https://medium.com/@chrisrust/how-application-defined-infrastructure-adi-will-disrupt-the-cloud-in-2018-cf28f989ffba

How **Application-Defined Infrastructure (ADI)** Will Disrupt the Cloud In 2018

Chris Rust

Follow

Jan 4, 2018 · 19 min read

INTRODUCTION

The world is in a golden age of information technology (IT) innovation. The mega-forces of cloud, mobile, big data analytics powered by machine learning, IoT, and massively scalable apps are re-shaping all aspects of business and society. At the center of this IT renaissance is an unprecedented global data center (DC) buildout in public, private, and hybrid cloud computing. According to Synergy Research Group, the number of hyperscale DCs around the globe grew from 300 at YE16 to 390 at YE17, with another 69 hyperscale DCs that are in various stages of planning or building.

In this paper, we briefly review the three major waves of DC infrastructure innovation to date. We then introduce the fourth wave of IT infrastructure: Application-Defined Infrastructure (ADI), and the technology forces and operational challenges driving its adoption by large enterprises.

A BRIEF HISTORY OF DC INFRASTRUCTURE

A DC is a purpose-built structure used to house computer systems and associated components, such as networking equipment, storage systems, and telecommunications equipment. It is the brains of the knowledge economy and to our connected world. The modern DC has its’ roots in the main-frame room of the 1960s, the telecommunications central office, and the enterprise

IT wiring closet. The past two decades have seen an explosion of creativity perfecting the art of the modern DC.

1997–2007, First Wave — Bare Metal Servers

Bare metal servers are single-tenant physical servers. Their strengths are high application (app) performance and predictability. Their weaknesses are high cost, medium complexity to provision apps, and low flexibility once apps are deployed. They continue as a solution of choice for specific performance sensitive workloads that merit dedicated infrastructure (i.e., databases). Bare metal is still often used for dedicated computer clusters that are built to support specific scale-out distributed computing apps (e.g., a Hadoop cluster). The requirements for greater flexibility and improved economics have made this approach limiting given the continually evolving app landscape.

2005-Present, Second Wave — Virtualization with Hypervisors

Virtualization is an emulation of a computer system that enables one physical computer to run one or more virtual machines (VMs)(see Figure 1).

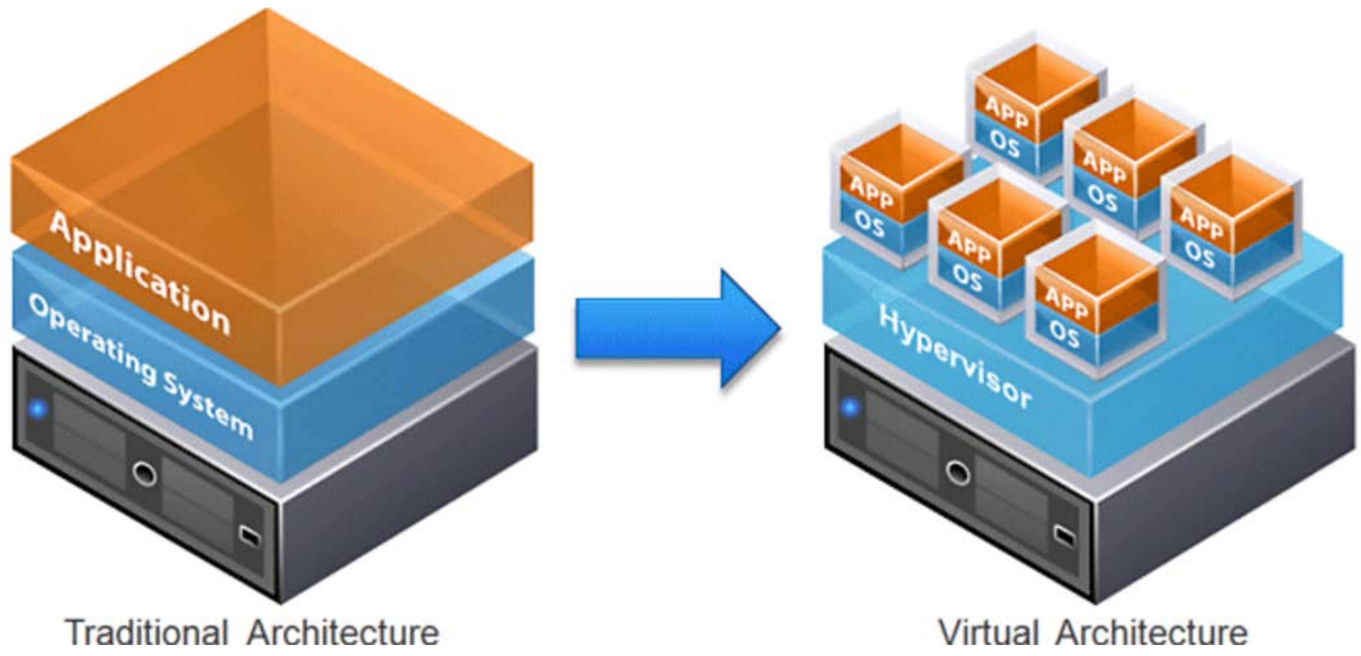

Figure 1: Making one computer look like many with hypervisor-based virtualization

While this concept goes back to the 1960s and the era of mainframe computers, it was brought to the forefront of IT efficiency gains by VMware in 1998 with their commercialization of the modern hypervisor. Before VMware, a significant amount of expensive computer resources were underutilized. The VMware hypervisor helped solve the need for greater IT efficiency by making one computer look like multiple computers, each one with their guest operating system. Released in 1999, the VMWare hypervisor was novel because it enabled virtualization on Intel x86 for the first time by using binary translation to replace privileged instructions to trap into the hypervisor. As of November 2017, VMware had grown into a highly profitable $53B market cap company with $1.98B of 3Q17 revenue and $443M of 3Q17 net income. With hundreds of thousands of businesses around the world running important portions of their operations on VMware's virtualized systems, VMware is the market share leader in hypervisor-based virtualization solution for enterprise private clouds. Other hypervisors include Microsoft Hyper-V, Linux KVM, and Xen.

The strengths of virtualization with hypervisors include: technology maturity; broad adoption; improved computer utilization by enabling multiple VMs; infrastructure software to build and operate clouds based on VMs; strong multi-tenancy support.

The weaknesses of virtualization with hypervisors include: high complexity; hypervisor resource overhead; client operating system resource overhead for each guest VM; non-negligible application performance hit when compared to bare metal infrastructure; the "noisy neighbor effect" when one user impacts the performance and stability of other users within the same physical server; the "IO blender effect" when multiple VMs send their IO requests at the same time and degrade storage performance; extremely high hypervisor energy consumption overhead depending on workload; and the time required to instantiate new VMs (each of which requires a full client OS).

The hypervisor vendors and cloud operators who have built their platforms on the hypervisor paradigm are working to address some of these weaknesses with techniques that include: hardware virtualization (e.g., Amazon Nitro Project); lightweight hypervisors; unikernels that package the OS and application into one bundle and eliminates the traditional partition between OS kernel and user space; and various approaches to serverless computing.

2010-Present, Third Wave — Hyper-Converged Infrastructure (HCI) (also based on hypervisors)

A Hyper-Converged Infrastructure (HCI) is a wholly software-defined IT infrastructure that virtualizes all of the elements of conventional "hardware-defined" systems (see Figure 2). HCI includes, at a minimum, virtualized computing (a hypervisor), a virtualized software-defined storage (SAN) and virtualized networking (Software-defined networking).

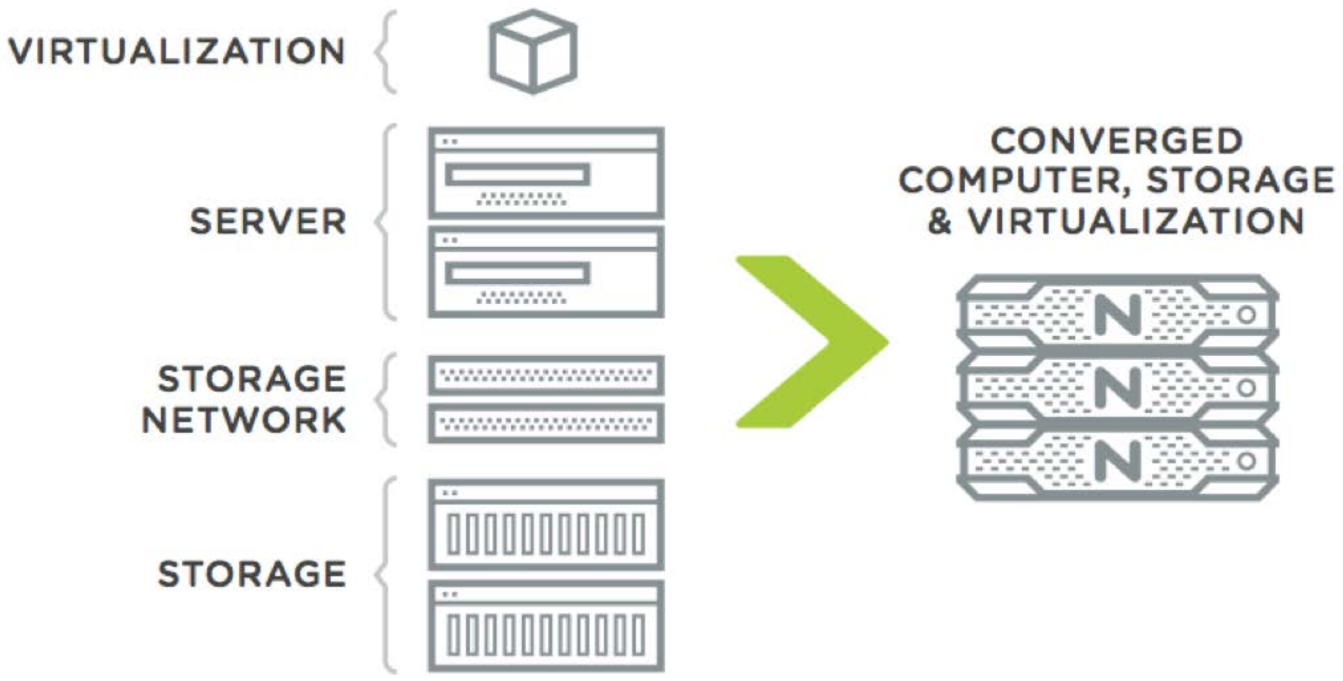

Figure 2: Hyper-converged infrastructure (Source: Nutanix Definitive Guide for HCI)

Simply stated, HCI integrates compute, storage, and network connectivity into a “cloud-in-a-box,” then provides a unified management view of both hardware and software assets to hide the complexity of the cloud. HCI uses sophisticated infrastructure software on top of bare metal commodity parts to simplify management and increase ease-of-use for end users in certain high-value apps (e.g., virtual desktop). HCI vendors include Dell/EMC, IBM, Lenovo, HP, Nutanix, Stratoscale, and Cisco.

The strengths of HCI include: ease of use by pre-packaging hardware and software together and hiding the underlying complexity of virtualization with the hypervisor, graceful scaling of infrastructure by growing clusters of HCI appliances and, simplification of the do-it-yourself approach to building a private cloud. HCI offers a strong solution for high-value applications such as desktop virtualization.

The weaknesses of HCI include: the ratios between compute and storage are locked in at the time these system resources are packaged together into an appliance; lack of support for some important classes of stateful apps (e.g. relational databases); limited support for massively scalable modern data stack distributed computing apps like Hadoop, MongoDB, Cassandra, and Spark; HCI vendor lock-in; and little to no deployments in larger clouds.

THE 4TH WAVE OF IT INFRASTRUCTURE

In the first three waves of IT transformation every IT project started with planning out the underlying Infrastructure first. For example, to deploy a database, planning first started with procuring and configuring severs or VMs, networks and storage. The chosen infrastructure components had to be planned to ensure they meet the application's current SLAs and anticipated growth. Only after all this infrastructure was planned and configured were apps brought online. But infrastructure exists to serve apps, not the other way around. Wouldn't it be better to start an IT project at the application — by describing just its needs and letting the infrastructure self-assemble and configure itself to meet those needs (current and anticipated)?

We are entering the 4th wave of IT infrastructure innovation where apps will define the infrastructure that serves them. In this 4th wave both apps and people are liberated from the shackles of the specifics of the underlying IT infrastructure. The underlying infrastructure itself might change, from bare metal to VMs to private or public cloud, but the interaction with an application remains unchanged. This 4th wave is an era of Application Defined Infrastructure (ADI), where infrastructure becomes increasingly invisible, and simplicity is once again the ultimate sophistication. The drivers for this 4th wave of IT infrastructure are given below, starting with a discussion of a significant secular trend — containers.

Containers

Containers is a technology that packages an application and its dependencies in a manner that allows it to be reliably moved from one computing environment to another (see Figure 3).

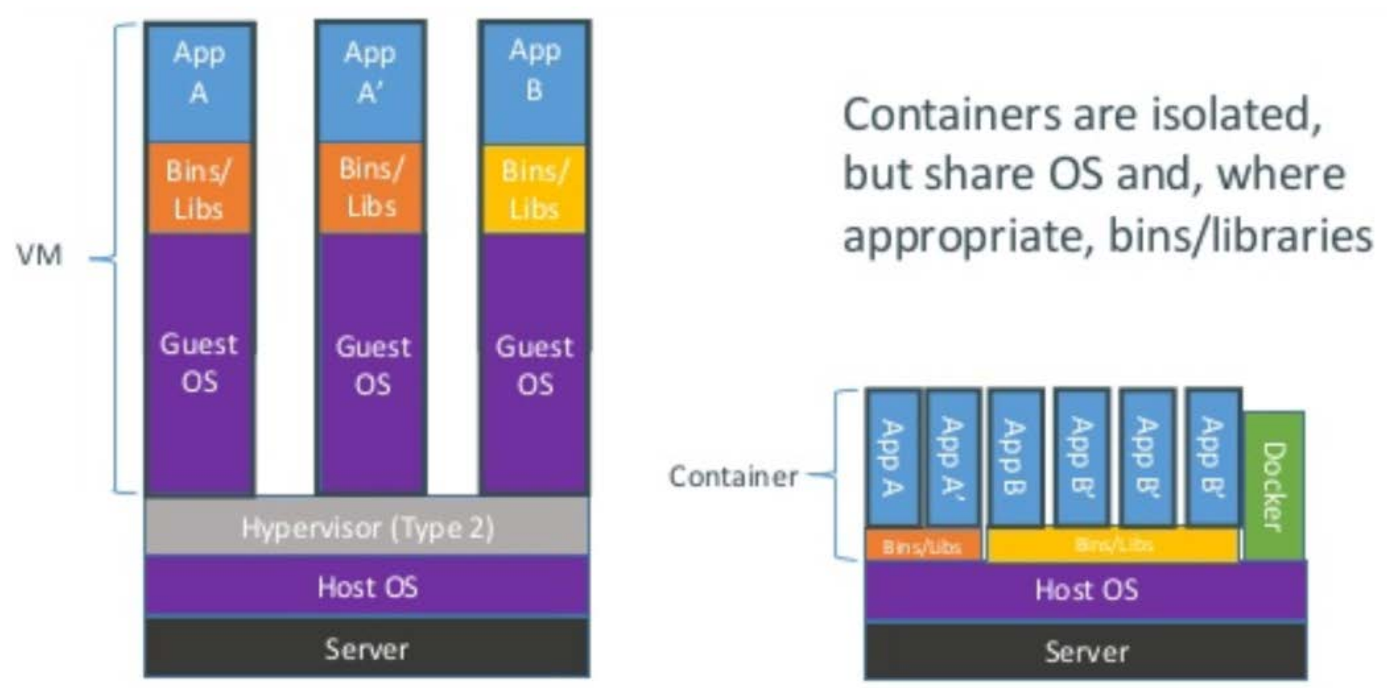

Figure 3: Containers vs. Virtualization with Hypervisor (Source: Docker)

Unlike VMs, which package an entire operating system along with the application, multiple containers running on a machine share the operating system. Each application running inside its own container continues to enjoy an isolation boundary that makes it appear like it is the only one running on that machine.

Containers became widely popular with Docker's introduction of application containers in 2013. Docker is playing a leadership role in popularizing the concept of containers by making application packaging in the form of Docker images an industry standard.

Containers, specifically Docker, significantly simplify how apps are configured. They strike the right balance between merging or, (where applicable) separating configuration from the application payload (the container image). Containers operate at close to zero overhead because containers do not virtualize in the hypervisor sense of the word. Instead they isolate apps (or portions of apps) from one another in secure partitions that run in a shared user space over one host operating system. This means apps run at bare metal speeds without consuming any additional resources.

However, containers by themselves are not sufficient as an IT infrastructure management paradigm because they are not "infrastructure aware." Organizations are discovering that existing data center infrastructure is not capable of dealing with large numbers of containerized apps since a single modern microservices-based web application can easily span hundreds or more containers. Organizations run many apps and often find their systems administration teams overwhelmed attempting to match resources with containers.

Containers improve server utilization by allowing multiple apps to run on the same server. But since all apps share the same storage,

storage performance can be erratic, which impacts overall application performance. To combat this, some organizations deploy critical apps on siloed infrastructure to ensure good performance, which leads to overprovisioned hardware and poor resource utilization. Cloud computing is evolving to address this. Cloud service providers have long offered Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS). The first wave of Containers-as-a-Service (CaaS) on bare metal is beginning to be developed by the largest Cloud service providers. The concept of “application state” is important to understand when discussing a CaaS offering.

Stateless and Stateful Apps

Understanding the concept of “app state” helps to understand the evolving requirements of the IT infrastructure that serves apps. App state is the data that application components need to perform their intended function. Apps may require configuration information, user credentials, user profile information, user history, clickstream data. Data associated with apps can be stored in many different physical locations: local server cache; in a file system; in a database table; or in a storage resource. There are many elements that contribute to a full understanding of app state: app persistence requirements (i.e. uptime, re-start requirement, data loss windows); configuration state; session state; infrastructure state (e.g. networking addresses, cluster state).

Stateless apps do not save client data generated in one session for use in the next session with that client. Each session is carried out as if it was the first time and responses are not dependent upon data from a previous session. Protocols like HTTP are stateless because the web server does not remember any state across page requests that is processes.

In contrast, Stateful apps store data from previous sessions, limiting the data that needs to be stored on the client end and retaining information on the server from one use to the next. Applications that needs to perform real-time work typically maintains some state locality to get very fast response times. Examples include content delivery networks, streaming media servers, identity management and authentication servers, and core transaction systems for payment processing. Many of the most important mission-critical applications often need to preserve and manage state. Complex, distributed Big Data, NoSQL and Database apps are stateful and need to run both on premises and in the cloud. Simply attaching a storage volume to Docker is not sufficient to support stateful apps because that doesn’t address performance predictability, app portability and high availability, lifecycle management etc. Thus, there is a pressing need for cloud computing infrastructure to do a much better job of supporting mission-critical stateful apps.

Container Orchestration

Putting applications into containers becomes interesting only when those containers can be efficiently deployed, managed, and scaled efficiently. Container orchestration engines perform the important function of managing clusters of containers, and is an important building block of the 4th wave of IT infrastructure — important enough to inspire a “container orchestration war.” Sample players include — Docker’s Swarm, Redhat’s OpenShift, Rancher’s Cattle, Mesosphere’s Marathon, AWS’ ECS, CoreOS’ Fleet. Container orchestration is strategic, but it is a component of a broader infrastructure management solution. The focus of these container orchestration engines is mainly on stateless cloud-native apps. However, there are efforts such as Kubernetes StatefulSet that hope to improve the ability to support open source databases like MySQL and PostgreSQL in containerized environments, but full infrastructure control is required to meet the service level agreements and quality-of-service and high availability requirements of demanding stateful apps.

THE 4TH WAVE — APPLICATION-DEFINED INFRASTRUCTURE (ADI)

ADI Requirements Summary

With the rise of containers, and the need to gracefully run both stateless and stateful applications on a shared multi-tenant infrastructure, the Application-Defined Infrastructure (ADI) is now required.

The ADI can be described as a container-based, application-aware computes and storage platform. The software efficiently abstracts underlying server, VM, network, and storage boundaries to produce a compute, storage, and data continuum. Many different containerized apps can run in this continuum without impacting one another’s performance.

Application portability and scalability are increased because compute and storage are decoupled; apps can be freely moved around the continuum without moving or copying data. Complex distributed apps like NoSQL, Hadoop, Cassandra, and Mongo can be deployed quickly and easily.

The ADI enables the intelligent provisioning of containers and storage based on individual application requirements as well as the topology of the environment, and it configures the application to make the best use of those components. The ADI ensures that all apps get sufficient compute, storage, and network resources to meet user-defined quality of service requirements; the result is predictable performance for all apps.

The ADI should provide the ability to automatically recover failed nodes and disks, and seamlessly move workloads between servers. As a result, hardware can be used more efficiently, and less hardware is required in reserve for inevitable performance spikes.