What is a Virtualized Container or Containerization

Developed By **Karen Baez**

WHAT IS A CONTAINER

To grasp what a container is, we must get familiar with keywords and their associated meanings..

Container

A virtual runtime environment that runs on top of a single operating system (OS) kernel and emulates an operating system rather than the underlying hardware. the OS is virtualized vs. the underlying hardware, which is what a Virtual Machine does. (2)Containers are a way to wrap up an application into its own isolated package. In its container, the application is not affected by applications or processes that exist outside of the container. Everything the application depends on to run successfully as a process is inside the container. Wherever the container might move, the requirements of the application will always be met, in terms of direct dependencies, because it is bundled with everything that it needs to run (library dependencies, runtimes, and so on).

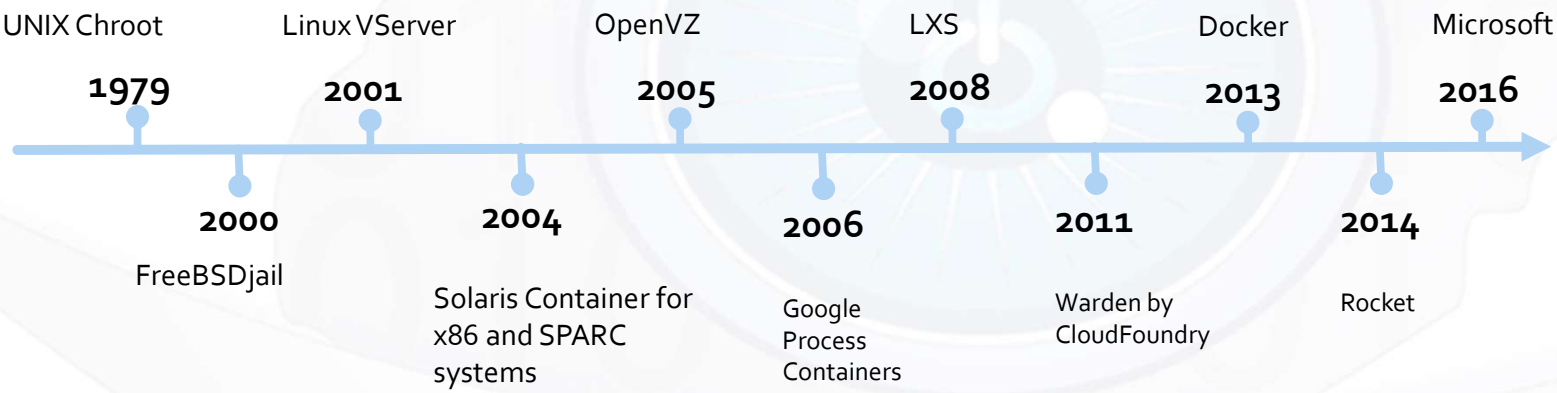
Container Engine

A managed environment for deploying containerized applications. The container engine allocates cores and memory to containers, enforces spatial isolation and security, and provides scalability by enabling the addition of containers.

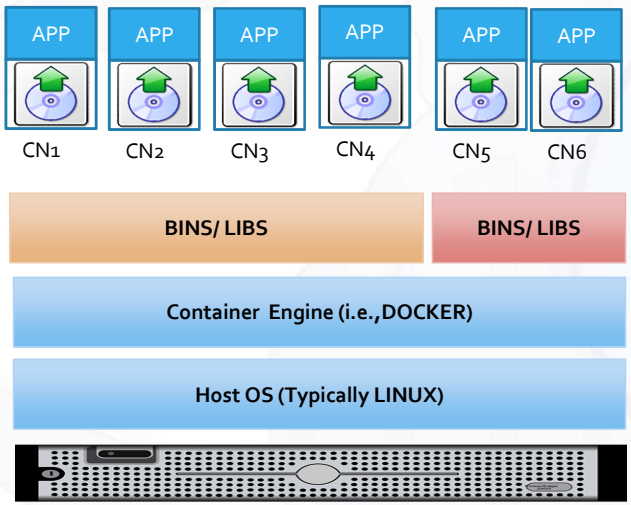
Hybrid Container Architecture

An architecture combining virtualization by both virtual machines and containers, i.e., the container engine and associated containers execute on top of a virtual machine. Use of a hybrid container architecture is also known as hybrid containerization.

WHEN DID IT BEGAN



WHAT IS THE TYPICAL ARCHITECTURE



Containers are a way to wrap up an application into its own isolated package. In its container, the application is not affected by applications or processes that exist outside of the container. Everything the application depends on to run successfully as a process is inside the container. Wherever the container might move, the requirements of the application will always be met, in terms of direct dependencies, because it is bundled with everything that it needs to run (library dependencies, runtimes, and so on).

Containers sit on top of a physical server and its host OS—for example, Linux or Windows. Each container shares the host OS kernel and, usually, the binaries and libraries, too. Shared components are read-only. Containers are thus exceptionally "light"—they are only megabytes in size and take just seconds to start, versus gigabytes and minutes for a VM.

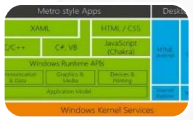
They reduce management overhead and because they share a common operating system, only a single operating system needs care and feeding for bug fixes, patches, and so on. This concept is similar to what we experience with hypervisor hosts: fewer management points but slightly higher fault domain. In short, containers are lighter weight and more portable than VMs. They provide a way to virtualize an OS so that multiple workloads can run on a single OS instance

WHO IS IN CONTROL OF SECURITY

Managed by Provider (s)

Consumer Interface

Container Engine



Runtime

Middleware

O/S

Networking

Servers

Storage

Managed by Consumer

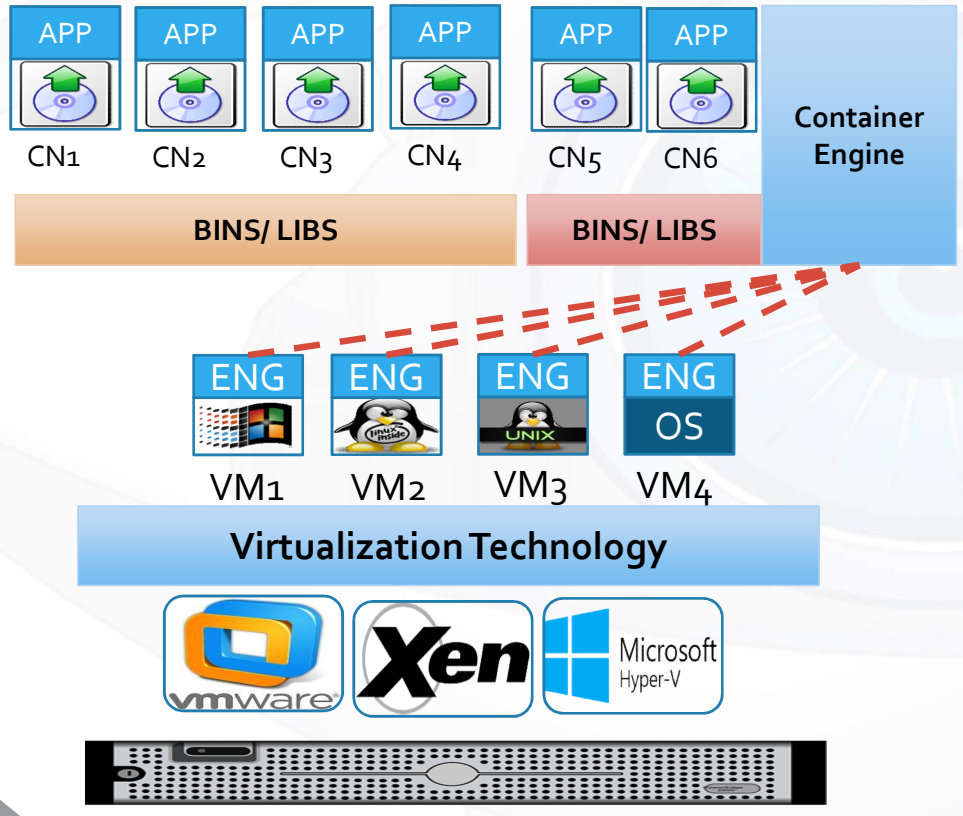
App Container

Data

HOW DOES IT DIFFERS FROM A VM

Virtual Machine (VM)	Container
Heavyweight	Lightweight
Limited Performance	Native Performance
Runs is own OS	Share same OS
Starts in Minutes	Starts in Milliseconds
Allocates Memory	Requires less Memory
Isolated and more secured	Process level isolation Less secured

WHAT IS A HYBRID ARCHITECTURE



A hybrid architecture means that a container sits on top of a virtual server within a virtualized environment.

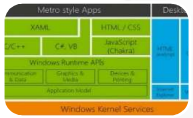
WHO IS IN CONTROL OF SECURITY

Managed by Provider (s)

Consumer Interface

Container Engine

CaaS



Runtime

Middleware

O/S

PaaS

Managed by Consumer

App Container

Data

Networking

Servers

Storage

Hypervisor

IaaS

WHAT ARE THE PROS

- Reduced IT Resources
- Fast App Startup
- Platform Independent
- Improved Development
- Spatial Isolation
- Operational Simplicity
- Portability
- Easier Migration
- Scalability
- Performance
- Cost Effective

WHAT ARE THE PROS

Line Item	Description
Reduced IT Resources	Containers are a very cost effective solution. They can potentially help decrease the operating cost (less servers, less staff) and the development cost (develop for one consistent runtime environment). **V**irtualization via containers decreases hardware costs by enabling consolidation (i.e., the allocation of multiple applications to the same hardware improves hardware utilization). It enables concurrent software to take advantage of the true concurrency provided by a multicore hardware architecture. In addition, it enables system architects to replace several lightly-loaded machines with fewer, more heavily-loaded machines to minimize SWAP-C (size, weight, power, and cooling), free up hardware for new functionality, support load balancing, and support cloud computing, server farms, and mobile computing.
Fast Startup	The average container size is within the range of tens of MB while VMs can take up several gigabytes. Therefore a server can host significantly more containers than virtual machines. Running containers is less resource intensive then running VMs so you can add more computing workload onto the same server. Provisioning containers only take a few seconds or less, therefore, the data center can react quickly to a spike in user activity.
Platform Independent	Containers can enable you to easily allocate resources to processes and to run your application in various environments.
Improved Development	Using containers can decrease the time needed for development, testing, and deployment of applications and services. Testing and bug tracking also become less complicated since you there is no difference between running your application locally, on a test server, or in production. Containers support Agile and continuous development processes by enabling the integration of increments of container-hosted functionality., and that makes them a great option for micro-services, DevOps, and continuous deployment.

WHAT ARE THE PROS

Line Item	Description
Spatial Isolation	Containers support lightweight spatial isolation by providing each container with its own resources (e.g., core processing unit, memory, and network access) and container-specific namespaces
Operational Simplicity	Running containers is less resource intensive then running VMs so you can add more computing workload onto the same server. Compared with virtual machines, containers are lightweight with regard to storage size. The applications within containers share both binaries and libraries.
Portability	**C**ontainers support portability from development to production environments, especially for cloud-based applications.
Easier Migration	Less code to transfer, migrate, upload workloads.
Scalability	A single container engine can efficiently manage large numbers of containers, enabling additional containers to be created as needed.
Performance	containers increase performance (throughput) because they do not emulate the underlying hardware. Note that this advantage is lost if containers are hosted on virtual machines (i.e., when using a hybrid virtualization architecture).

WHAT ARE THE CONS

- Security
- Flexibility
- Networking
- Shared Resources
- Isolation

WHAT ARE THE CONS

Line Item	Description
Security	Containers share the kernel, other components of the host operating system, and they have root access. This means that containers are less isolated from each other than virtual machines, and if there is a vulnerability in the kernel it can jeopardize the security of the other containers as well. In addition, they are not by secured by default and require significant work to make them secure. One key issue is ensuring that the Image in itself comes from a trusted source and any open source firmware/software within it is fully vetted and supported. As with safety, traditional security accreditation and certification processes have not being fully molded to take virtualization into account, making security (re)accreditation and (re)certification difficult. The key is understanding that assessments performed have to be molded by the technology, something most organizations ignore. And molding means incorporating technology specific controls and test that are tied to such technology.
Flexibility	You need to start a new server to be able to run containers with different operating systems. For Enterprise Systems this can be a high constraint.
Networking	deploying containers in a sufficiently isolated way while maintaining an adequate network connection can be tricky. While there's a few solutions that can help alleviate the issue, they have limits. In addition, maintenance of multiple containers can impact maintenance and configuration management.

WHAT ARE THE CONS

Line Item	Description
Shared Resources	Applications within containers share many resources to include (1) container-specific resources, container engine, and OS kernel, (2) for Hybrid Architectures, virtualization by VM resources including the virtual machine, the hypervisor, and the host operating system if using a type-2 hypervisor, and (3) multicore processing resources including (a) *processor-internal* resources (L3 cache, system bus, memory controller, I/O controllers, and interconnects) and (b) *processor-external* resources (main memory, I/O devices, and networks). Such shared resources imply (1) single points of failure exist, (2) two applications running in the *samecontainer* can interfere with each other, and (3) software running in *onecontainer* can impact software running in *anothercontainer* (i.e., interference can violate spatial and temporal isolation).
Isolation	Although containers provide significant support for isolation, the use of containers does not guarantee isolation. Temporal interference may cause delays that cause hard, real-time deadlines to be missed. Spatial interference can cause memory clashes. Whether by VM or containers

Risk

Overview of key items

ARE THERE ARCHITECTURAL RISKS

Typical Infrastructure	Yes! No	Hybrid Infrastructure
Account Management		Incident Response
Access Control		Contingency Planning
Hosting Environment		Integrity
Underlying Security		

TYPICAL INFRASTRUCTURE RISKS

Typical Infrastructure

Hybrid Infrastructure

Risk	Description
Account Management	The underlying layer within the container can pose a high risk if the organization offering the services doesn't have a solid account management process in place for the underlying OS and the Container Engine.
Access Control	Most containers required a high level privilege account, in Linux they typically use ROOT. This is an issue because if the underlying OS is compromise, then every single container is compromised as well. And while this apply to both, typical and hybrid infrastructure, it would be more prevalent on a typical infrastructure. Why? Because a Hybrid Infrastructure is less likely to be impacted, since the VM's
Hosting Environment	The hosting environment encompasses the physical environment and the underlying perimeter infrastructure. It's important to grasp how the underlying parties responsible for those layers secure their environment and whether or not they are hosted internally within the CSP's network and/or at a datacenter that's offsite. If a datacenter, is important to determine if such complies with specific security standards such as SAE16/18 (SOC2)
Underlying Security	In a typical architecture, you need to determine how the OS is being hardened and how often it is backed up. In a hybrid architecture, you need to determine if there's multiple actors, you need to identify how the cloud service levels are hardened, and how key security concerns related to the underlying infrastructure are implemented. For more on each cloud service, review previous videos.

HYBRID INFRASTRUCTURE RISKS

Typical Infrastructure

Hybrid Infrastructure

Risk	Description
CaaS	The underlying layer within the container can pose a high risk if the organization offering the services doesn't have a solid account management process in place for the underlying OS and the Container Engine.
Incident Response	When outsourced from a 3rd party, its essential to understand how Incident response will be handled on the underlying layers, especially if using a Hybrid Implementation, as there could be multiple ecosystem actors involved. Those lack of a solid incident response plan from the service provider can be catastrophic and impact forensic efforts.
Contingency Planning	Understanding how to address a contingency when there's issues with the applications is key. This includes a timeframe if the issues arise from the underlying infrastructure. Qs. From a Typical Architecture- how often are the physical server images made? How are they stored? How are they tested to ensure they work as intended? Qs. From a Hybrid Architecture- Are all VMs replicated? If yes, are they limited to a specific geographical location? If multiple cloud actors, how does each actor handle contingency planning an how will it impact the VMs? Again is important to become familiar with the 3 Cloud Service Levels (IaaS, PaaS, SaaS) to grasp the security considerations that must be in place to fully understand the container's security status.

ARE THERE DATA RISKS

Typical Infrastructure	Yes! No	Hybrid Infrastructure

Data Protection

Regardless of architecture option (Typical or Hybrid), how data is protected during transmission, processing, and storage is key. Ideally, all sensitive data is maintained outside of the container on a separate, secured environment. But in the end, determining the mechanisms apply is key. Are they using SOAP? If yes, what version. Are they using APIs? If yes, how are they secured? Is data encrypted in transit? If yes, what encryption algorithm is used?

Data Sanitization

for security purposes, data within a container should be restricted to processing and transmission, not storing. But IF data is kept, then is important to grasp how it will be sanitized and confirmed.

Incident Response

Data breaches occur often, most of them initiated via the Application Interface due to vulnerabilities within the Web Servers and/or underlying infrastructure, in this case, the Operating System. Knowing the architecture type (typical vs hybrid) will be key, but especially IF this service is provided by a 3rd Party in order to understand how Incident Response will be handled and who handless it, if multiple actors.

ARE THERE WEB SERVER RISKS

Typical Infrastructure

Hybrid Infrastructure

Apache

IIS

Most security attacks via the Application Interface are due to un-configured or misconfigured Web Servers (i.e., Apache or IIS). Because a container is meant to support multiple applications, and each application might require unique settings or package versions, most developers prefer to incorporate the web server components within the virtual container in order to avoid conflicts between the applications in order to perform updates to its associated libraries, which tend to be unique per application. Therefore it is of essence that Web Servers are secured.

Security Considerations

Overview of key items

SECURITY CONSIDERATIONS

- Data
- Content Image
- Content Management
- Hosting Environment
- Access Control
- Security Integration
- Infrastructure
- Access Points

SECURITY CONSIDERATIONS

Risk	Description
Data	Avoid storing data within the containers. If outsourcing, data protection requires that you have a full understanding on the underlying architecture and the service's provider status as it pertains to Enterprise Risk Management.
Content Image	The base image is the most important component when it pertains to security, because it is used as the starting point from which you create derivative images. Ensuring that the images are extracted from a Trusted organization is key. Even when using trusted images, though, adding applications and making configuration changes will Introduce new variables and risks. Anything introduced must be vetted for security.
Content Management	When bringing in external content to build your apps, you need to have proactive content management in mind. Qs: Are the container images signed and from trusted sources?, Are the runtime and operating system layers up to date?, How quickly and how often will the container be updated?, How quickly and how often will the container be updated?
Hosting Environment	Is it hosted within the CSP's Network or a 3rd Party Service Provider? How many Cloud Actors are part of the ecosystem where the systems are hosted? What protections are applied within the hosting environment? Do they have an Enterprise Risk Management Framework in place?

SECURITY CONSIDERATIONS

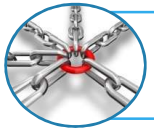
Risk	Description
Access Control	to manage both access to, and promotion of, all container images used by the organization. That means protecting the images you download as well as the ones you build. Using a private registry will allow you to control access through role-based assignments while also helping you manage content by assigning metadata to the container. Metadata will provide information like identifying and tracking known vulnerabilities. A private registry also gives you the power to automate and assign policies for the container images you have stored, minimizing human errors that may introduce vulnerabilities into your container. Q: What role-based access controls can you use to manage container images?, Are there tagging abilities, to help sort images? Can you tag images as approved only for development, and then testing, and then production?, Are there tagging abilities, to help sort images? Can you tag images as approved only for development, and then testing, and then production?, Are there tagging abilities, to help sort images? Can you tag images as approved only for development, and then testing, and then production?, Can you use the registry to assign and automate policy (e.g. checking signatures, code scans, etc.)?
Security Integration	Do you have automate policies to flag builds with security issues, especially as new security vulnerabilities are found. Because patching containers is never as good of a solution as rebuilding them, integrating security testing should take into account policies that trigger automated rebuilds. Running on component analysis tools that can track and flag issues is the first part of this step. The second part is establishing tooling for automated, policy-based deployment. Q: How can you prevent patching running containers, and instead use triggers to rebuild and replace containers with automated updates?

SECURITY CONSIDERATIONS

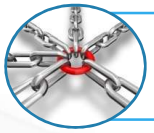
Risk	Description
Infrastructure	Another layer of container security is the isolation provided by the host operating system (OS). You need a host OS that provides maximum container isolation. This is a big part of what it means to defend your container deployments environment. The host OS is enabled using a container runtime, ideally managed through an orchestration system. To make your container platform resilient, use network namespaces to sequester applications and environments, and attach storage via secure mounts. An API management solution should include authentication and authorization, LDAP integration, end-point access controls, and rate limiting. Q: Which containers need to access one another? How will they discover each other?, How will you control access and management of shared resources (e.g. network and storage)?, How will you manage host updates? Will all of your containers require updates at the same time?, How will you monitor container health?, How will you automatically scale application capacity to meet demand?
Access Points	New vulnerabilities and exploits generated from the creativity of hackers behind the Apache Struts, the Linux stack clash, and the dirty cow exploits – all made infamous by major data breaches and ransomware attacks – denote that even Linux systems aren't safe. With this in mind, Access points within a container should incorporate firewall level protection.

FIREWALL INTEGRATION

The choices available for container firewall deployments are as varied as those for traditional firewalls and include:

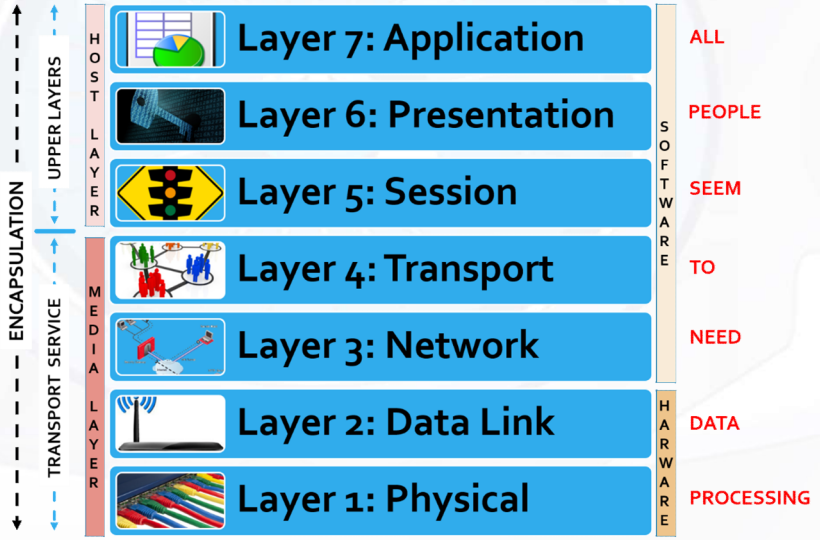


OSI Layer 3 & 4 Filtering



Application Level Attack Detection

OSI Layer 7 firewall



ALTERNATE INFRASTRUCTURE

Converge Infrastructure		Hyper-converge Infrastructure
Relies on hardware	Yes! No	software-defined
employs building blocks		Flexible, maneuverable And Scalable
Storage Physical Server		Storage Service on each node

ALTERNATE INFRASTRUCTURE

Converge Infrastructure  **Hyper-converge Infrastructure**

Risk	Description
Converged Infrastructure (CI)	brings the four core aspects of a data center -- compute, storage, networking and server virtualization -- into a single chassis. The hardware-focused, building-block approach of VCE (a joint venture of EMC, Cisco, and VMware).
Hyper-converged Infrastructure (HCI)	adds tighter integration between more components through software. The software defined approach of Nutanix, VMware, and others called hyper-converged infrastructure
Storage	In a converged architecture, the storage is attached directly to the physical servers. Flash storage generally is used for high-performance applications and for caching storage from the attached disk-based storage. It shares storage to all compute and virtual machines (VMs) . It incorporates a centralized array accessible using a traditional storage network (FC with FSPF or ISCSI/NFS) In a hyper converged architecture, it has the storage controller function running as a service on each node in the cluster to improve scalability and resilience. In this architecture, storage to all compute and virtual machines (VMs) is NOT shared, instead it incorporates distributed drives in each server forming a centralized file system.

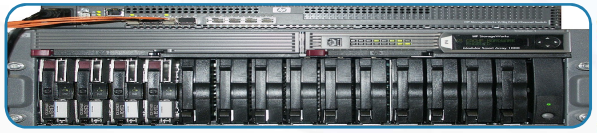
CONVERGE VS HYPER-CONVERGE

Converge Infrastructure

Virtualization Technology

vmware | Xen | Microsoft Hyper-V

Server



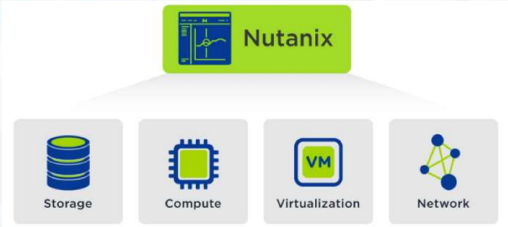
Storage Network

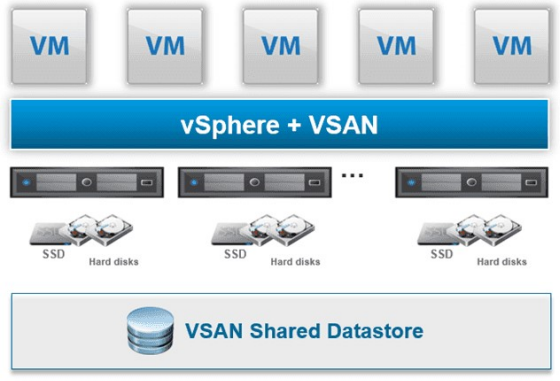
Storage Area Network (SAN)

Storage System

Network-attached Storage (NAS)

Hyper-Converged Infrastructure





Key Controls

Overview of key items

WHAT CONTROLS ARE KEY

- IA- Identification and Authentication
- AC- Access Control
- AU- Audit and Accountability
- CM- Configuration Management
- CP- Contingency Planning
- IR- Incident Response
- MP- Media Protection
- PE- Physical and Environmental Protection
- SA- Systems and Services Acquisition
- SC- Systems and Communications Protection

CaaS Samples

Send them to
info@cyberadeptness.com

Need Help?

We have over 20+ years of combined experience in the field and a unique process to streamline the requirements.

Contact us today to schedule a meeting and determine how we may be of help to your organization. Our processes are flexible to accommodate compliance needs, regardless of sector (i.e., Healthcare, Finance, Law, Education, etc.).

References

- Carnegie – Virtualization via Containers - https://insights.sei.cmu.edu/sei_blog/2017/09/virtualization-via-containers.html
- Container Virtualization: What makes it work so well? - https://cloudacademy.com/blog/container-virtualization/
- RedHat Container Security- https://www.redhat.com/en/topics/security/container-security

maneuverable

- A guide to securing docker and kurbonet containers with a firewall- https://www.sdxcentral.com/articles/contributed/a-guide-to-securing-docker-and-kubernetes-containers-with-a-firewall/2018/03/