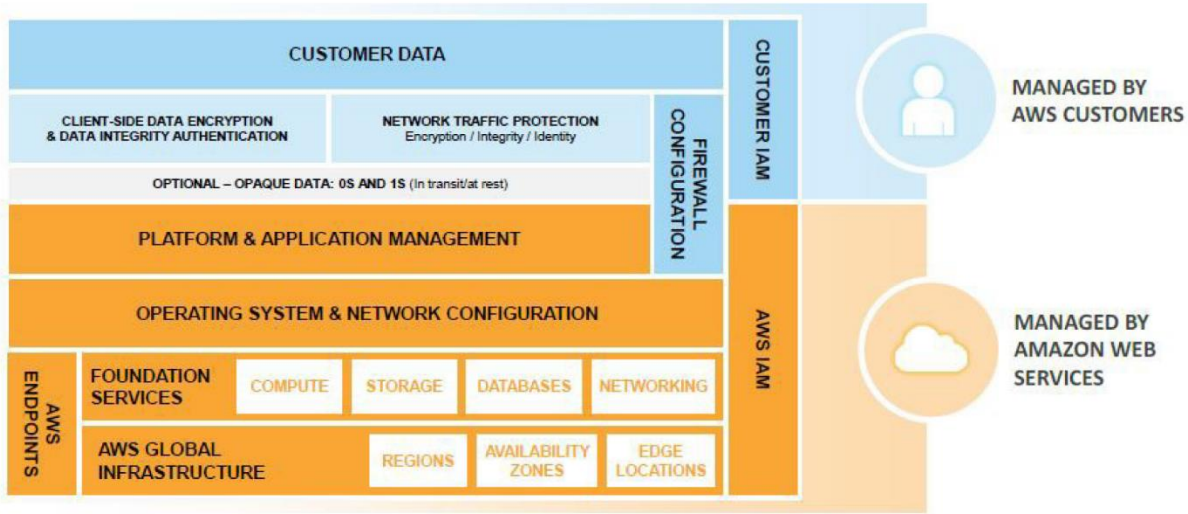
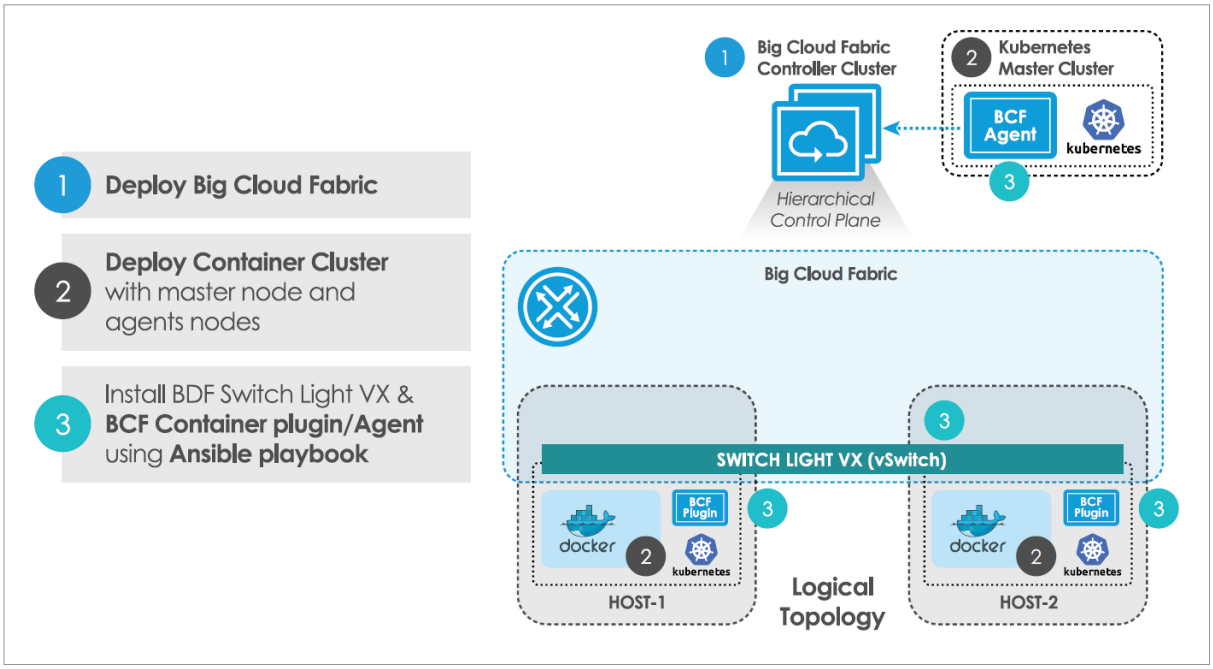
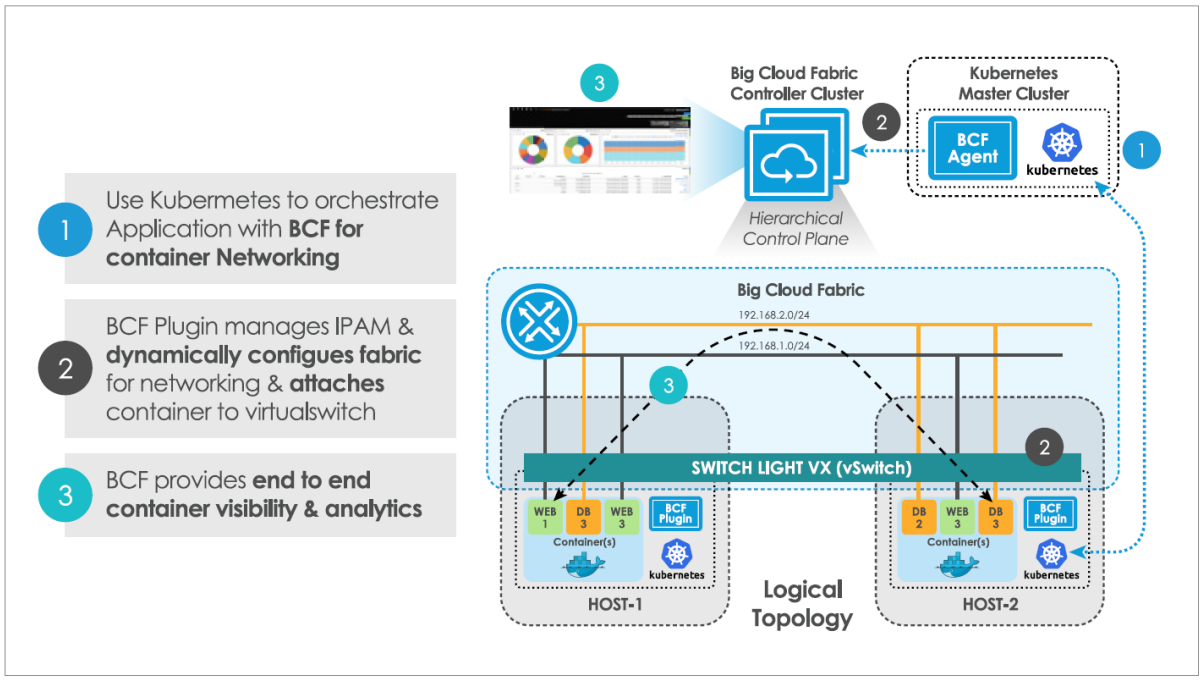


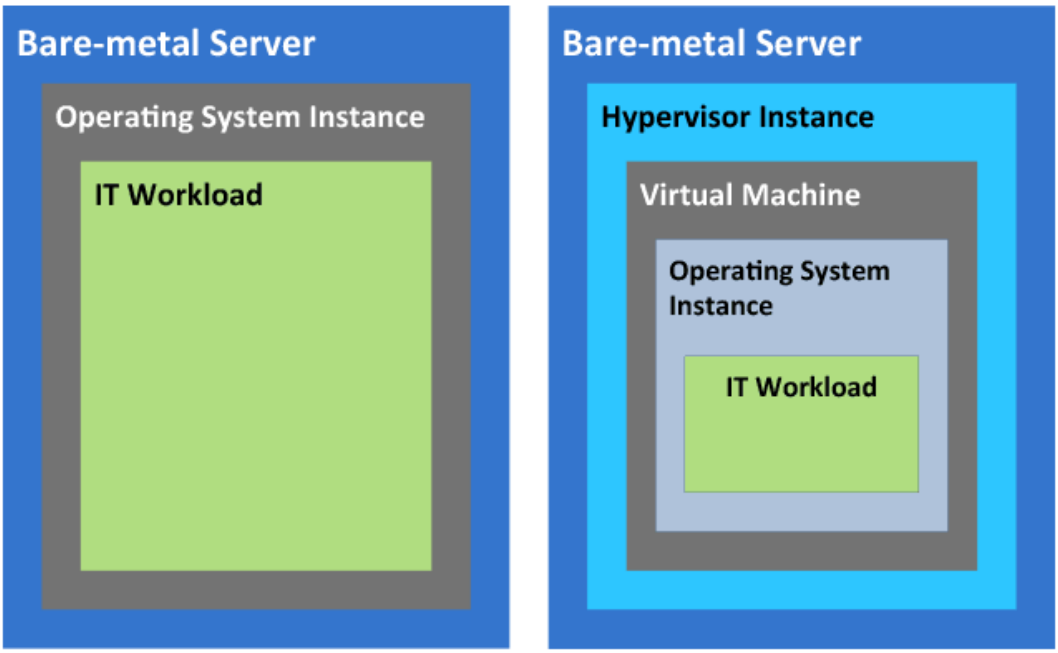
Figure 2: Shared Responsibility Model for Container Services



Big cloud fabric BCF three simple steps to deploy a BCF container solution January 2017.



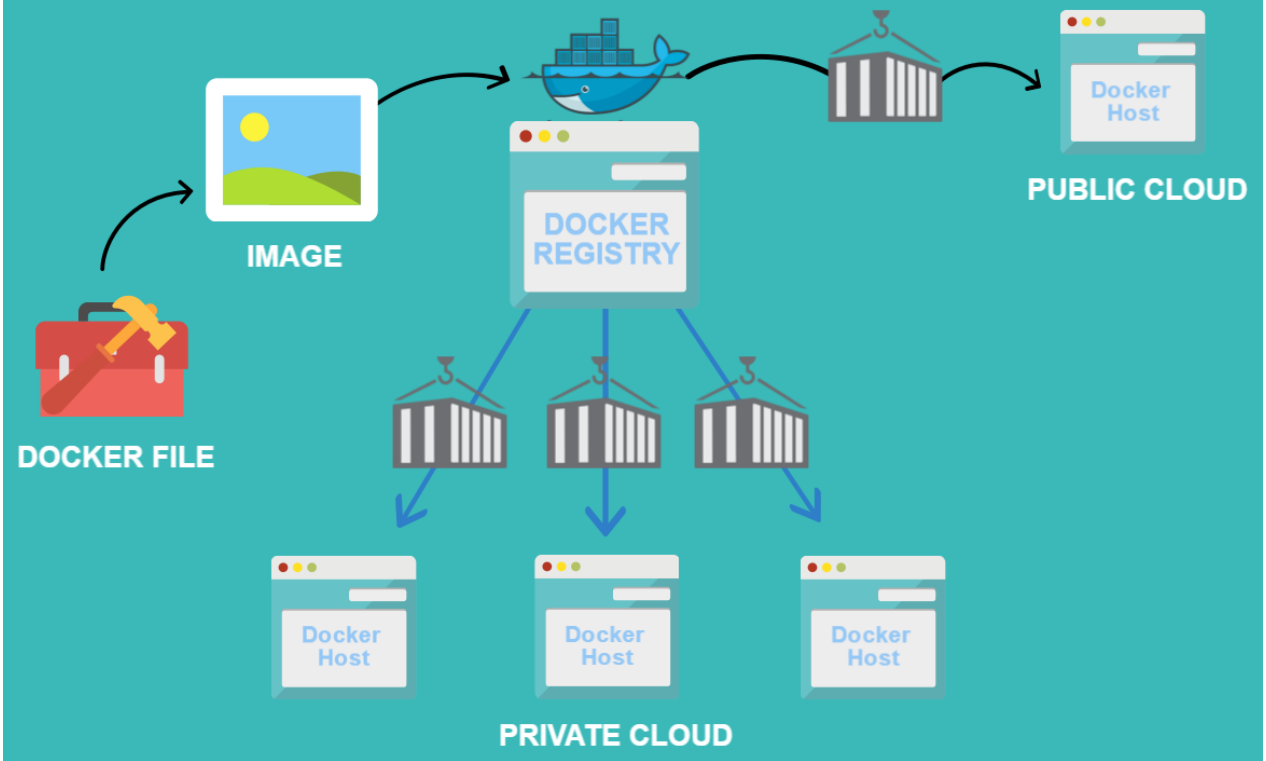
Big cloud fabric BCF container integration benefits – January 2017.

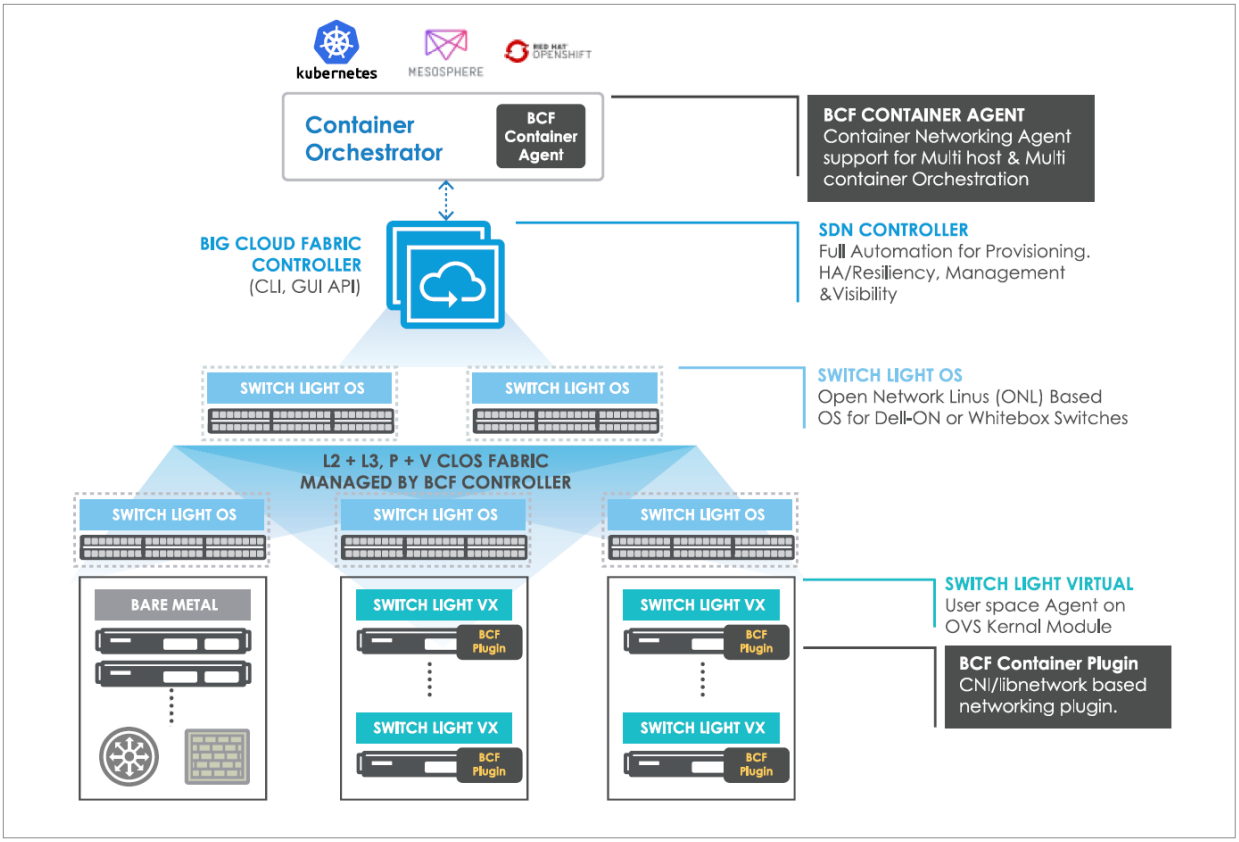


Conceptual view diagram of IT workload container.

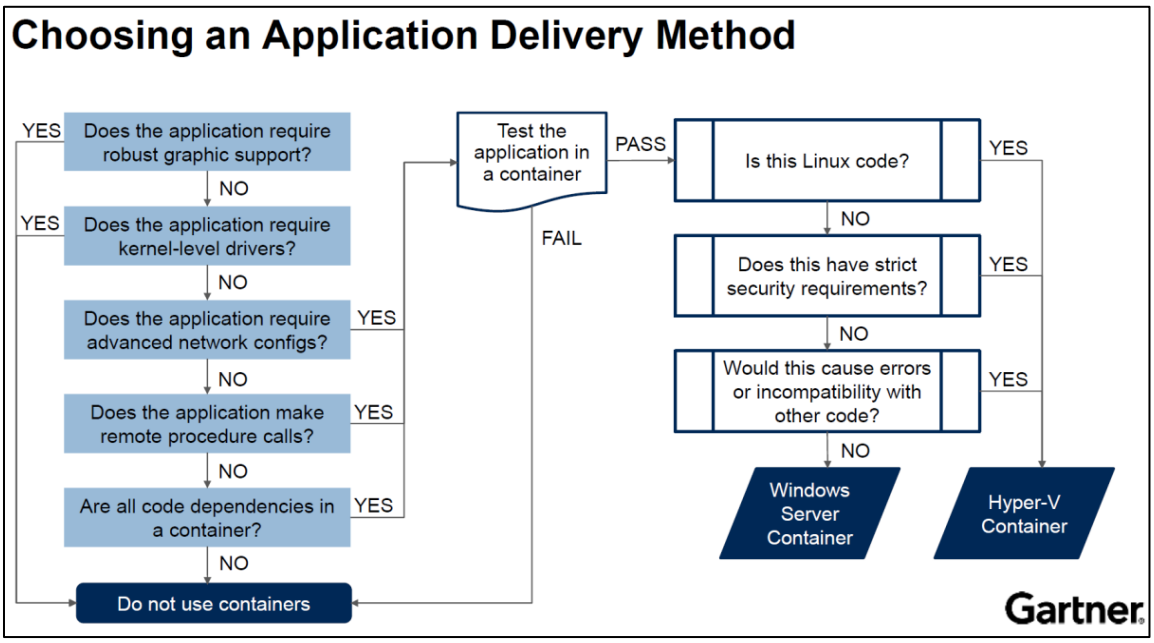
What is Docker Registry?

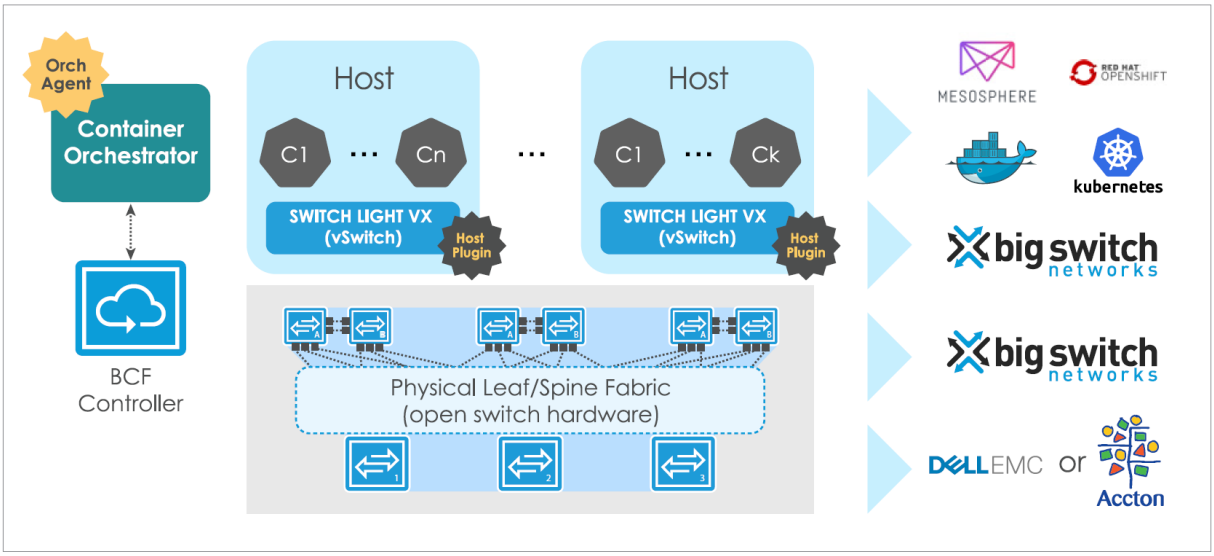
The concept of registry is central to Docker ecosystem. A registry is a place where you store and share container images. A user can publish images into a registry and other users can pull images from it to deploy.





Big cloud fabric delivers container-ready networking – January 2017





BCF's container solution leveraging container plug-ins on the host and agent on container orchestrator from Gartner 2018.

In the container model (Figure 11-4), applications *share* a single instance of the operating system. Both Microsoft Windows and the various Linux distributions have been enhanced to support the isolation required to ensure that each application appears to "own" the OS. Applications are "packaged" to be deployed on container-capable systems.

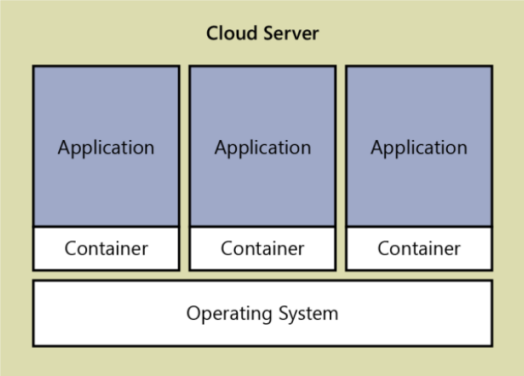
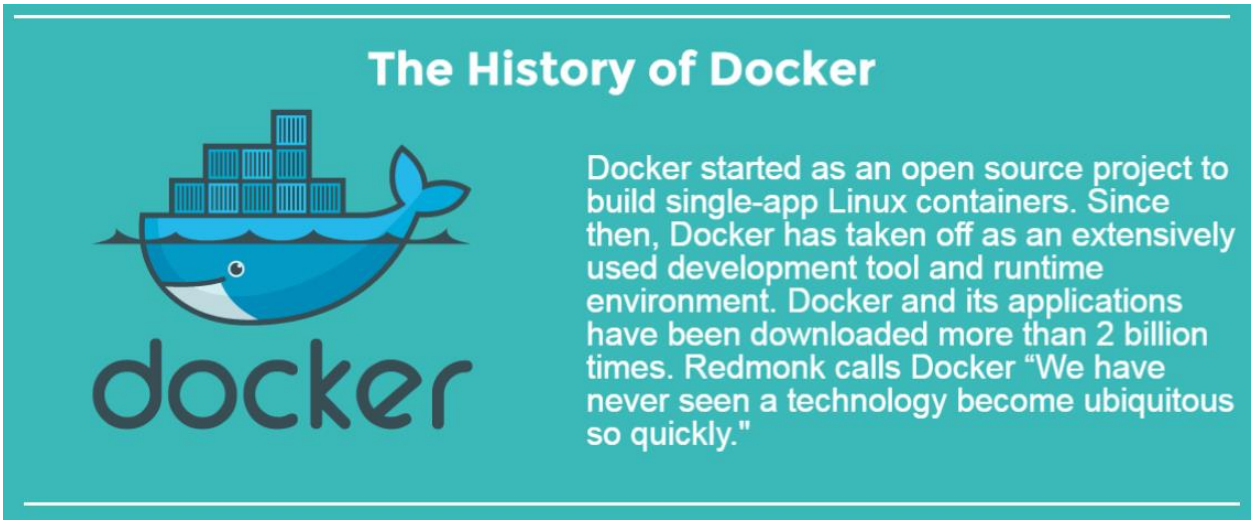


Figure 11-4: Container architecture

One result is that application startup is considerably faster because the overhead of loading an entire operating system for each app is avoided. Another is that you can create standard, portable packages, or *images*, such as an image for a web server or for a database, and you can deploy these without complex installation.

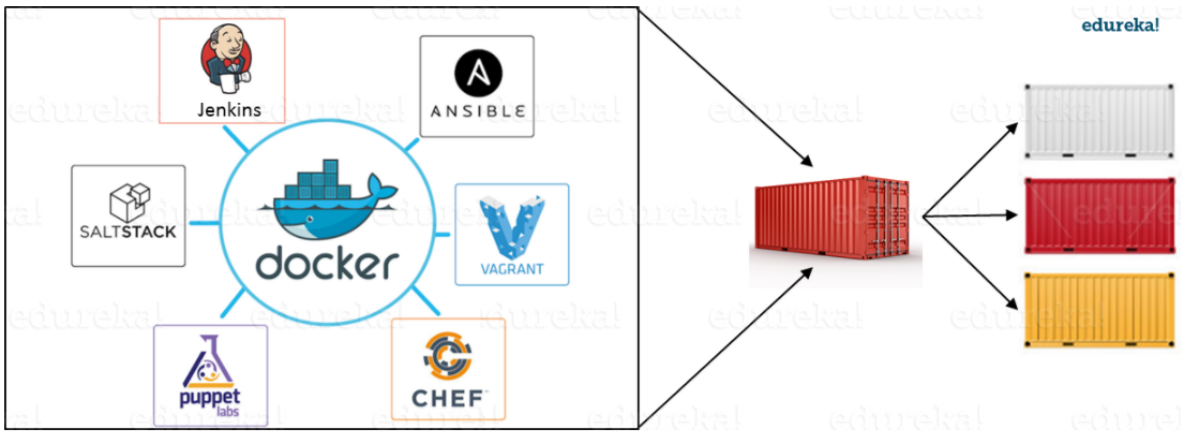
Another result is that containers achieve much more efficient utilization of the hardware because on a given server the number of actual operating system instances is limited (and might be only one).

Container architecture cloud strategy 2nd edition



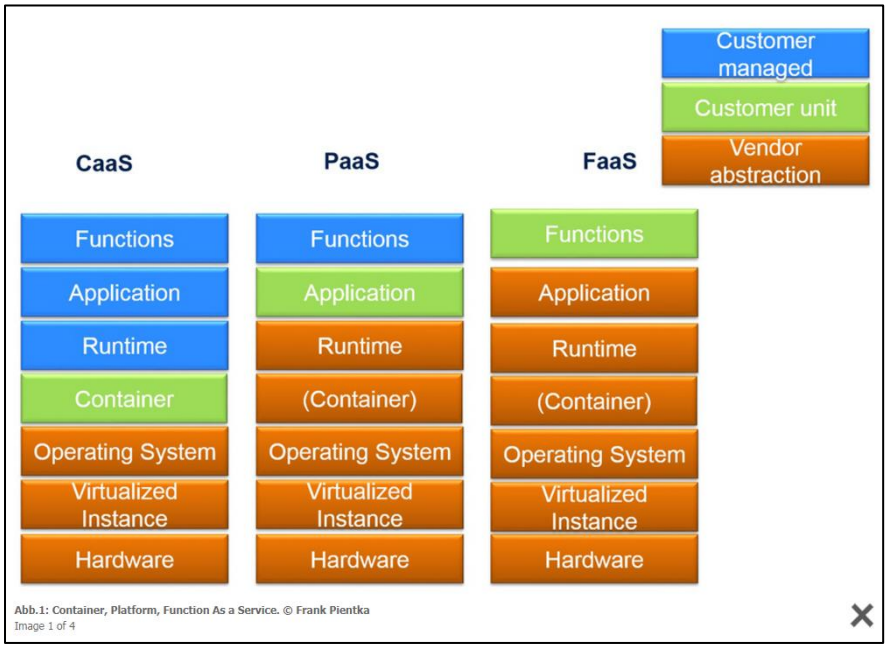
Containerization Tools

- Containerization tools are other set of tools which help in **maintaining consistency** across the environments where the application is developed, tested and deployed. It eliminates any chance of errors/ failure in production environment by **packaging** and **replicating** the same dependencies and packages used in development/ testing/ staging environment.
- The clear winner here is **Docker**, which was among the first containerization tool ever. Earlier, this act of maintaining consistency in environments was a challenge because VMs and servers were used, and their environments would have to be managed manually to achieve consitency. ***Docker containers*** threw this challenge up above and blew it out of the water. (Pun intended!)

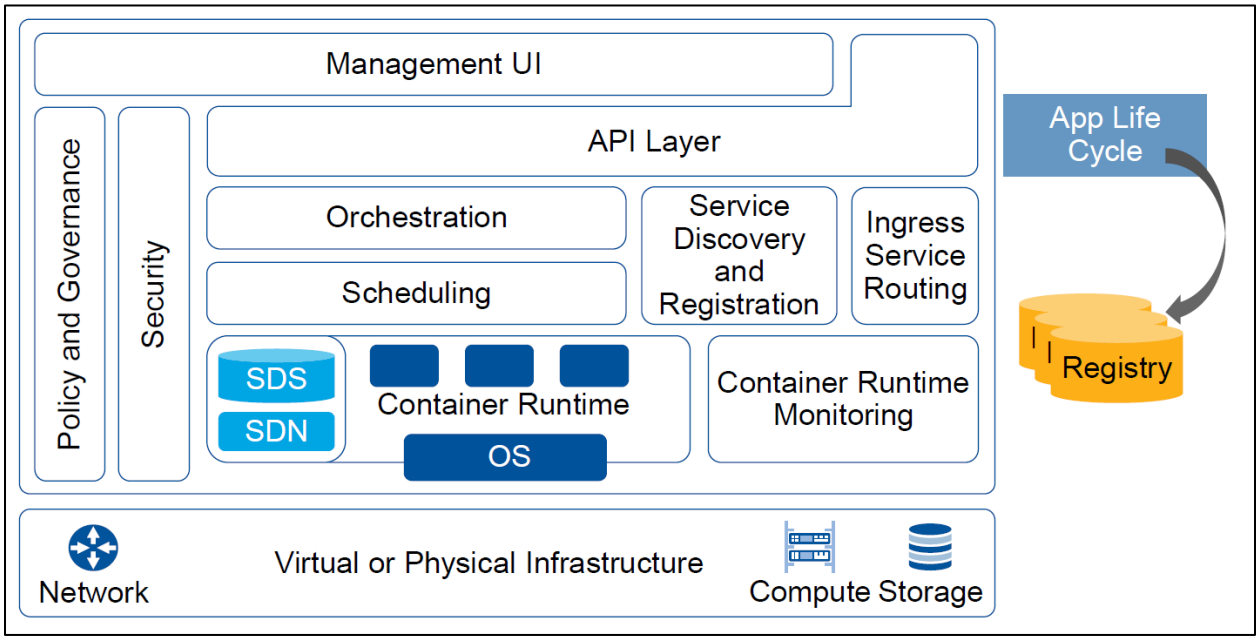


- Another containerization tool is **Vagrant**. But off-late, a number of cloud solutions have started providing support for container services. **Amazon ECS**, **Azure Container Service** and **Google Container Engine** are a few of the cloud services that have started radical support for Docker containers. This is the reason why Docker is the clear winner.

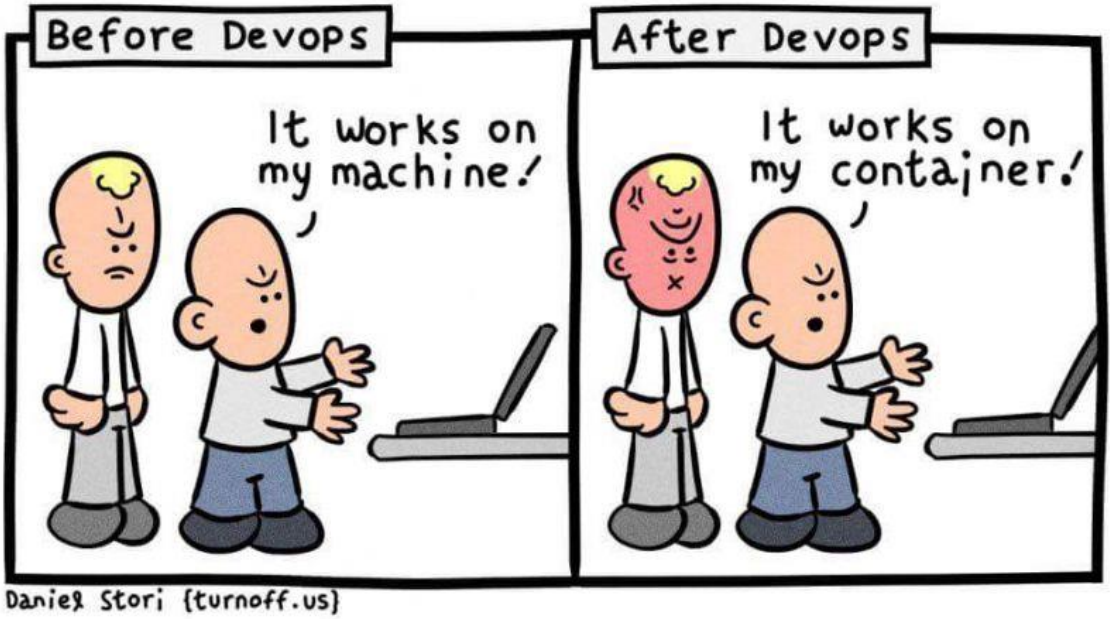
You can read more about Docker from here: *What is Docker?* So now, let's move on to the final topic in this DevOps tools blog.

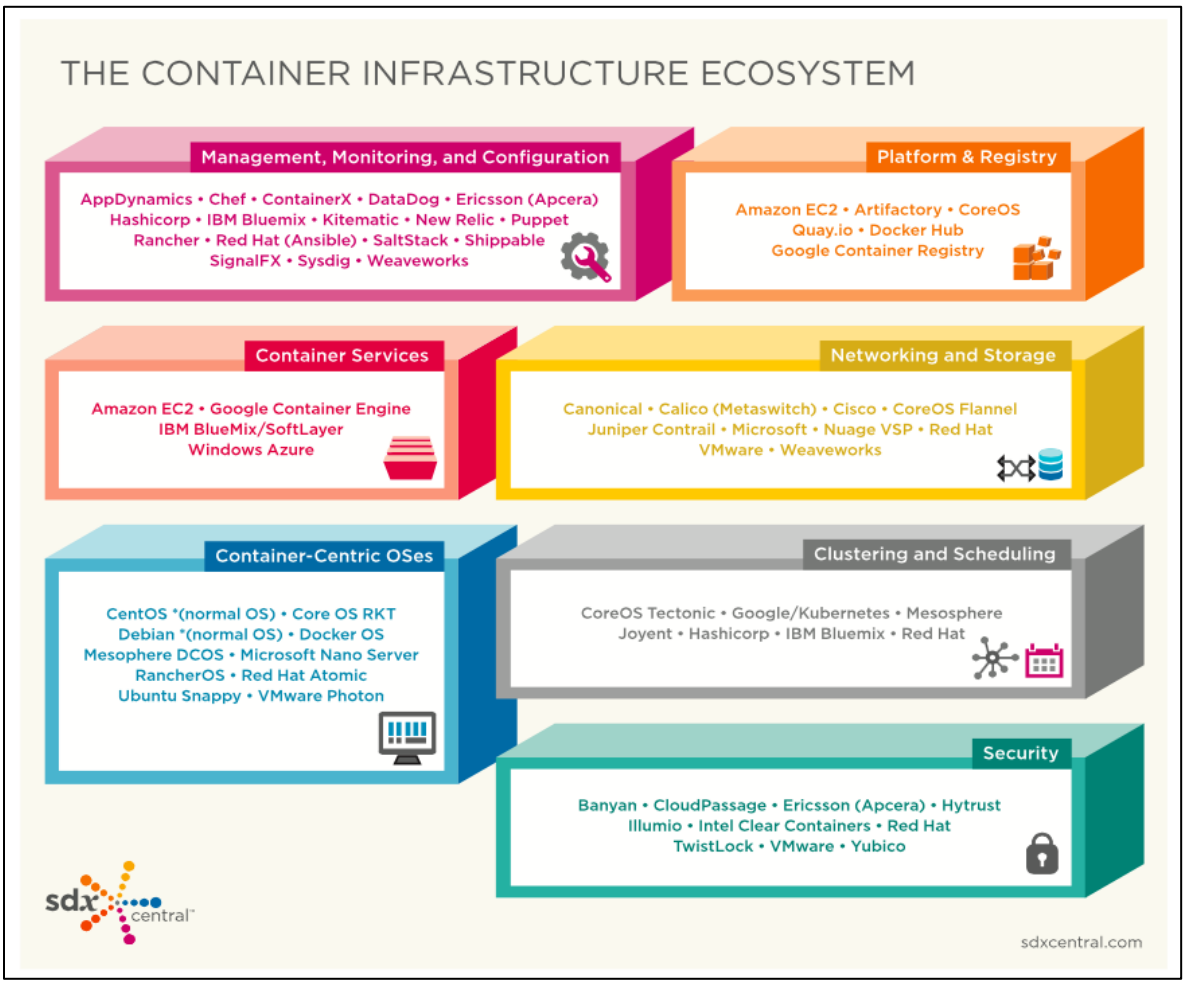


Container platform function-as-a-service



Container-as-a-Service (CaaS) Reference Model
Gartner 2018





Linux Container Infrastructure Ecosystem

Figure 3 shows an example of a container image. This image depicts an Ubuntu image with an Apache installation. The image is a composition of three base Ubuntu layers plus an update layer, with an Apache layer and a custom file layer on top.

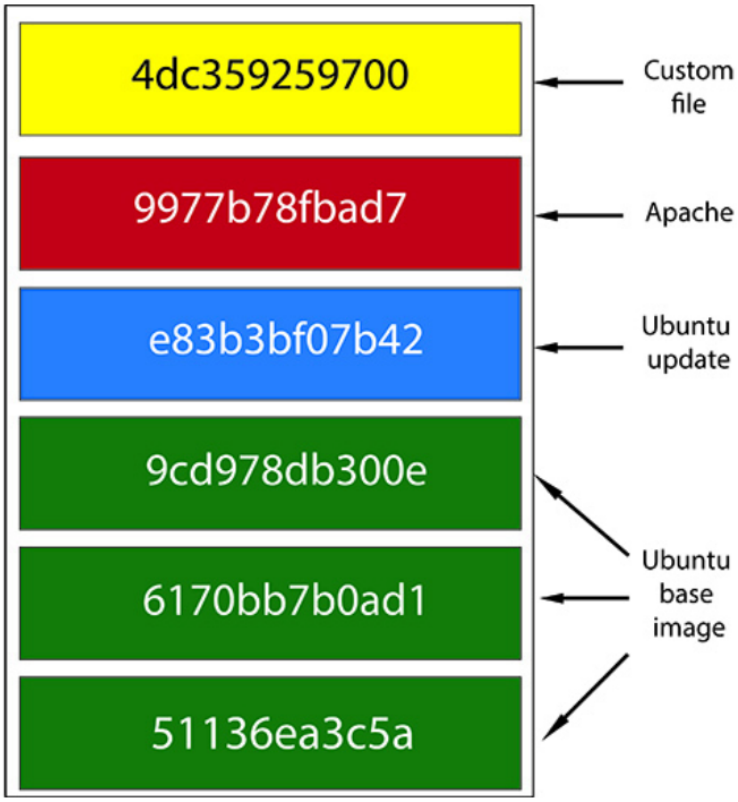
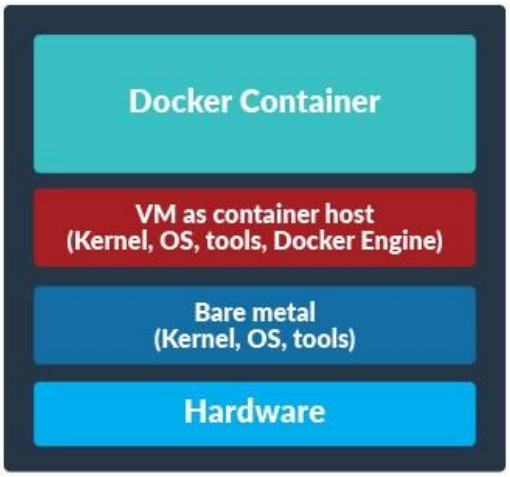


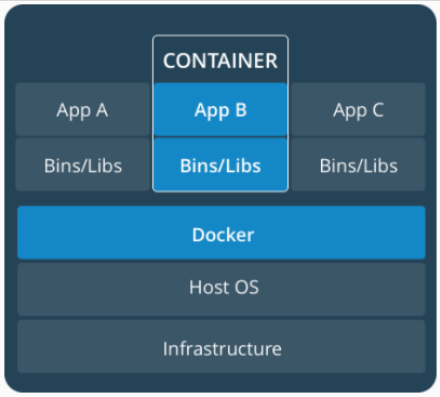
Figure 3: Docker image view

A running Docker container is an instantiation of an image. Containers derived from the same image are identical to each other in terms of their application code and runtime dependencies. But unlike images that are read-only, each running container includes a writable layer (a.k.a. the container layer) on top of the read-only content. Runtime changes, including any writes and updates to data and files, are saved in the container layer only. Thus multiple concurrent running containers that share the same underlying image may have different container layers.



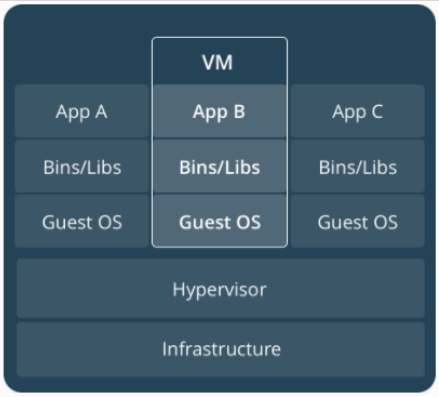
Docker Container Infrastructure

Bare Metal Container Infrastructure



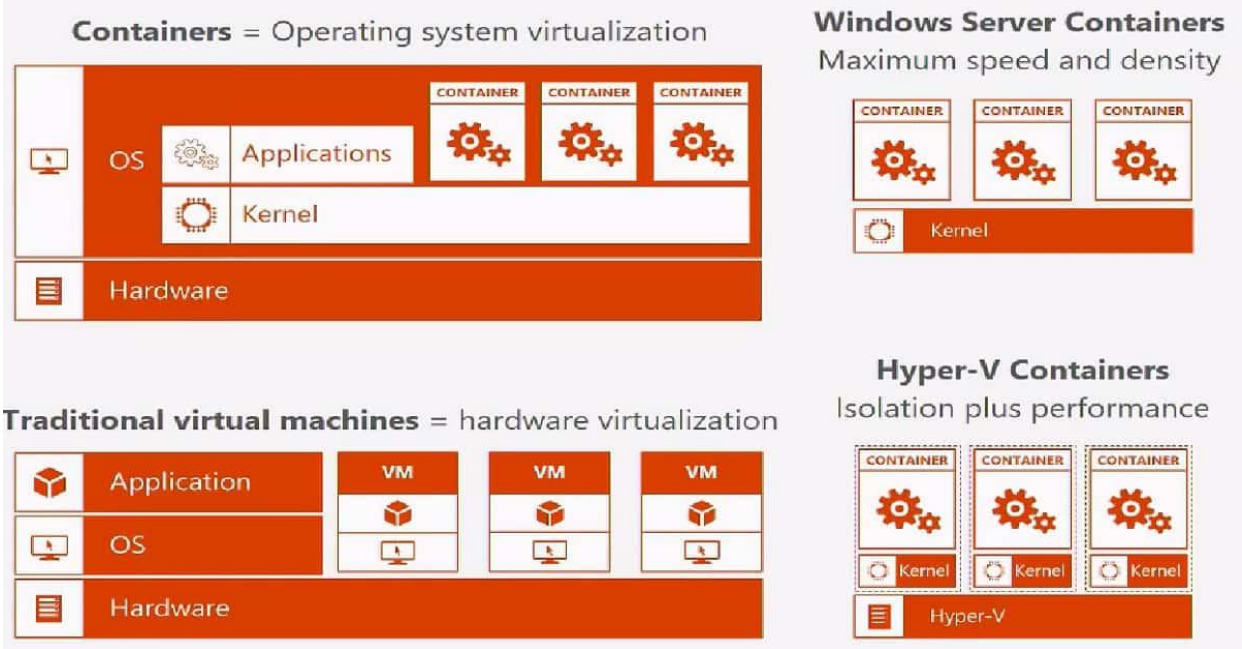
CONTAINERS

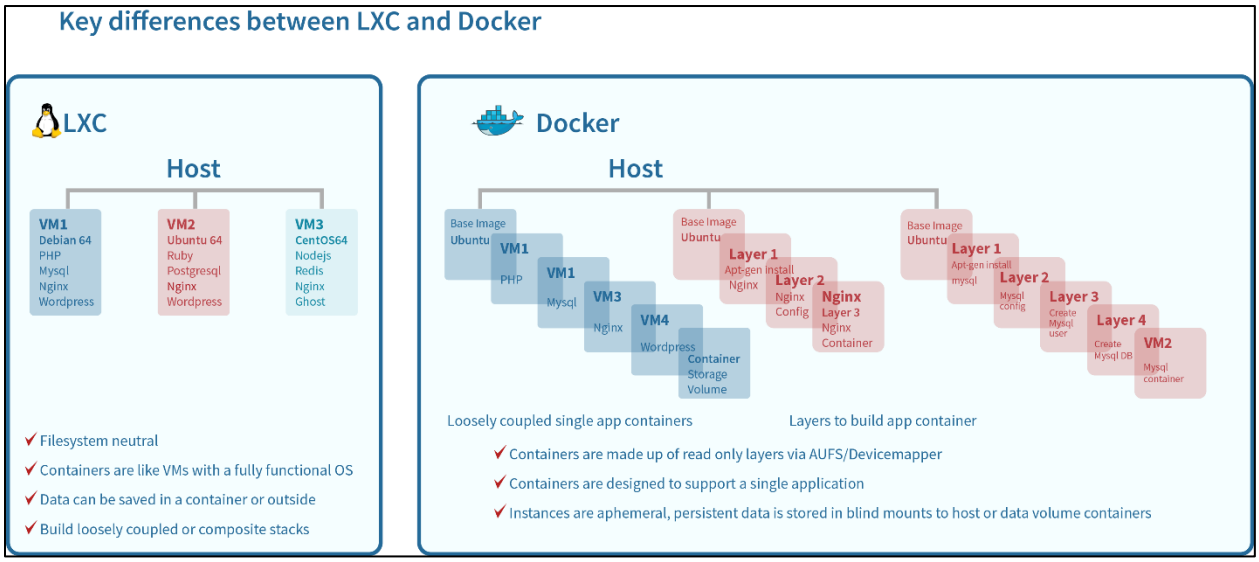
Containers are an abstraction at the app layer that packages code and dependencies together. Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space. Containers take up less space than VMs (container images are typically tens of MBs in size), and start almost instantly.

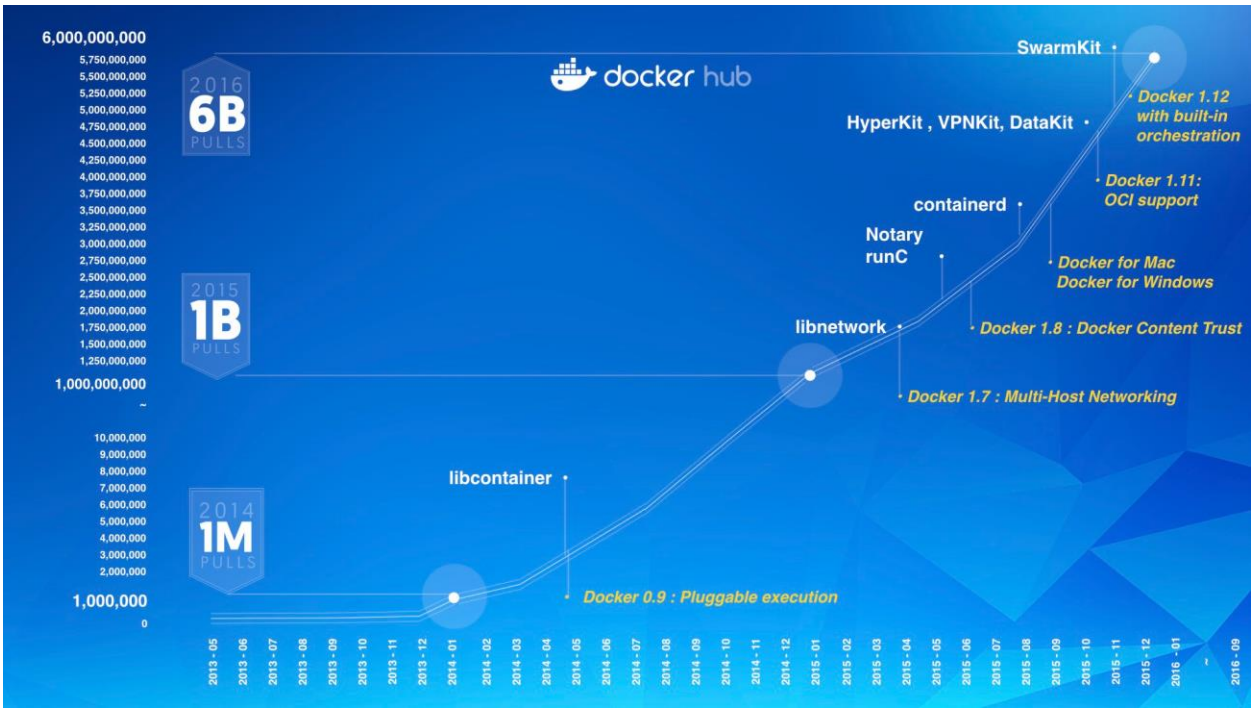


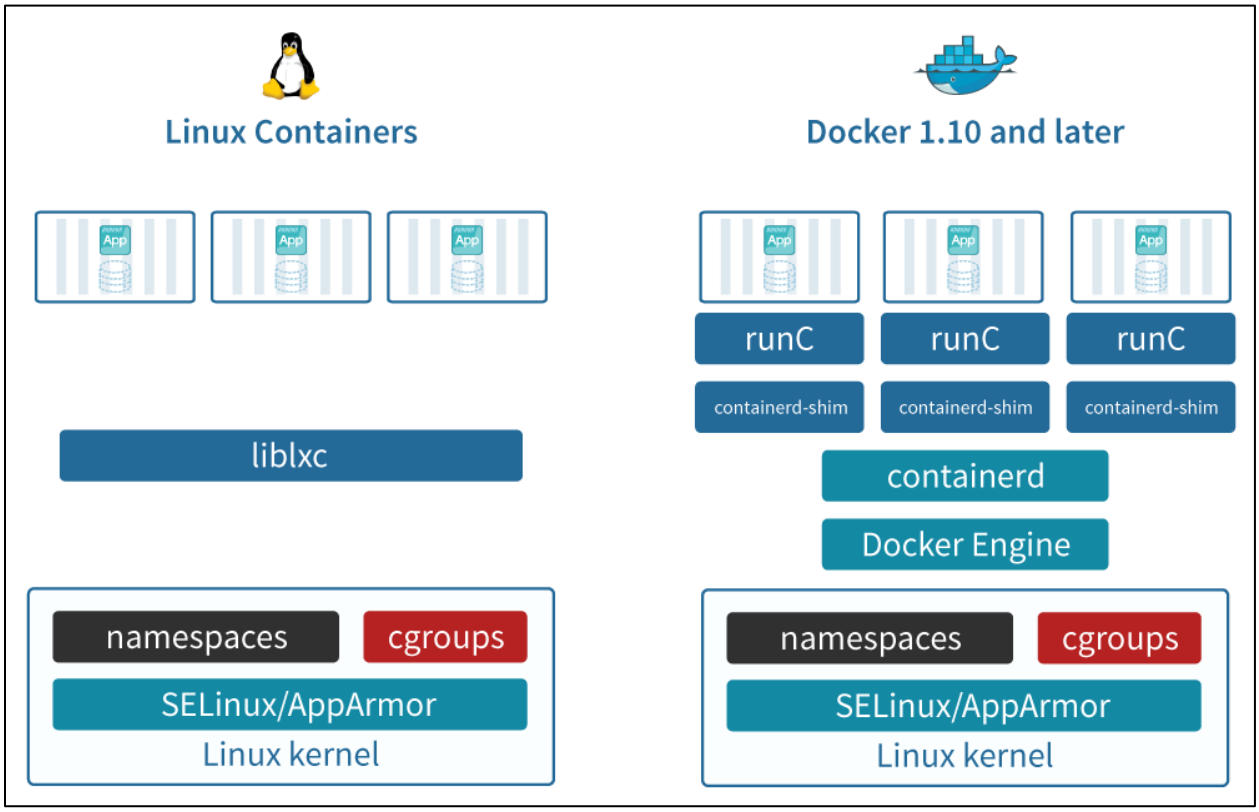
VIRTUAL MACHINES

Virtual machines (VMs) are an abstraction of physical hardware turning one server into many servers. The hypervisor allows multiple VMs to run on a single machine. Each VM includes a full copy of an operating system, one or more apps, necessary binaries and libraries - taking up tens of GBs. VMs can also be slow to boot.

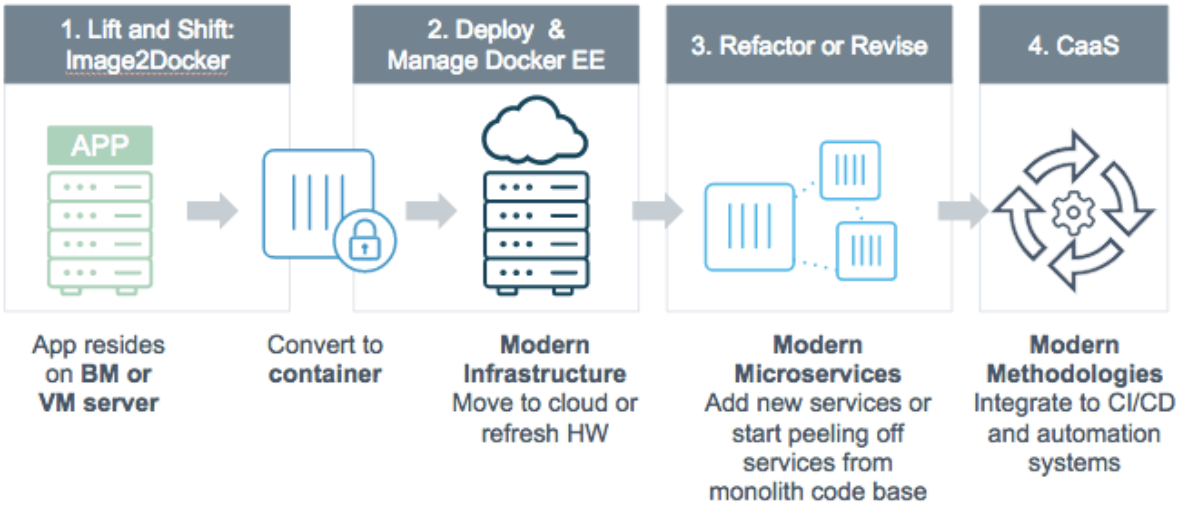

Containers = Operating system virtualization
OS
Applications
Kernel
Hardware
CONTAINER
CONTAINER
CONTAINER
Windows Server Containers
Maximum speed and density
CONTAINER
CONTAINER
CONTAINER
Kernel
Traditional virtual machines = hardware virtualization
Application
OS
Hardware
VM
VM
VM
Hyper-V Containers
Isolation plus performance
CONTAINER
CONTAINER
CONTAINER
Kernel
Kernel
Kernel
Hyper-V


Key differences between LXC and Docker
LXC
Host
VM1
Debian 64
PHP
Mysql
Nginx
Wordpress
VM2
Ubuntu 64
Ruby
Postgresql
Nginx
Wordpress
VM3
CentOS64
Nodejs
Redis
Nginx
Ghost
Filesystem neutral
Containers are like VMs with a fully functional OS
Data can be saved in a container or outside
Build loosely coupled or composite stacks
Docker
Host
Base Image
Ubuntu
VM1
PHP
VM1
Mysql
VM3
Nginx
VM4
Wordpress
Container
Storage
Volume
Base Image
Ubuntu
Layer 1
Apt-gen install
Nginx
Layer 2
Nginx
Config
Nginx
Layer 3
Nginx
Container
Base Image
Ubuntu
Layer 1
Apt-gen install
mysql
Layer 2
Mysql
config
Layer 3
Create
Mysql
user
Layer 4
Create
Mysql DB
VM2
Mysql
container
Loosely coupled single app containers
Layers to build app container
Containers are made up of read only layers via AUFS/Devicemapper
Containers are designed to support a single application
Instances are aphemeral, persistent data is stored in blind mounts to host or data volume containers


6,000,000,000
5,750,000,000
5,500,000,000
5,250,000,000
5,000,000,000
4,750,000,000
4,500,000,000
4,250,000,000
4,000,000,000
3,750,000,000
3,500,000,000
3,250,000,000
3,000,000,000
2,750,000,000
2,500,000,000
2,250,000,000
2,000,000,000
1,750,000,000
1,500,000,000
1,250,000,000
1,000,000,000
10,000,000
9,000,000
8,000,000
7,000,000
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
2016
6B
PULLS
2015
1B
PULLS
2014
1M
PULLS
docker hub
SwarmKit
HyperKit , VPNKit, DataKit
Docker 1.12 with built-in orchestration
Docker 1.11: OCI support
containerd
Notary
runC
Docker for Mac
Docker for Windows
libnetwork
Docker 1.8 : Docker Content Trust
Docker 1.7 : Multi-Host Networking
libcontainer
Docker 0.9 : Pluggable execution
2013 - 05
2013 - 06
2013 - 07
2013 - 08
2013 - 09
2013 - 10
2013 - 11
2013 - 12
2014 - 01
2014 - 02
2014 - 03
2014 - 04
2014 - 05
2014 - 06
2014 - 07
2014 - 08
2014 - 09
2014 - 10
2014 - 11
2014 - 12
2015 - 01
2015 - 02
2015 - 03
2015 - 04
2015 - 05
2015 - 06
2015 - 07
2015 - 08
2015 - 09
2015 - 10
2015 - 11
2015 - 12
2016 - 01
2016 - 09


Linux Containers
App
liblxc
namespaces
cgroups
SELinux/AppArmor
Linux kernel
Docker 1.10 and later
App
runC
containerd-shim
containerd
Docker Engine
namespaces
cgroups
SELinux/AppArmor
Linux kernel

How Docker Modernizes Legacy Apps



17

docker

Docker integration

Docker

An open source engine that automates the deployment of any application as a portable, self-sufficient container that can run almost anywhere

Partnership

Enable the Docker client to manage multi-container applications using both Linux and Windows containers, regardless of the hosting environment or cloud provider

Docker Hub

Huge collection of open and curated applications available for download. https://hub.docker.com

Collaboration

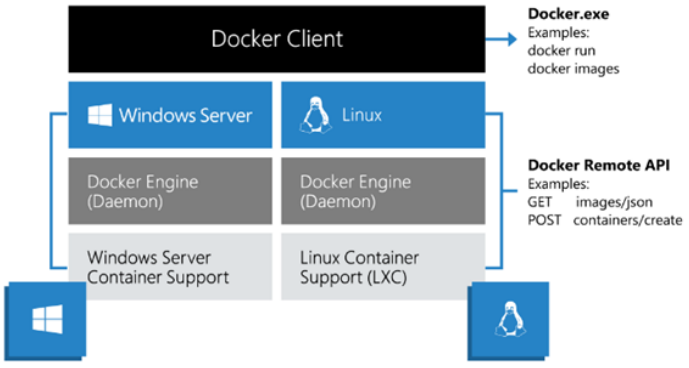
Bring Windows Server containers to the Docker ecosystem to expand the reach of both developer communities

Docker Engine

Docker Engine for Windows Server containers is developed under the Docker open source project

Docker Client

Windows customers can use the same standard Docker client and interface on multiple development environments



The history of containers

Container technology has come a long way from its chroots, starting with Google's exploration into cgroups and working up into widespread organizational adoption.

2004-07
Google used containers (cgroups) at scale

2004-2007

2008

2013

2014

2015

JANUARY 2008
cgroups merged into Linux kernel

AUGUST 2008
LXC released: Linux containers, based on cgroups, isolate resources without a VM

MARCH 2013
Docker hit the scene: Containerization platform based on Linux; integrated natively with other OSes: Windows, macOS

OCTOBER 2013
CoreOS Linux (now Container Linux by CoreOS) released

2013 RESEARCH
30 containerization vendors vie for users (451 RESEARCH)

JUNE 2014
Docker 1.0 is released

NOVEMBER 2014
Canonical launches LXD: Built on LXC with new features and functionality

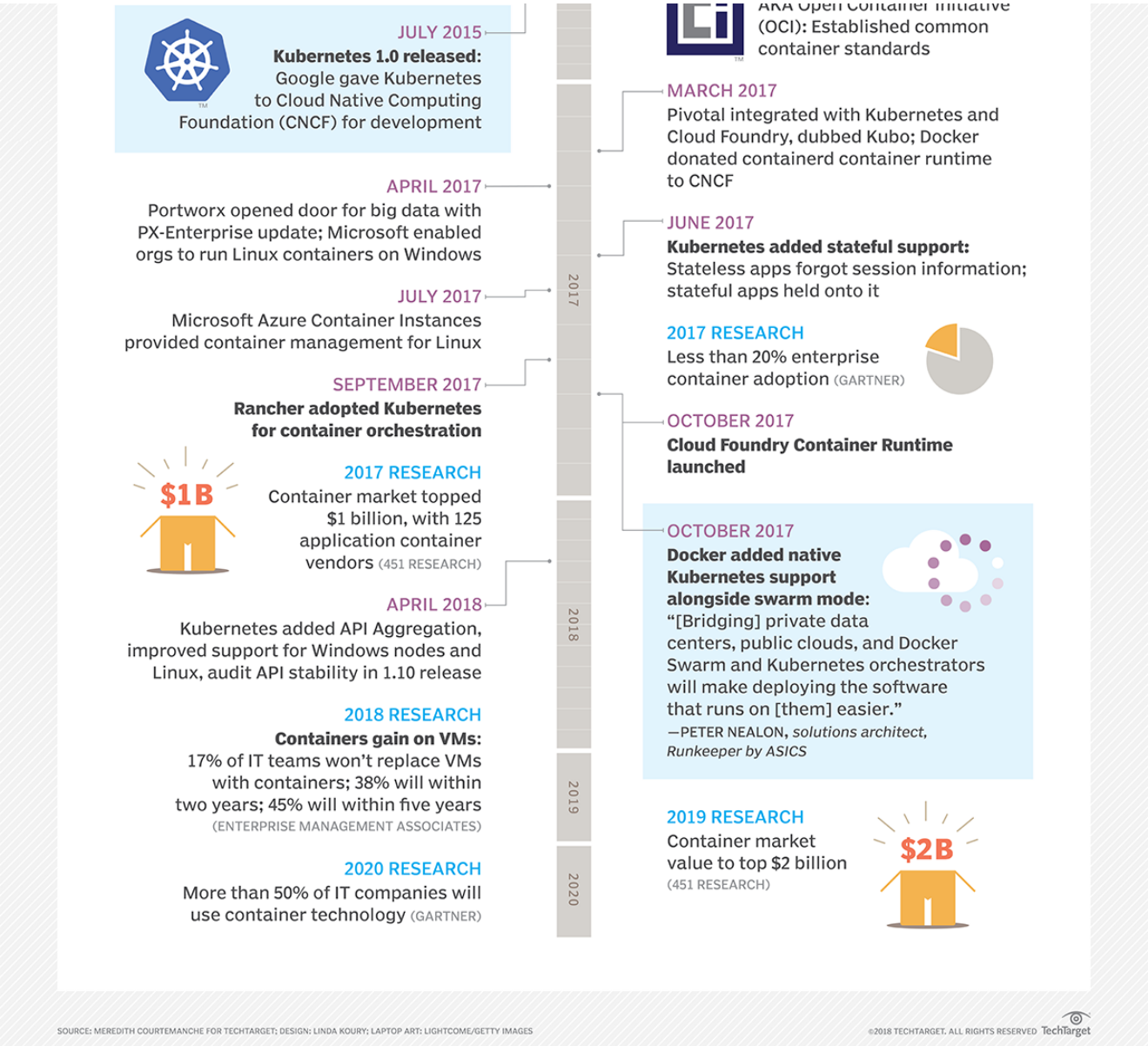
JUNE 2014
Kubernetes project announced: A now widely popular open source container orchestrator and scheduler created by Google

DECEMBER 2014
CoreOS introduced rkt containers as Docker alternative

APRIL 2015
CoreOS first to provide commercial Kubernetes offering

JUNE 2015
Open Container Project created AKA Open Container Initiative (OCI): Established common container standards

JULY 2015
Kubernetes 1.0 released:


JULY 2015
Kubernetes 1.0 released: Google gave Kubernetes to Cloud Native Computing Foundation (CNCF) for development
APRIL 2017
Portworx opened door for big data with PX-Enterprise update; Microsoft enabled orgs to run Linux containers on Windows
JULY 2017
Microsoft Azure Container Instances provided container management for Linux
SEPTEMBER 2017
Rancher adopted Kubernetes for container orchestration
$1B
2017 RESEARCH
Container market topped $1 billion, with 125 application container vendors (451 RESEARCH)
APRIL 2018
Kubernetes added API Aggregation, improved support for Windows nodes and Linux, audit API stability in 1.10 release
2018 RESEARCH
Containers gain on VMs: 17% of IT teams won't replace VMs with containers; 38% will within two years; 45% will within five years (ENTERPRISE MANAGEMENT ASSOCIATES)
2020 RESEARCH
More than 50% of IT companies will use container technology (GARTNER)
2017
2018
2019
2020
AKA Open Container Initiative (OCI): Established common container standards
MARCH 2017
Pivotal integrated with Kubernetes and Cloud Foundry, dubbed Kubo; Docker donated containerd container runtime to CNCF
JUNE 2017
Kubernetes added stateful support: Stateless apps forgot session information; stateful apps held onto it
2017 RESEARCH
Less than 20% enterprise container adoption (GARTNER)
OCTOBER 2017
Cloud Foundry Container Runtime launched
OCTOBER 2017
Docker added native Kubernetes support alongside swarm mode: "[Bridging] private data centers, public clouds, and Docker Swarm and Kubernetes orchestrators will make deploying the software that runs on [them] easier."
—PETER NEALON, solutions architect, Runkeeper by ASICS
2019 RESEARCH
Container market value to top $2 billion (451 RESEARCH)
$2B
SOURCE: MEREDITH COURTEMANCHE FOR TECHTARGET; DESIGN: LINDA KOURY; LAPTOP ART: LIGHTCOME/GETTY IMAGES
©2018 TECHTARGET. ALL RIGHTS RESERVED
TechTarget

Containers

A new approach to build, ship, deploy, and instantiate applications

Physical

- Applications traditionally built and deployed onto physical systems with 1:1 relationship
- New applications often required new physical systems for isolation of resources

Virtual

- Higher consolidation ratios and better utilization
- Faster app deployment than in a traditional, physical environment
- Apps deployed into VMs with high compatibility success
- Apps benefited from key VM features i.e., live migration, HA

Physical/virtual

Package and run apps within containers

Key benefits

- Further accelerate app deployment
- Streamline development and testing
- Reduces the "it works on my machine" problem
- Lower costs associated with app deployment
- Increase server consolidation

Performance of an application inside a container is generally the same or better when compared to the same application running within a VM.
Be sure to understand why you are adding containers to your enterprise or your existing virtualization environment could end up looking like these two VMs: