Containers have emerged as a red-hot topic among Gartner's clients. This research note answers the most common questions I&O leaders are asking about how to realize the agility that containers promise.

Key Findings

- The container ecosystem is complicated to understand due to the plethora of new and esoteric tools, and the unique problems they solve when compared with the traditional platforms (such as virtualization) that are well-understood by I&O.
- Enterprises are beginning to evaluate containers for traditional, monolithic applications to modernize their delivery along with the ability to run on a variety of infrastructure.
- Building and deploying cloud-native, container-based applications in production require substantive experience with architecting and managing distributed systems, which can be complex on their own.
- The Microsoft Windows container ecosystem is not yet mature enough for widespread production use and lacks broad support from container orchestration tools.

Recommendations

I&O leaders responsible for accelerating infrastructure innovation and agility should:

- Adopt public cloud-based container-as-a-service platforms for turnkey deployment and operation of containers without the heavy operational burden on-premises.
- Employ PaaS frameworks on-premises or in their own public cloud IaaS accounts, when desiring cloud service provider independence yet with best-of-breed cloud-native architecture support.
- Use containers throughout the full life cycle of their applications, including development, test and production, to reap the full benefits of containers.
Weigh the costs required to rearchitect an existing application into a microservices architecture against the potential gains before embarking on long projects that may not be worth it. If the decision is made to rearchitect, then start with the stateless portions first.

Analysis

One of the challenges in understanding containers from the perspective of I&O is that there is not a one-for-one analog between elements in container and virtual machine (VM) deployments. Further, nearly any workload can be virtualized with immediate benefits. But the same is not true of containers: Some workloads will work better than others. This research answers the 10 most common questions from I&O leaders to better understand container deployments and the ideal workloads to deploy when using them.
Table 1. Questions and Answers About Docker, Containers and Microservices

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer Highlights</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. What are the most appropriate uses of containers?</td>
<td>Microservices-based applications are the ideal type of application to use with containers, though enterprises see packaging benefits when using containers with monolithic applications.</td>
</tr>
<tr>
<td>2. What are the critical components that make up a container deployment on-premises, and how are they used?</td>
<td>Container deployments require radically different components compared with VMs (such as an image registry, orchestration and scheduling, container runtime, and security components).</td>
</tr>
<tr>
<td>3. What’s the state of container services in the public cloud?</td>
<td>Most deployed containers exist using public cloud IaaS. As a result, cloud IaaS providers are innovating at a rapid pace because of the enormous opportunity to support a new paradigm of cloud-native workloads.</td>
</tr>
<tr>
<td>4. What’s the relationship between products (such as Cloud Foundry and OpenShift) as it relates to containers?</td>
<td>Cloud Foundry is &quot;opinionated&quot; in its use of containers. As a result, there are limits to use cases and patterns that could be achieved with containers used on their own, but Cloud Foundry can make it easier to use containers in a safe and efficient way.</td>
</tr>
<tr>
<td>5. What’s the state of Windows containers? How does VMware factor into containers?</td>
<td>The current state of Windows containers is not ready for production use at scale. VMware’s vSphere Integrated Containers (VIC) provides a pathway for virtualization admins to integrate a Docker engine with vSphere and run it inside lightweight VMs.</td>
</tr>
<tr>
<td>6. Should I choose a container as a service, Kubernetes distribution or the bring your own (BYO) container management solution?</td>
<td>Ultimately, customers need to choose between having the control that comes from an array of tuning parameters or the simplicity that comes from an &quot;opinionated&quot; installation.</td>
</tr>
<tr>
<td>7. Is the container technology inherently less secure than VMs? What security best practices should we follow?</td>
<td>VMs and containers both achieve security through logical isolation. Neither is inherently unsecure, but the specific means by which they are secured are different.</td>
</tr>
<tr>
<td>8. Are containers suitable for processing sensitive data and applications?</td>
<td>Service providers are building compliance-specific capabilities into their container offerings to directly address HIPAA, PCI and similar use cases.</td>
</tr>
<tr>
<td>9. Are containers suitable for commercial off-the-shelf (COTS) applications?</td>
<td>Containers are suitable for COTS applications if supported by the vendor, have meaningful architectural benefits that support containers and fit naturally into a customer environment.</td>
</tr>
<tr>
<td>10. Should we refactor an application, such as those based on Oracle and IBM, to better support containers?</td>
<td>The effort to refactor an entire, traditional monolithic application simply to support a microservices architecture in production could be a long and arduous path that may not be worth the effort.</td>
</tr>
</tbody>
</table>

Source: Gartner (March 2018)
1. What Are the Most Appropriate Uses of Containers?

The ideal type of application to use with containers in production is a microservice. A microservice is a singularly focused task that represents a small portion of a larger application. Because a microservice is focused on a single task, it can be independently scaled from other portions of an application. For example, a microservice that generates thumbnails can be scaled up and down, independently of a microservice that serves user requests. Further, microservices can be deployed and released independently of each other. This is made possible because the microservices are singularly-focused and loosely coupled.

Microservices do not need containers to exist. However, the benefits of microservices can be accelerated and achieved more rapidly when built with container infrastructure. Software developers can iterate and improve on the microservices that they develop without affecting the microservices of other developers, as long as external-facing APIs do not break compatibility.

Most traditional enterprise applications do not follow a microservices architecture, nor are microservices suitable for all enterprises. However, some vendors of enterprise applications are beginning to offer their applications in containers for use as microservices.

2. What Are the Critical Components That Make Up a Container Deployment On-Premises, and How Are They Used?

**Management User Interface and API:** The management user interface (UI) includes an intuitive graphical UI, which is based entirely on the APIs. External systems should be able to bypass the UI and use the API directly without loss of functionality.

**Image Registry:** Public image registries (such as Docker Hub) provide repositories of publicly curated container images. Many container management systems include a private image repository where organizations can manage and curate their own images.

**Orchestration and Scheduling:** The orchestration layer interfaces with the application. It keeps the containers running in the desired state and maintains SLAs, assuming frequent software updates. It also performs rolling updates and rollbacks. Kubernetes has emerged as the de facto standard for container orchestration.

**Container Runtime:** The container runtime provides the ability of cluster nodes to fetch container images in a registry; to manufacture the right file structure to run a container on a host; to interact with network and storage plug-ins; and to create, start and stop containers.

**Security and Governance:** The security component enforces security policies, including secret management, security scanning, image signing, networking segmentation and encryption, and role-based access control (RBAC). The governance component defines, centralizes and reports on policies based on the environment resource utilization and access.

**Monitoring:** The monitoring component provides runtime visibility at the cluster node, container and microservices level. Ultimately, it will close the feedback loop with the orchestration layer, which will generate scheduling decisions. Both container runtime and kernel-level host monitoring should be available.
**DevOps Workflow:** DevOps isn’t critical to container deployments, but they are commonly used together. DevOps tooling, CI/CD pipeline toolchains and container image generators store and retrieve container images in the image registry. The tools integrate with the container management system.

3. What Is the State of Container Services in the Public Cloud?

Public cloud IaaS providers are innovating at a rapid pace because of the enormous opportunity to support a new paradigm of cloud-native workloads. A number of providers (such as AWS, Google and Microsoft) offer managed container services along with native support for Kubernetes, the leading container orchestrator.

Many of these providers require that customers still provision virtual machines to form a Kubernetes cluster. This requires that a customer have planning and forethought about the configuration of virtual machine instances in the cluster (in terms of memory, CPUs and storage), and the overall number of instances to provision.

However, the trend of cloud IaaS providers offering completely managed services without the need to deal with virtual machines or the underlying infrastructure is now being applied to containers as well. Azure Container Instances and AWS Fargate offer managed container services that abstract the underlying infrastructure such that developers focus on tasks that need to execute rather than the number of instances in a Kubernetes cluster.

For an in-depth review of this market, refer to the "Market Guide for Public Cloud Container Services."

4. What Is the Relationship Between Products (Such as Cloud Foundry and OpenShift) and Containers?

Cloud Foundry, an application PaaS platform, is an "opinionated" container framework described as such because it is preconfigured for ease of installation and maintenance, but it makes design and implementation decisions as a result of such simplicity. Recent announcements from PaaS vendors, including Red Hat for their OpenShift product, include support for serverless functions, indicating a future for all types of cloud-native application development.

When evaluating PaaS offerings for container-focused use cases it is important to note the full scope of the desired and required services. Distinguish whether they are served as supported container instances or require custom configuration and are available via third-party component stores as part of a vendors' ecosystem. Such services include service mesh, API management and gateway, messaging middleware, operations and performance monitoring, application development services, and databases.

PaaS frameworks — which are container environments extended with platform as a service features like patching, versioning and managing the health of the container instances — support:
The Open Container Initiative (OCI), which uses Docker’s container format (e.g., Cloud Foundry Application Runtime and its commercial distributions, such as Pivotal Cloud Foundry Application Service) or direct support (e.g., OpenShift, Cloud Foundry Container Runtime)

Orchestration via the de facto standard Kubernetes container orchestrator (e.g., SUSE Cloud Application Platform based on Cloud Foundry Application Runtime, Cloud Foundry Container Runtime and its Pivotal Cloud Foundry Container Service distribution, and Red Hat OpenShift)

Common services to support developer workflows (for example, Open Service Broker, which manages connections to external application dependencies)

These platforms are available both as software distributions (open-source PaaS frameworks and their commercially supported equivalents, termed "Cloud Enabled Application Platforms") and as cloud-hosted PaaS offerings.

5. What Is the State of Windows Containers? How Does VMware Factor Into Containers?

The Windows container ecosystem is not yet mature enough for widespread production use. Windows containers lack broad support from container orchestration tools. For example, the most recent Kubernetes software release (1.9) only offers beta support for Windows containers. In addition, there are very few commercial Windows applications that are supported in containers. Windows server support for authentication, security and networking is limited, and several existing components are in either "beta" or "preview" release. Despite these limitations, I&O leaders may want to consider Windows containers for specific use cases (such as containerizing in-house developed .NET applications) to derive the agility, portability and cost benefits.

VMware’s vSphere Integrated Containers provides a pathway for virtualization admins to integrate a Docker engine with vSphere and run it inside lightweight VMs. This would provide the opportunity to support established methods of maintaining availability in production environments (such as live migration), which will be necessary for stateful apps running in containers. As part of the VIC offering, VMware also provides a container management portal and private registry for securely storing images. In addition, VMware announced a managed Kubernetes service called Pivotal Container Service (PKS), which is integrated with its core products (such as vSphere, VSAN and NSX-T). While PKS uses the latest stable open-source distribution of Kubernetes, with no proprietary extensions, the underlying solution components include proprietary products from VMware (such as NSX-T).

6. Should I Choose a Container-as-a-Service Offering, Kubernetes Distribution or the BYO Container Management Solution?

There are varying products in the market that allow enterprises to deploy containers at scale. These products differ by the degree of abstraction that they provide for application developers. Among the most abstracted products are container-as-a-service offerings (many were previously PaaS products, such as Cloud Foundry and OpenShift). These CaaS offerings abstract the infrastructure details from developers and provide application tooling in a structured or opinionated manner. BYO
container management approaches offer the least abstraction and most flexibility, but with the addition of complexity.

- **BYO:** Avoid BYO except in rare cases. Few organizations have the expertise to pull it off. For organizations that want to BYO, they can forgo all of these tools and opt to integrate a number of fast-moving open-source products. This option is only recommended for the most advanced enterprises.

- **Kubernetes distribution:** Choose a less opinionated K8S distro if you have developers who are skilled, understand infrastructure (at least conceptually) and are able to get their hands a bit dirty. Some CaaS offerings are less opinionated and offer the option of some developer tooling while also giving developers the option of using the container orchestration/scheduling components (such as Kubernetes [K8S]) more directly. Other tools are just packaged K8S. They are meant for developers who want to get their hands dirty.

- **PaaS:** Use a PaaS offering if your developers could stand to benefit from a near-complete abstraction over the infrastructure and container management, and if your enterprise requires moving quickly and are just getting into microservices development.

7. **Is the Container Technology Inherently Less Secure Than VMs? What Security Best Practices Should We Follow?**

The container technology isn’t inherently unsecure. In fact, applications and users are isolated on a per-container basis to prevent compromise from other containers. However, I&O leaders should acknowledge that the integrity of the shared host OS kernel is critical to the integrity and isolation of the containers that run on top of it. In addition, intercontainer communication needs to be monitored and secured, and traditional-host-based tools are ineffective in doing it.

We advise I&O leaders to adhere to the following best practices:

- Use a hardened operating system, often a "Thin OS," which can limit the attack surface and complement it with a rigorous and automated patch management process.

- Leverage controls both during the build and run phase of containers, where applications are scanned for vulnerabilities as they are promoted through the software development life cycle process.

- Proactively detect and monitor abnormal behavior by using container granular security tools that can provide container-native and service-level views, and that can aid in the prevention of malicious application traffic.

8. **Are Containers Suitable for Processing Sensitive Data and Applications?**

Containers can be used for processing sensitive data and applications if deployed securely using a combination of existing tools (such as encryption) and newer, container-focused approaches, using templates with preinstantiated security features, networks, platforms and libraries. Containers are an ideal application delivery mechanism to limit the dependencies that an application may require to
certain versions that have been accepted by the enterprise. However, for this approach to be successful, security teams must get involved early in the process of deploying containers for sensitive data.

In reference to public cloud IaaS, the hyperscale providers are making significant inroads, specifically with their container services, in terms of satisfying compliance requirements for mandates (such as HIPAA and PCI).

9. Are Containers Suitable for COTS Applications?

Software vendors are increasingly offering their products in a container as an optional deployment model. In some cases, vendors have adopted agile development methods. End-to-end and container-based deployments are just part of their modern manner of writing, testing and delivering software. Often, these products have been rearchitected as a composable set of microservices that are a natural fit for containers. In other cases, vendors are considering container deployments because they fear the perception of being out-of-touch if they do not offer a modern mode of application delivery. In these cases, COTS and containers may be a not-so-natural fit, because either the application is still monolithic in nature or the application doesn't have properties that are conducive to independent scaling, or a combination of these two cases.

Container deployment of COTS applications may not be ideal for your enterprise if you don't already have existing container infrastructure to manage or if the requirements are nonstandard compared to your own. Additionally, Gartner clients are beginning to inquire about containerizing COTS applications that are currently deployed without the support of the vendor. In the vast majority of cases, containerizing COTS applications without explicit support from the vendor should be avoided due to the possibility of unwanted application behavior.

10. Should We Refactor an Application, Such as Those Based on Oracle and IBM, to Better Support Containers?

The decision to refactor a monolithic application to support containers will depend on whether you plan to use containers in development/test, in production or in both stages. In production, containers deliver the most benefit when deployed to support applications that have elastic workloads consisting of microservices architectures.

Alternatively, enterprises often start by refactoring the stateless portions of an application, such as the front ends of web applications, into microservices to make appropriate use of containers. Enterprises often use microservices architectures to build new applications, which eliminates the need to refactor later.

Gartner Recommended Reading

Some documents may not be available as part of your current Gartner subscription.

"Four Key Container Deployment Considerations for I&O Leaders"
"How to Prepare Your Enterprise for a Docker Containers Initiative"

"Market Guide for Container Management Software"

"Preparing for Containers in the Windows World"

"What You Need to Know About Docker and Containers"

"Docker Containers Will Impact Enterprise Storage Infrastructure"

"How to Select a Storage Approach for Persistent Containers"

**More on This Topic**

This is part of an in-depth collection of research. See the collection:

- Research Roundup: Building and Marketing Cloud-Based Offerings — 1Q18