OPTIMIZING TELCO CONTAINERIZATION STRATEGIES

Miguel Castaneda, Industry Analyst
Jake Saunders, Vice President, Asia-Pacific & Advisory Services
Dimitris Mavrakis, Senior Research Director

TABLE OF CONTENTS

EXECUTIVE SUMMARY 1

5G CORE NETWORKS 3
Need for Cloud-Native Architectures 3
Containers for 5G 4
Container Management Tools 5
Summary Conclusions 6

VIRTUAL MACHINES OVERVIEW 6
Multi-Tenancy via Virtualization 6
Benefits 7
Challenges 7

BARE METAL OVERVIEW 8
Increased Performance via Reduced Overhead 8
Benefits 8
Challenges 9

RUNNING CONTAINERS ON BARE METAL VERSUS A VM FOR THE TELCO ENVIRONMENT 10
Navigating the Complexity Between Both Deployment Models 10
Running Containers on Bare-Metal Servers Versus VMs 11
Key Recommendations 14

EXECUTIVE SUMMARY

Containers are a foundational element for 5G and on-demand network slicing services, both of which require light and scalable network functions. The direct benefit of containers would be improved application performance, as containerized functions are simpler and stateless. Stateless programs are generally lighter (due to the absence of persistent local data) and are easier to debug. The lightweight nature of containers allows for agile deployments of short-lived and ephemeral services, which is a distinct feature that characterizes Continuous Integration/Continuous Delivery (CI/CD) networks. Additionally, running containers is more energy efficient than Virtual Machines (VMs), as they do not require running a complete Operating System (OS) in each container, whereas each VM requires spinning up its own OS. This would equate to less consumption of memory and Central Processing Unit (CPU) resources compared to VMs.

5G containerized services will span different domains, such as central core networks and edge data centers. Containers are particularly useful at the network edge where low latency, resiliency, and portability are prioritized. The utility of containers is also emphasized in networks that are going through migrations of certain network functions to cloud-native environments. Notable 5G edge use cases that can benefit from containers are Virtualized Radio Access Networks (vRANs), Multi-Access Edge Computing (MEC), and Virtual Customer Premises Equipment (vCPE).

Fundamentally, containerized applications running on bare metal perform better than those running on VMs because they operate without the compute overhead of a hypervisor. Because of the lightweight footprint of containers, the speed of instantiating or recovering services is optimized and the reduced overhead allows for higher density of containers to run on bare-metal servers. VM instantiation, on the other hand, includes an underlying OS and disk resources that can slow down the provisioning process.

Nevertheless, containers deployed on top of VMs in a Cloud Native Function (CNF) architecture (where the hypervisor overhead is present) offer several operational benefits as the containers can have a separate lifecycle from the underlying VMs. For example, a software upgrade or recovery might not require the instantiation of a new VM, whereas software upgrades for bare-metal servers are more time-consuming. Therefore, because containers are lightweight, the benefits of starting, recovering, and upgrading services are substantially faster than it would be on bare-metal servers.

The decision to run containers on either VMs or bare-metal servers in a Communications Service Provider (CSP) network is not as clear-cut and their relative strengths and weaknesses depend on the contexts in which these deployment models are applied. As they develop their cloud-native networks, CSPs must avoid sweeping generalizations that overprescribe one technology without properly assessing the cost and time-to-market considerations of each approach. The notion that "running containers on bare metal will be the unquestionable model" for cloud-native networks is a tenuous one. CSPs must be aware that the decisions pertaining to using either VMs or bare-metal servers must be evaluated based on the considerations highlighted by ABI Research in this report.

Cloud-native transformation decisions should ultimately be oriented toward the goal of maximizing operational uptime and Return on Investment (ROI). CSPs must use the best technology to provide quicker deployments and reduce operational costs, while at the same time guaranteeing carrier-grade container standards. Containers deployed on VMs are a competitive alternative to containers on bare metal and are likely to predominate in the short to medium term due to their initial market momentum. The current shift from traditional Network Functions Virtualization Infrastructure (NFVI) architecture to a cloud-native telco cloud deployment will not be an immediate transition and will most likely undergo a gradual evolution. In the short to mid-term, NFVI and a cloud-native methodology are expected to work side by side in a CSP's network, and the coexistence of both architectures will necessitate the deployment of containers on both VMs and bare-metal servers. However, the coexistence of both VM and bare-metal based containers is also expected to carry over in the long term, with VM containers the preferred choice in the central cloud, while bare-metal containers will be widely used at the edge.

The decision to employ VMs or bare-metal servers will, in turn, be decided on a case-by-case basis. By focusing decisions based on use cases and applications, CSPs can effectively smooth the transition to cloud-native architectures. The strengths of VMs and bare-metal will be maximized in specific use cases and applications that the respective deployment models are well equipped to handle. The benefits of running containers on VMs for a CSP's transition to cloud native can be summarized as follows:

- CSPs would have more flexibility in dynamically scaling network performance as VMs can easily run multiple hosts to accommodate containers running on differing OS, while bare-metal servers are limited to the lone OS kernel of the host. This will also allow CSPs to incrementally build out their cloud-native networks in a cost-efficient manner.
- VMs' current mature ecosystem, comprehensive orchestration options, and requisite security/compliance standards would also help a faster transition to cloud-native architectures compared to a nascent ecosystem of bare-metal servers.
- CSPs are already familiar with managing VMs and putting CNFs in VMs will be a continuation of existing strategies. Deploying bare-metal systems would add complexity, as CSPs will need another team to manage bare-metal-related functions.
- Many network functions are not cloud native yet, meaning they still need VMs to run VNFs. At present, because bare-metal container solutions are still immature, nearly all the commercial deployments of container solutions in the telco industry have been VM-based containers. In the current ramp up of 4G and 5G services, VM containers and other interdependencies, such as the Virtual Network Function Manager (VNFM), Virtual Infrastructure Manager (VIM), and underlying hardware, are provided by the same vendor as the VNFs. The reduced complexity of single-vendor interdependent architectures enhances time to market for a cloud-native network.

5G CORE NETWORKS

NEED FOR CLOUD-NATIVE ARCHITECTURES

Cloud-native networks will be the cornerstone for CSPs to unlock the commercial potential of a full Standalone (SA) 5G core network. Leading CSPs, such as Orange, AT&T, Vodafone, etc., are currently investing in and, in some cases, mandating cloud-native methodologies and DevOps patterns of software development.

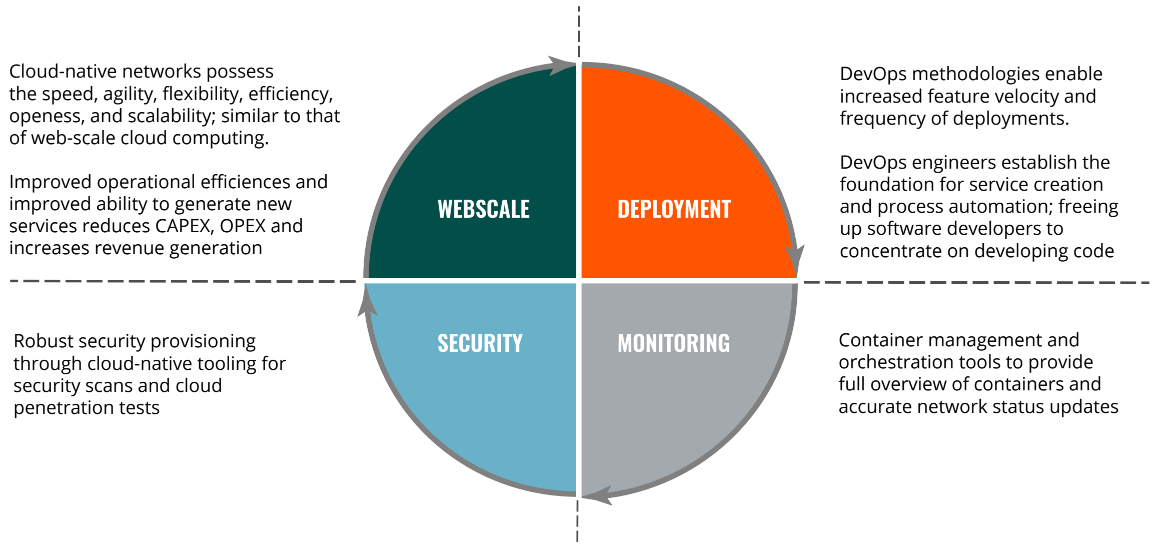
The DevOps software development model aims to streamline the collaboration model of software developers and the IT department. DevOps engineers establish the foundation for service creation and process automation. This will free up software developers to concentrate on developing code, while DevOps engineers can expedite the testing and deployment of the code in a more automated process without additional human overhead. CI/CD is a DevOps tactic focusing on software-defined life cycles and automation. Specifically, CI/CD is a practice that ensures that all changes to code (i.e., bug fixes, configurations, feature upgrades) are always in a deployable state. This on-demand deployment model will be key in providing the dynamism and versatility of network service delivery in SA 5G.

The vision of cloud-native methodologies promotes scalability, complete resilience across hybrid (virtual and physical) architectures, and ultra-rapid deployment and innovation cycles for new features, applications, and services. Specifically, there are five broad benefits that come from a cloud-native network:

1) **Enables Scalability:** Cloud-native operations scale to tens of thousands of self-healing multitenant nodes, while retaining the ability to function when the load increases.
2) **Increases Agility and Maintainability:** A loosely coupled, lightweight virtualization approach (i.e., containers) gives CSPs the agility to quickly create new products and services.
3) **Realizes Resiliency:** CSPs are seeking systems that are able to adapt to changing conditions in the network. Cloud-native methodologies provide functionality in robustness and availability.
4) **Improves Efficiency and Resource Utilization:** CSPs can benefit from the efficiency of resource utilization obtained from using containerized microservices.
5) **Avoids Vendor Lock-In:** Cloud-native computing promotes ecosystem openness, which, in turn, enables cross-vendor, cross-domain deployments on any public, private, and hybrid cloud.

Figure 1: Objectives of Cloud-Native Methodologies

(Source: Cisco/ABI Research)


Cloud-native networks possess the speed, agility, flexibility, efficiency, openess, and scalability; similar to that of web-scale cloud computing.
Improved operational efficiences and improved ability to generate new services reduces CAPEX, OPEX and increases revenue generation
WEBSCALE
DEPLOYMENT
DevOps methodologies enable increased feature velocity and frequency of deployments.
DevOps engineers establish the foundation for service creation and process automation; freeing up software developers to concentrate on developing code
SECURITY
MONITORING
Robust security provisioning through cloud-native tooling for security scans and cloud penetration tests
Container management and orchestration tools to provide full overview of containers and accurate network status updates

[1]*Radio/reference signals sent to establish a channel estimation.*

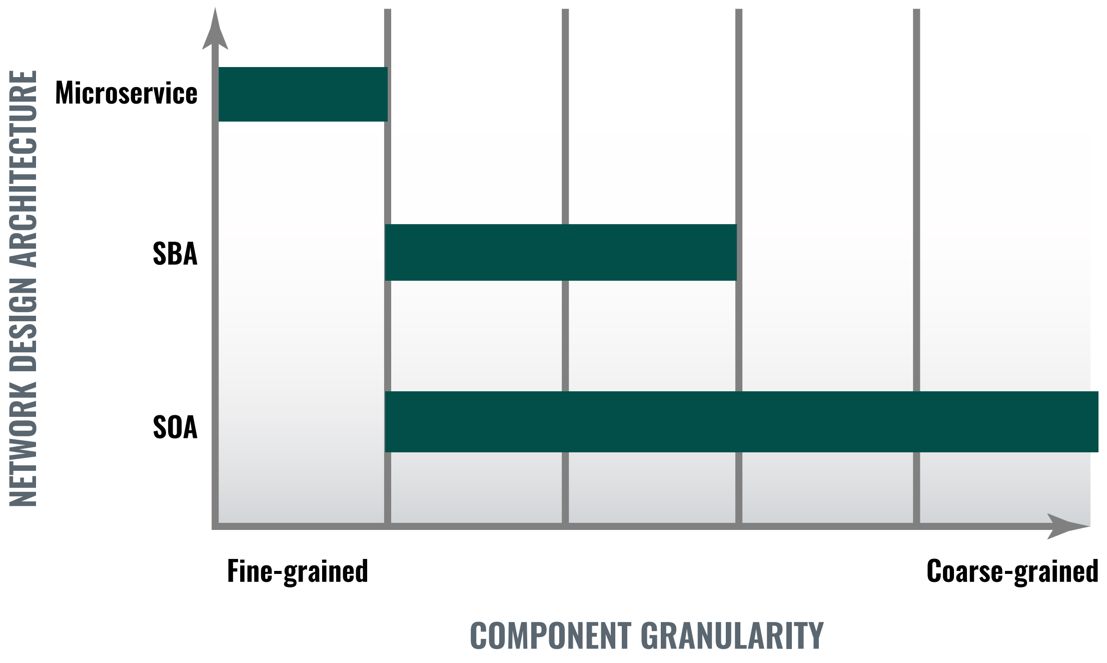
SERVICE-ORIENTED ARCHITECTURES, SERVICE-BASED ARCHITECTURES, AND MICROSERVICES

Applications built and deployed using cloud-native methodologies use what constitutes a key cloud-native pillar—microservices. Microservices present an approach to software development that deconstructs the support and architecture of the application's portfolio into small, highly-cohesive, and loosely-coupled component-like services. By segmenting an application into microservices, which can be reused for other applications, CSPs can create a more flexible and dynamic network infrastructure and processes. As opposed to traditional monolithic applications, microservices can be developed independently without affecting the rest of the microservices that make up the application. This improves the efficiency and productivity of developers by allowing for parallel development. This concurrent work is not disruptive; code execution is maintained, as the microservices continue to work together reliably with stable Application Programming Interfaces (API), despite the impermanence of the fast-changing code.

Microservices can scale on-demand to handle on-premises workload bursts that can be spun up on the public cloud. In contrast to coarse-grained models, such as a Service-Oriented Architecture (SOA), which revolve around large applications and "imperative/prescriptive" programming language (script-like), a microservice is a "declarative" service-based architectural design pattern. In other words, a DevOps automation methodology predicated on a declarative (model-like) programing paradigm reports the properties of end state and then takes automatic steps to achieve it. Between these two extremes lies Service-Based Architecture (SBA), which is a hybrid approach that preserves the architecture style of microservices, but increases service granularity in the core network.

Figure 2: Network Design Architecture (Microservices, SBA, and SOA)

(Source: ABI Research)



CONTAINERS FOR 5G

Containers provide small, shared images by isolating the OS, arguably the bulkiest part, and isolating application dependencies. Containers are a foundational element for 5G and on-demand network slicing services, both of which require light and scalable network functions. The direct benefit of containers would be improved application performance, as containerized functions are simpler and stateless. Stateless programs are generally lighter (due to the absence of persistent local data) and easier to debug. The lightweight nature of containers allows for agile deployments of short-lived and ephemeral services—a distinct feature that characterizes CI/CD networks. Additionally, running containers is more energy efficient than VMs, as they do not require running a complete OS in each container, whereas each VM requires spinning up its own OS. This would equate to less consumption of memory and CPU resources compared to VMs.

5G containerized services will span different domains, such as central core networks and edge data centers. Containers are particularly useful at the network edge where low latency, resiliency, and portability are prioritized. The utility of containers is also emphasized in networks that are going through migrations of certain network functions to cloud-native environments. Notable 5G edge use cases that can benefit from containers are vRANs, MEC, vCPE.

Aside from enabling these telco use cases, containers are also instrumental in supporting high-bandwidth, high-reliability, and low-latency use cases of different enterprise verticals (e.g., healthcare, financial services, and manufacturing) by virtue of containers being the suitable mode of delivering applications directly to the enterprise premises. Establishing localized compute and processing resources at the edge domain is a prerequisite to enable these high-impact, low-latency use cases across these verticals. Beyond providing lower latencies, these edge sites must also allow for workload consolidation and accommodate the specific performance, security, and availability standards of the verticals that they support. Containers will also be useful for MEC sites that aim to support multiple industries also necessary for onboarding third-party applications.

Another consideration for operators is that there is no uniform edge cloud architecture. Some edge deployments are decentralized clouds operating autonomously from the broader network, while other edge deployments might rely more on distributed hierarchy architectures or a centralized cloud that pools control functions in a central location. Containers, through the use of cloud-native APIs and components, are the ideal method of deploying workloads across this heterogenous cloud architecture environment.

CONTAINER MANAGEMENT TOOLS

The trend toward the decomposition of applications into containerized microservices has driven the need for an orchestration tool to manage the influx of containers within a deployment. This need has opened up a market for container orchestration tools that can automatically orchestrate life cycle management and autonomously determine where and when a container should be provisioned into an application.

Google's open-source project, Kubernetes, is increasingly becoming the preferred container management tool. Kubernetes' popularity as an emerging container orchestrator tool can be seen by the variety of Kubernetes-based offerings across multiple vendors. Kubernetes can be deployed as a Container-as-a-Service (CaaS) (via Ericsson Cloud Container Distribution or Google Kubernetes Engine (GKE)), a Platform-as-a-Service (PaaS) (through Amazon Web Services (AWS) Elastic Kubernetes Services), or bundled as a combination of PaaS and CaaS (i.e., Red Hat OpenShift Container Platform (OCP)).

The Cloud Native Computing Foundation (CNCF) monitors Kubernetes development and maintains a Kubernetes conformance test to ensure the interoperability of Kubernetes products among different vendors. The standardization and versatility of the container orchestration tool will benefit end users by giving them additional autonomy in their cloud-native buildouts by avoiding vendor lock-in and enabling a cost-efficient multi-vendor setup that is commensurate with respective network demands.

Container orchestration frameworks are key enablers for CI/CD deployments. As mentioned earlier, CI/CD is a common DevOps practice that empowers development teams to seamlessly deliver code changes iteratively in a more automated process.

SUMMARY CONCLUSIONS

The advent of cloud-native architectures requires cloud-native tools that can help CSPs handle diverse requirements of different end verticals and spur continuous innovation in the telco environment. CSPs that are initiating early adoption of cloud-native tools, technologies, and processes will be in a strong position to gain a competitive advantage and hasten their transition to SA 5G.

Containers are expected to be a significant part of a CSP's push toward a cloud-native network due to its capabilities in resource management (isolated dependencies optimize compute resources) and operations (quicker upgrades relative to OpenStack) that container technologies bring forward. A CSP's decision-making in the process of selecting container deployment models for its cloud-native 5G deployment depends on several factors. In order to fully assess the right deployment models for containerized environments, CSPs need to have an accurate assessment of the maturity of their current network, the parameters and requirements of the use cases that they desire to offer, and the cost considerations in deploying either solution.

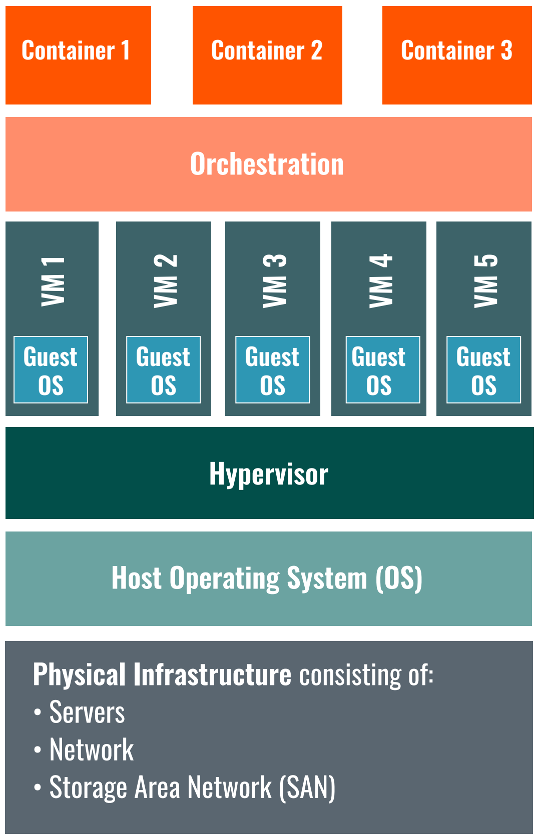
Section 3 and Section 4 compare and contrast the respective benefits/challenges of VMs and bare-metal servers. The information gleaned from the comparison between both options will then be used as a basis to inform both the short- and long-term strategies of running containers on bare-metal and running containers on VMs in Section 5.

VIRTUAL MACHINES OVERVIEW

MULTI-TENANCY VIA VIRTUALIZATION

Figure 3: Virtual Machine Components

(Source: ABI Research)



VIRTUAL MACHINES

Virtual Machines (VMs) are primarily used in cloud computing environments to meet two (2) requirements:

- To provide isolation multiple workloads and entities through the **hypervisor** by anstracting a server's software from its underlying hardware.
- To avoid resource monopolization, while preserving multi-tenant provisioning.

VM-centric virtualization lays the groundwork for:

- **Infrastructure-as-a-Service** (IaaS; e.g., servers, storage, networking) cloud-computing model;
- which in turn serves as the foundation for **Platform-as-a-Service** (PaaS; e.g, development tools, business analytics, databases) and **Software-as-a-Service** (SaaS; e.g., hosted apps) deployment environments.

Server virtualization has become the preferred method of application deployment in the enterprise datacenter with the main benefit being that VMs are able to s**upport multiple guest operating systems.** This **multi-tenant environment** would be able to support a diverse range of containerized microservices.

BENEFITS

MULTI-TENANCY

VMs enable the consolidation of much more functionality in less hardware by using virtualized workloads. This added versatility can allow users to accomplish a broader range of tasks on the same server. A multi-tenant environment enables this by dedicating a share of the respective resources that each instance requires. In this scenario, each instance's associated data and workloads are treated separately and isolated away from other tenants' workloads. From the perspective of running containers on VMs, a multi-tenant environment is also a single container cluster serving multiple projects wherein each project is isolated from each other.

VM containers in multi-tenant environments have better resource utilization. VMs also occupy server resources, as each VM ties up its allotted resources even when not running. The hypervisor layer abstracts the server's OS from its hardware and allocates the requisite CPU, Random Access Memory (RAM), and other server resources allows multiple VMs to co-exist in a single hardware platform. This partitioning of resources is more efficient when compared to bare-metal servers that are ideally suited for single-tenant environments.

PORTABILITY AND SCALABILITY

In terms of container orchestration in different cloud environments, running containers on VMs would be preferable, as workloads can be conveniently scaled/ported between hosts that do not necessarily need to have the same underlying host OS. Using a particular virtualization landscape through VMs can establish a consistent software environment to facilitate running containerized applications in different cloud environments.

CHALLENGES

COMPROMISED COMPUTE

VMs inherently have a substantially higher footprint in terms of CPU, memory, and storage capacity due to the additional layers of abstraction. Even before running the actual application code, a VM would need to use storage resources to host a guest OS and would have to dip into CPU and memory resources to run all the necessary system processes.

"NOISY NEIGHBORS"

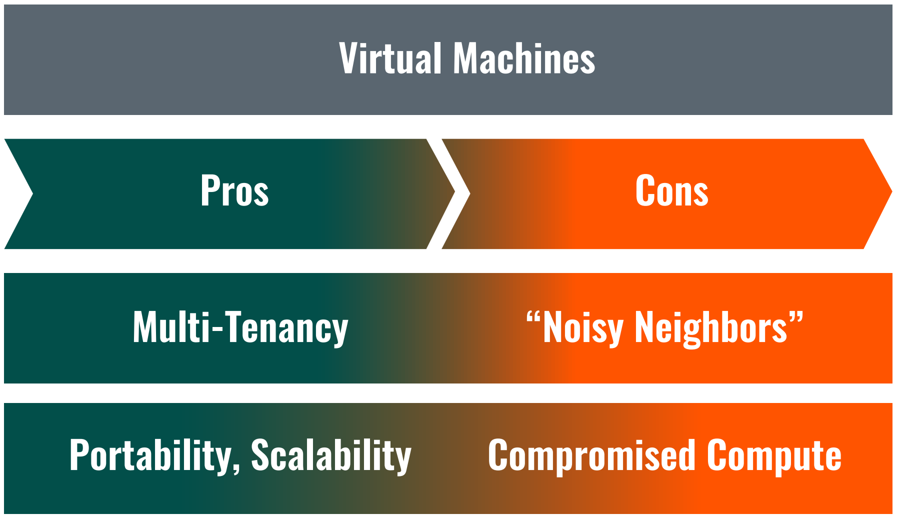
Another disadvantage of having multi-tenant environments is the possibility of reduced server performance due to the "noisy neighbors" scenario. Noisy neighbors are instances when particular VMs are running inordinately data-intensive processing compared to the other tenants in the server. This imbalance will reduce the resources available for the other applications and might also slow down the running times of other VMs, as the VM server might not be able to handle the performance demands due to the limitations of compute and power.

The different application containers residing inside a VM are limited to the finite amount of CPU, memory, and Input/Output (I/O) resources available to them. An excessively resource-consuming or aberrant container can interfere with the performance of other concurrently running containers. Because virtualization management solutions cannot address this issue, the solution is often to run just one container per VM, resulting in a decrease in container density. However, if a VM is used by only one tenant, the impact between containers is ensured by the VNF vendor. In practice, noisy neighbor interference is not huge. In the bare-metal container scenario, the container isolation is suboptimal and will have increased potential vulnerabilities to noisy neighbor interferences.

Deploying VM containers, however, will require OpenStack, while bare-metal containers do not. VM containers would, therefore, require the maintenance of more objects. This would require more emphasis on management due to the increased complexity (relative to bare-metal containers) of the technology stack. Conversely, the advantage to this is that VM capabilities can be provided and maximized through this setup.

Figure 4: Pros and Cons of VMs (Summary)

(Source: ABI Research)

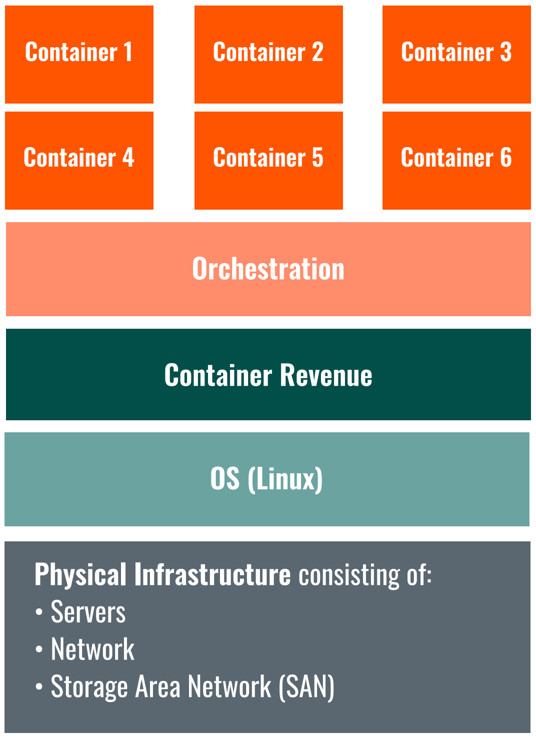


BARE METAL OVERVIEW

INCREASED PERFORMANCE VIA REDUCED OVERHEAD

Figure 5: Bare-Metal Server Components

(Source: ABI Research)



CONTAINERS ON BARE_METAL

- Bare metal servers are very similar to dedicated servers in that they are both **single-tenant machines.** This type of machine provides users with total access to the hardware, translating to higher compute capabilities.
- This superior compute power relative to VMs can be acheived due to how bare-metal servers **do not use a hypervisor layer.** To eliminate the requirement for virtualization layers and reduce the compute overhead of the system, the OS is loaded directly onto the server.
- Bare-metal servers, however, have their disadvantages. Assigning all workloads to bare-metal servers is inefficient as they are **costly and inflexible in handling dynamic workloads.**

BENEFITS

PERFORMANCE

Bare-metal servers are dedicated servers that have better compute capabilities compared to VMs due to: 1) their direct access to the processing resources; and 2) how they operate with less compute overhead from the lack of a hypervisor. Furthermore, the complete control and isolation of a server's physical resources increases performance for demanding applications and can easily handle data-intensive workloads. Single tenancy removes the impact on the performance and stability of other users within the same server because there is no other user on a bare-metal server. This can translate to enhanced disk and Input/Output Operations Per Second (IOPS).

Containers run on bare-metal use system resources more efficiently than VM-based containers. With a dedicated server, users get complete control over the physical machine. They have the flexibility to choose their own OS and avoid the "noisy neighbor" challenges of shared infrastructure, and can finely tune hardware and software for specific data-intensive workloads. It is worthwhile to note, however, that the performance improvements of running containers on bare metal over VMs is not significant. In third-party lab trials, it was estimated that the performance improvement was only 8%.

CHALLENGES

COST

While VMs can be more costly, requiring one more license for the hypervisor compared to bare-metal servers in VM-to-VM container transitions; transitioning from a conventional network to VM containers would also require more licenses compared to bare-metal containers. Certain processes also translate into higher costs for bare-metal servers. An example would be physical server upgrades. To upgrade a bare-metal server, operators must recreate the container environment from scratch on the new server. If the container environment were part of a VM software image, operators could simply move the image to the new host.

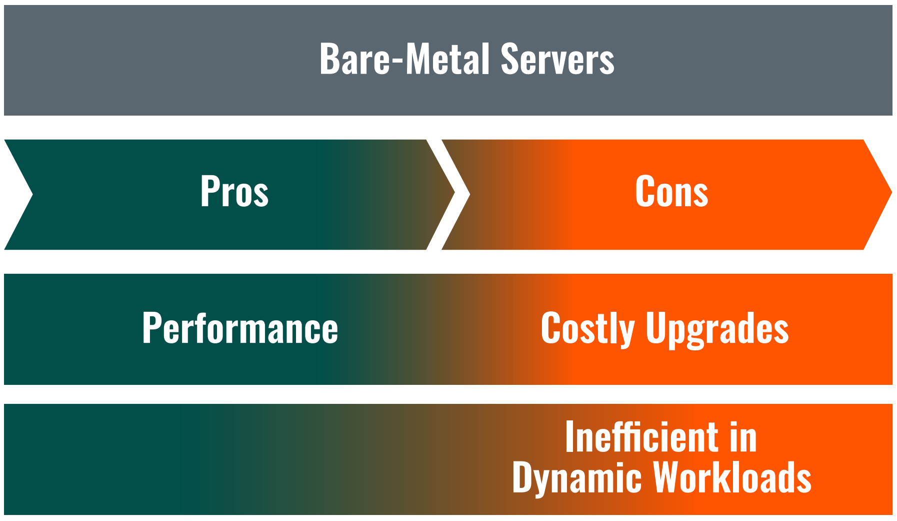
INEFFICIENT IN DYNAMIC WORKLOAD ENVIRONMENTS

Bare metal is not the ideal solution in an environment with highly variable workloads where fast spin-up/down times (not performance) are prioritized. Bare-metal servers are not suitable for supporting applications that have fast spin-up/down times or temporary workloads that only run for a short period of time. In an environment that aims for high scalability, VMs would be the better choice. Bare-metal deployments that aim to support variable workloads will require buying, installing, and updating new equipment, which is a costlier and more time-consuming process. CSPs would have to power down the unit, which would disrupt the network up-time to make the changes.

Container environments are OS-dependent, so one that is built for Linux will only operate on Linux, for example. That will potentially put limits on migration and may work against cloud deployments using bare-metal servers that do not operate on Linux as bare-metal servers only support one host OS. This is a disadvantage, as containers are designed to be able to migrate workloads between on-premises sites and public cloud locations.

Figure 6: Pros and Cons of Bare-Metal Servers (Summary)

(Source: ABI Research)



RUNNING CONTAINERS ON BARE METAL VERSUS A VM FOR THE TELCO ENVIRONMENT

NAVIGATING THE COMPLEXITY BETWEEN BOTH DEPLOYMENT MODELS

Fundamentally, containerized applications running on bare metal perform better than those running on VMs because they operate without the compute overhead of a hypervisor. Because of the lightweight footprint of containers, the speed of instantiating or recovering services is optimized and the reduced overhead allows for higher density of containers to run on bare-metal servers. VM instantiation, on the other hand, includes an underlying OS and disk resources that can slow down the provisioning process.

Nevertheless, containers deployed on top of VMs in a CNF architecture where the hypervisor overhead is present offer a number of operational benefits, as the containers can have a separate life cycle from the underlying VMs. For example, a software upgrade or recovery might not require the instantiation of a new VM, whereas software upgrades for bare-metal servers are more time-consuming. Therefore, because containers are lightweight, the benefits of starting, recovering, and upgrading services are substantially faster than they would be on bare-metal servers.

The decision to run containers on either VMs or bare-metal servers in a CSP network are not as clear-cut and their relative strengths and weaknesses depend on the contexts in which these deployment models are applied.

- For example, in terms of cost, physical server upgrades are generally more costly for bare metal. VMs, however, are inherently more costly given that VMs require one more license for the hypervisor compared to bare-metal servers.
- From a security perspective, bare-metal servers, in theory, can address compliance and security issues better than VMs because they provide a single-tenant environment that not only improves performance, but can also ensure protection of sensitive data. However, the virtual environments of VMs are isolated and can equally ensure security and privacy as well as bare-metal servers. Due to the presence of a hypervisor, running containers on VMs would be more secure (relative to running containers on bare metal), as there is a separation between container workloads and each VM. Bare-metal servers do not have this inherent separation between underlying container workloads.
- Determining the utility of VMs and bare-metal servers based on scalability and flexibility is not straightforward. Using a particular virtualization landscape through VMs can establish a consistent software environment to facilitate running containerized applications in different cloud environments. Bare-metal servers, on the other hand, can also support scalability by either changing to more powerful servers, adding extra RAM, or adding additional servers to support workloads. Bare-metal servers can come in a broad range of configurations and be customized to address specific use cases.
- In terms of performance on VMs, different guest OSs can be selected for each application to achieve optimal performance. Bare-metal containers, however, can only be configured for common configurations, while VMs can be optimized accordingly. On the other hand, bare-metal servers have better compute capabilities compared to VMs due to: 1) their direct access to the processing resources; and 2) how they operate with less compute overhead from the lack of a hypervisor.

Given this complexity, the choice of using VMs or bare-metal servers for running containers is not a straightforward decision and it would be more prudent for operators to select the most appropriate deployment model for their circumstances.

ABI Research believes that using the appropriate deployment models should be driven by:

- The operational and cost considerations associated with transitioning a current network toward cloud native
- The identified use cases that can benefit the most from the respective strengths of each deployment model

RUNNING CONTAINERS ON BARE-METAL SERVERS VERSUS VMS

As they develop their cloud-native networks, CSPs must avoid sweeping generalizations that overprescribe one technology without properly assessing the cost and time-to-market considerations of each approach. The notion that "running containers on bare metal will be the unquestionable model" for cloud-native networks is a tenuous one. CSPs must be aware that the decisions pertaining to using either VMs or bare-metal servers must be evaluated based on the considerations highlighted by ABI Research in this report.

PERFORMANCE

Running containers on VMs enables different guest OSs to be selected for each application to achieve optimal performance. Bare-metal containers, however, can only be configured for common configurations, while VMs' OS kernel performance can be optimized based on the different service-level requirements of different applications. Furthermore, running Kubernetes on bare metal has comparatively lower resource utilization and weaker scheduling capabilities compared to VMs. This would be disadvantageous in multi-vendor scenarios, as the resource usage efficiency would decrease due to how bare-metal servers run applications in a siloed manner on a unified OS.

In multi-tenant scenarios, hosts need to be used for isolation. As a result, resource pools are divided into silo modes, reducing resource utilization. In addition, the more silo resource pools, the more complex the management, such as preparing spare parts and upgrading hardware.

DEPLOYMENT AND MAINTENANCE

VMs can use a single IT platform for easier management, monitoring, and optimization. Bare-metal servers, on the other hand, may have increased complexity in management (especially in multi-tenant configurations) due to how bare-metal workloads are segregated within discrete pools of compute, storage, and network resources of a single vendor. In VM container deployments, each vendor can bring its own CaaS. CIM calls VIMs using standard OpenStack APIs, which facilitates integration. Invoking between applications and the CaaS is complex. Therefore, the integration and Operations and Maintenance (O&M) costs are significant.

Resource/service decoupling, which is when tasks or applications consume resources independently of each other, is easier at the virtualization layer. Bare-metal servers do not have the virtualization layer and would, therefore, experience greater difficultly achieving service decoupling at the container layer. VNFs in bare-metal containers are more heavily coupled with host OSs and hardware than VM containers. For example, a VNF upgrade may lead to an OS upgrade due to compatibility issues between the image and OS. An OS upgrade would lead to an upgrade of all VNFs, which would result in network disruptions—an unfavorable scenario in the telco domain.

Lastly, the virtualization aspect in the hypervisor layer and VMs brings about better management of configuration/updates and the persistence of data allows for a rollback to previous configurations in case of failures, making VMs a better option for a multi-vendor environment. Running containers on bare-metal servers in multi-vendor environments, on the other hand, is more complex. The number of interfaces between a bare-metal server's host OS and containers is estimated to be more than 300. Host OS upgrades (which require reinstallation, affecting run time) would necessitate the upgrades of supported containers and associated interfaces. Extra hardware is also necessary when migrating services. Additionally, tenants need to be isolated using physical hosts in bare-metal servers, resulting in resource fragmentation and low resource utilization in multi-tenancy scenarios. Furthermore, vulnerabilities, such as a misconfiguration, may cause unauthorized access to containers, which is an additional attack surface for malevolent actors.

SECURITY AND RESOURCE ISOLATION

The main benefit of running containers on VMs over bare metal is based on how it can enable a quicker route to cloud-native deployments relative to bare metal. This is mainly due to how the portability of CNFs across different environments with a different OS is limited for bare metal, as each container makes use of the OS kernel of the host. VMs, on the other hand, allow CNFs to move between hosts easily due to the ability of VMs to support multiple guest OSs through the hypervisor.

VMs also allow CSPs to mitigate the risk in transitioning to cloud-native through node splitting. Node splitting reduces the risk of all clusters failing, which is more prevalent in bare-metal deployments that are limited to hosting a single node per server. In a VM container scenario, when VMs of multiple Network Functions (NFs) are deployed on the same physical server, the hypervisor can be used to implement strict resource isolation between NFs to avoid resource contention between the NFs. For example, Quality of Service (QoS) isolation capability of the virtualization layer can be used to implement strict isolation of storage and network forwarding bandwidth between NFs. In the bare-metal container scenario, the container layer does not have a complete resource isolation mechanism. NF containers share and compete for resources and will impact services running. For example, if a large number of storage I/Os have access to the container of NF 1, the container of NF 2 may fail to access the remote storage, causing an internal exception of the NF.

The lack of resource isolation inherent in bare metal (due to a single host OS) is a vulnerability, as containers on the server will be more susceptible to container faults in the event of server breakdowns or security attacks. After pods are breached, the impact extends to the host environment. A compromised container in bare-metal environments may affect the performance quality of other containers in the server. Additionally, the risk of storage network isolation is greater than that of VM containers. Because the storage network is globally interconnected at Layer 2, the risk is extended to the entire site. To put it simply, security risks will occur when the storage does not support multi-tenancy (globally interconnected at L2), whereas tenant 1 will be able to view the volume information of tenant 2 and vice versa.

Virtual Local Area Network (VLAN) isolation or multiple physical storage devices isolation need to be used for bare metal. Bare-metal containers require more complex security detection isolation, tools, and integration solutions. The reason is that containers on the same host may have different capabilities and privileges. Given that the isolation capability is less capable than that of VMs, there has to be minimum prescribed privileges attached to the containers. This is a more complex endeavor.

Figure 7: Security and Resource Isolation Summary

(Source: ABI Research)

	Containers on Bare-Metal	**Containers on VMs**
Multi-tenant, Full Isolation of Shared Physical Machine	Lack of resource isolation of bare metals (due to a single host OS) will lead to a less secure environment and the attack surface is large (there are 250+ call interfaces for bare-metal).	Better security due to hypervisor's strict resource isolation policies. Attack surface is lower, with only 10+ virtual hardware interfaces.
Method of Security Isolation in Multi-Tenant Environment	Physical machine isolation	VM isolation
Granular Management of Multi-tenant Security Isolation	Deploying and managing containers on bare-metal is not ideal as different containers may have different capabilities and privileges. Shared kernel will limit bare-metal servers to basic configurations for containers.	VMs' OS kernel performance can be optimized base on the different service level requirements of different applications.
Impact of Container OS Single-Point Failure	Shared kernel will cause the server to be more susceptible to container faults in events of server breakedowns or security attacks.	Node splitting will mitigate the risk of all clusters failing in events of server breakdowns.

RELIABILITY

The use of OpenStack in containerized VM environments provides a strong QoS mechanism to ensure optimal performance and prevent interference. Bare-metal servers that only run Kubernetes (without OpenStack) are relatively less equipped for QoS assurance, as they lack key features (e.g., storage QoS and network QoS). Heavy traffic pods deployed on the same host will result in traffic congestion, possible system restarts, and/or compromised backend storage.

TRANSITION TO CLOUD-NATIVE NETWORKS

Containers will be gradually relied upon more as CSPs transition toward cloud-native networks and the selection of the right container deployment model will be highly dependent on where CSPs are in this transition. In the interim, VMs and bare metal servers are expected to co-exist as CSPs progressively adopt containers within their network architecture. The issue with this co-existence is that both modes will be using differing hardware resource pools, which would result in heavy manual workloads for resource arrangement.

Given this, initial CSP cloud-native deployments are oriented more toward building their stack on a virtualization layer (e.g., the DISH in the United States is still running VMware on top of AWS and Ericsson cloud-native deployments are still based on OpenStack), relegating the bare-metal deployment option to the secondary option for now. Bare-metal deployments will more likely materialize in the longer term, given that the technology and applicable use cases are not fully mature and widespread.

USE CASE-DRIVEN DEPLOYMENT MODEL

Bare-metal servers definitely have their place in certain use cases that are ramping up in popularity. Edge cloud deployments can benefit from containers on bare metal. Edge environments have a smaller footprint and running containerized microservices at the edge would be more viable through bare-metal servers that can support higher container densities and enable better resource maximization. Containers on bare metal would be more ideal for use cases that demand lower latencies. Bare-metal servers can quickly spin up services (in milliseconds) that only require a few minutes or hours of uptime (e.g., cloud gaming or functions with stringent timing/synchronization protocols—fronthaul).

However, the disparity between the time it takes for the service creation/startup/scaling speed between bare-metal servers and VMs is not that wide when accounting for peripheral factors. Figure 8 highlights the latency differences between bare metal and VMs based on specific conditions. The text in bold in the table represents the parity in performance of VMs relative to bare metal in the stated conditions.

Figure 8 Deployment Comparison

(Source: ABI Research)

	Containers on Bare-Metal	Containers on VMs
Creation Time	~ 10s	When the VM is not ready: Minute-level + Creation time **When the VM is ready: Same as the bare-metal container**
Startup Duration	~ 1.5s	**If the VM startup duration is not included, then less than 1.5s** If including the VM startup Duration, then 30s
Scaling	Within seconds as no pre-creation or temporary VM creation and management is required	**Second-level: When VMs are already pre-created and managed** Minute-level: For temporary VM creation and management.

The suitability of the respective deployment models for edge sites ultimately varies depending on which services these edge sites are intended to support. Edge sites are usually designed to support multiple services and facilitate the coexistence of multiple applications for different verticals. These wide-ranging services encompass finance, video surveillance, healthcare, and cloud gaming applications, just to name a few. It would be narrow-sighted for CSPs to rely on a single container deployment model without accounting for the actual requirements of the use cases being supported at these edge sites. Different applications will need different resources depending on QoS requirements associated with the respective vertical. Financial applications, for example, will need VM containers due to their superior security capabilities, while deployed User Plane Function (UPF) in MEC deployments are more suited to bare-metal containers that can help meet the low latencies of MEC use cases. CSPs would also benefit from having a more flexible deployment model (VMs, rather than bare metal), as the network parameters and unique requirements of these enterprise use cases are still in the early stages of development.

Additionally, use cases at the core require more management and security (especially for multi-vendor environments). These security considerations will also prompt CSPs to use containers running on VMs over bare-metal servers. CSPs will also have to bear in mind that containers on bare metal have not yet seen mass commercial deployments for CSPs and might consider deploying containers on existing architecture (i.e., VMs), which might give them an earlier go-to-market strategy for cloud-native deployments.

KEY RECOMMENDATIONS

Cloud-native transformation decisions should ultimately be oriented toward the goal of maximizing operational uptime and ROI. CSPs must use the best technology to provide quicker deployments and reduce operational costs, while at the same time guaranteeing carrier-grade container standards.

Containers deployed on VMs are a competitive alternative to containers on bare metal and are likely to predominate in the short to medium term due to their initial market momentum. The current shift from a traditional NFVI architecture to a cloud-native telco cloud deployment will not be an immediate transition, but rather will undergo a gradual evolution. In the short to mid-term, NFVI and cloud-native methodology are expected to work side by side in a CSP's network, and the coexistence of both architectures will necessitate the deployment of containers on both VMs and bare-metal servers. The coexistence of both VM and bare-metal-based containers is also expected to continue into the long term, with VM containers the preferred choice in the central cloud, while bare-metal containers will be widely used at the edge.

The decision to employ VMs or bare-metal servers will, in turn, be decided on a case-by-case basis. By focusing decisions based on use cases and applications, CSPs can effectively smooth the transition to cloud-native architectures. The strengths of VMs and bare metal will be maximized in specific use cases and applications that the respective deployment models are well equipped to handle.

The benefits of running containers on VMs for a CSPs' transition to cloud-native can be summarized as follows:

- CSPs would have more flexibility in dynamically scaling network performance, as VMs can easily run multiple hosts to accommodate containers running on differing OSs, while bare-metal servers are limited to the lone OS kernel of the host. This will also allow CSPs to incrementally build out their cloud-native networks in a cost-efficient manner.
- VMs' current mature ecosystem, comprehensive orchestration options, and requisite security/compliance standards would also help a faster transition to cloud-native architectures compared to a nascent ecosystem of bare-metal servers.

- CSPs are already familiar with managing VMs, and putting CNFs in VMs will be a continuation of existing strategies. Deploying bare-metal systems would add complexity, as CSPs will need another team to manage bare-metal-related functions.
- Many network functions are not cloud native yet, meaning they still need VMs to run VNFs. At present, because bare-metal container solutions are still immature, nearly all the commercial deployments of container solutions in the telco industry have been VM-based containers. During the current ramp up of 4G and 5G services, VM containers and other interdependencies, such as the VNFM, VIM, and underlying hardware, are provided by the same vendor as the VNFs. The reduced complexity of single-vendor interdependent architectures enhances time to market for a cloud-native network.

Published October 2021

©2021 ABI Research

157 Columbus Avenue 4th Floor

New York, NY 10023   United States

Tel: +1 516-624-2500

www.abiresearch.com