IBM Bluemix Container Service

A Bluemix offering built on open-source Docker technology

Introduction

Containers technology originated over 20 years ago with web-hosting vendors seeking to optimize the density of websites residing on each server in a datacenter. IBM, Sun, Google made key contributions to those early iterations. More recently, by isolating an application and its dependencies inside a container, Rocket and Cloud Foundry have evolved standards for working with containers within cloud infrastructure. And Docker has eliminated the issues that previously resulted in a containerized application working in one environment but not another.

In the context of IBM's partnership with Docker, this document provides an overview of IBM Bluemix Container Service (IBCS) as an enterprise-ready solution for using Docker in application life-cycle management.

Docker

Docker container technology is open-source with over 2,000 contributors and driven by a 12-member governance advisory board selected by the community.

Docker's key innovation was simplifying the container consumption model, eliminating vendor lock-in and promoting application portability. An obvious indication of its value is that, since Docker's launch in March 2013, users have downloaded over 2 billion Docker images from the public repository.

Enterprise Technology Research found 97 percent of enterprises surveyed had imminent plans to adopt Docker technology.*

*ETR : All-Time Spending Intention Leaders by Brad LaScolea

Definitions

Docker image

The building blocks from which containers are launched. An image is the read-only layer that never changes. Images can be created based on the committed containers.

Dockerfile

A text document that contains all of the required commands to build a Docker image.

Docker registry

A registry server that provides hosting and delivery of images.

Layer

Each file system that is stacked when Docker mounts rootfs. The base image is the read-only layer, and each application added to that image is its own layer on top of the previous ones in existence.

Docker container

A running instance, generated from a Docker image. A self-contained environment built from one or more images. Information available at the container level includes the image from which it was generated, memory used, IP address assigned to it and other pertinent information.

IBM and Docker Partnership

In December 2014, as part of an ongoing commitment to open-source technology, IBM formed an initial partnership with Docker, renewing it in February 2016. Through this partnership, by further abstracting container code from host infrastructure, the two companies build on Docker's open platform, enhancing the toolchain and streamlining the user's experience from application development through deployment.

By providing choice in deployment, overlay networking, scalable groups, integrated monitoring/logging/security, IBM makes Docker containers enterprise-ready.

Besides IBM Bluemix Container Service, IBM® UrbanCode™ Deploy and IBM PureApplication® also both support native Docker containers.

Open Container Initiative (OCI)

Docker announced the Open Container Initiative in June 2015. The OCI is an open governance structure for creating open industry standards around container formats and runtimes. Docker seeded OCI with runC from Docker and commits to keeping this technology free from reliance on any particular company, operating system, stack or client. IBM plays an important role within OCI by participating on the board, providing core code maintainers and leveraging OCI technology for its offerings.

Introducing IBM Bluemix Container Service

IBM Bluemix Container Service is one of the first hosted Containers–as-a-Service (CaaS) offerings, beginning in June 2015. Designed for enterprise workloads that require inherent scalability and reliability, IBM Bluemix Container Service provides users the lifecycle management tools required to help securely acquire Docker base images; build in application content; and deploy, run and manage containerized applications on Bluemix.

IBM Bluemix Container Service coexists with Cloud Foundry application runtimes, Bluemix infrastructure virtual machines, and event driven actions via OpenWhisk as compute options that give customers flexibility in managing their workloads.

Reviewing IBM Bluemix Container Service offerings

IBM Bluemix Container Service (IBCS) offers three types of service--public cloud, on-premises, and for application image or content delivery.

Cloud

Based on customer requirements, IBM provides three options for using containers in the cloud—public, dedicated and local. Containers in the public cloud are hosted on a multi-tenant host. Dedicated containers reside on their own host in an IBM datacenter. And local containers run on hosts in the customer's own datacenter that IBM monitors and manages on a subscription basis.

On-premises

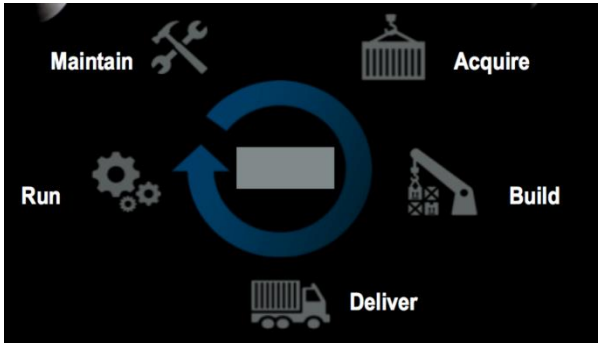
Either through a collaborative refactoring or lifting and shifting systems and services, the IBM Bluemix Container Service on-premises option integrates containers into a customer's own continuous development pipeline.

Content

With the content option, IBM Bluemix Container Service delivers Liberty profile, the IBM MobileFirst™ platform, StrongLoop Process Manger and IBM DB2® Express as Docker images that run in any environment that supports the Docker engine.

Managing an application lifecycle with IBM Bluemix Container Service

In production, running an application in a container image involves a continuous process, including the versioning that marks the end of one image and the beginning of the next.



Let’s take a closer look at the s lifecycle and some of the IBM Bluemix Container Service and tools relevant to each step.

Step 1 - *Acquire*

Step 2 - *Build*

Step 3 - *Deliver*

Step 4 - *Run*

Step 5 - *Maintain*

Step 1 - *Acquire*

The container lifecycle starts with creating a content acquisition plan. And the first step in creating that plan usually involves answering these basic questions:

- *Where should I start?*
- *Is there content in Docker Hub that I can leverage?*
- *Are there official images I'm going to use?*
- *Are there standard images that we're going to build inside our enterprise and make available to our internal development teams?*

Docker Hub is the place where the Docker community shares and leverages existing Docker collateral, and stores their images either in a public or private repository.

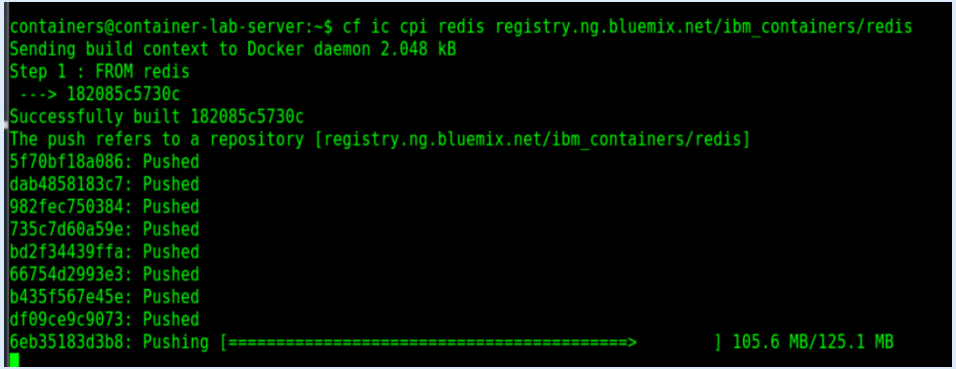
Besides offering Docker images preloaded with specific IBM software, the IBM Bluemix Container Service offers enterprise users a private registry to hold images tagged with relevant Docker namespaces.

A user's private Docker image registry in IBM Bluemix Container Service is closely linked to the container runtime environment.

Private Image Registry

IBM Bluemix Container Service enables users to push, pull or copy images to their private registry, which allows rapid deployment of containerized apps within the Bluemix cloud.

Using Docker containers in the IBM Bluemix Container Service begins with instantiating a first container or uploading existing images to a private, hosted registry.



One of the great things about Docker is the idea of layered images. That an application can derive from a base image that itself derives from a different base image enables vendors to scalably extend software capabilities in an ongoing set of images while leveraging contributions from the Docker community.

Step 2 - *Build*

The second step in the container lifecycle is the build process.

Here the questions usually include:

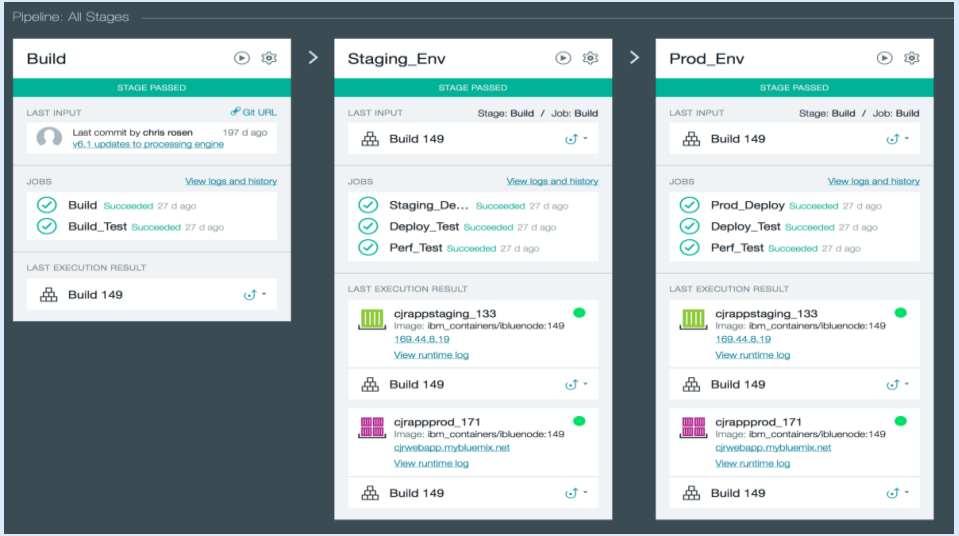
How do I take an acquired image, actually build in my application, then run the containerized app through an automated delivery pipeline?

With IBM Bluemix Container Service, a shared Docker build service in Bluemix automates building and publishing a Docker image into the user's private registry for deployment on the cloud platform. Because the same Dockerfile used on one's laptop can be used in IBM Bluemix Container Service, consistency in building upon any image and ultimately running a containerized app remains paramount.

Step 3 - *Delivery*

The delivery and deployment step involves answering these questions:

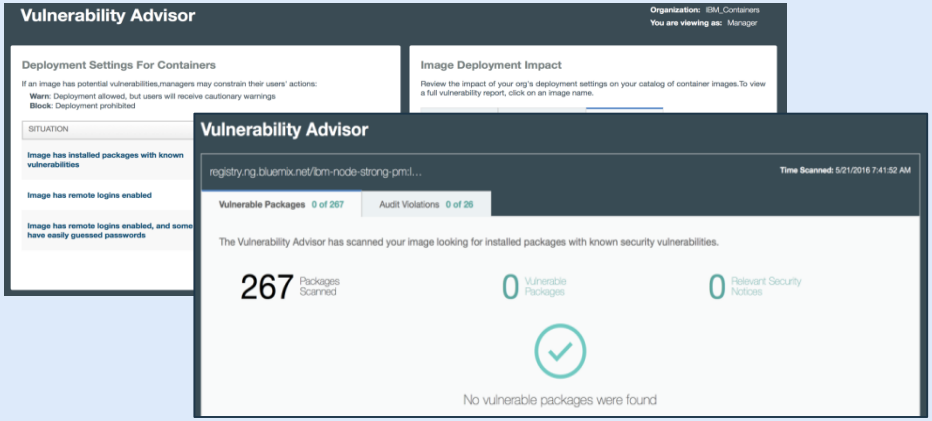
- How do I integrate this with my DevOps pipeline?
- What is the process of deploying that application into production?
- In addition to the delivery of the app the first time, how do I continuously develop and deploy enhancements to that environment?
- How do I orchestrate the deployment in a non-destructive way and perform canary testing?
- A user needs to determine if the new version is performing as it should prior to routing all traffic to this version while supporting an automated rollback capability.



The integrated delivery pipeline within Bluemix allows users to define where and how their application is deployed. For example, the delivery pipeline supports logic that takes a source code change in a Git repository as a trigger to a build and auto-deploy a QA version of an application and impose a gate for manual approval before final delivery of the build into production. The pipeline service uses tools like Jenkins to deliver that end-to-end automation.

Vulnerability advisor

IBM Bluemix Container Service helps integrate security into the deployment process. As part of the release in July 2015, we began to include a Vulnerability Advisor (VA), which allows you to scan any image, regardless of source, prior to deploying your first live container from that image.



VA performs an agentless introspection, scanning both for known vulnerabilities in each layer within the image, and weak policies (pertaining to password settings, for example) or configuration oversights (**sshd** enabled is common). By scanning from a policy and content perspective, VA determines a security posture before you instantiate your first live container, reducing the environment's exposure to attacks. VA also allows the administrator to control whether images with known vulnerabilities are allowed to be deployed or should be blocked.

Step 4 - *Run*

The run step sets the management system and the runtime environment around your container. In this step, you determine how to scale your application and how you would recover from failure. You also determine how to connect your container to other services and applications in your environment.

IBM Bluemix Container Service are deployed to bare metal for increased performance while helping to maintain security using IBM Security's broad portfolio. The IBM Bluemix Container Service also provides monitoring and logging, security compliance insight, overlay networking, native Docker CLI support, load-balanced container groups, auto-recovery, auto-scaling and fully qualified domain names.

Because IBM manages the host infrastructure and security of the environment, the process of using IBM Bluemix Container Service to run containers begins with loading or building a Docker image rather than with first deploying Docker hosts on virtual machines.

IBM Bluemix Container Service supports both the native Docker Compose CLI and APIs, facilitating a smooth transition from running Docker on a laptop to running it on the Bluemix cloud platform.

Container groups

IBM Bluemix Container Service offers scalable container groups, which are not available in open-source Docker. Container groups consistently deploy the same image with the same allocated resources to all members of the group, providing flexibility to meet the requirements of specific workloads, and including an integrated load balancer to distribute incoming requests within the group.

Auto-recovery

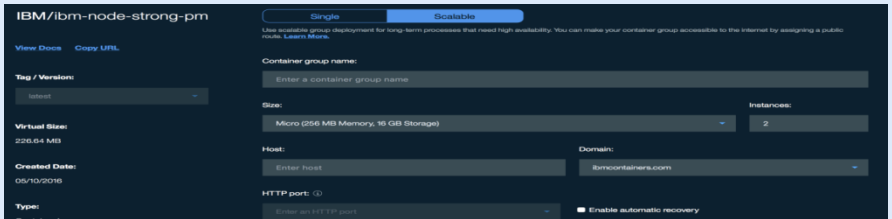
If a container fails, auto-recovery re-instantiates a new one within the container group.

Fully Qualified Domain Names

The GoRouter service exposes the group to public access through fully qualified domain names.

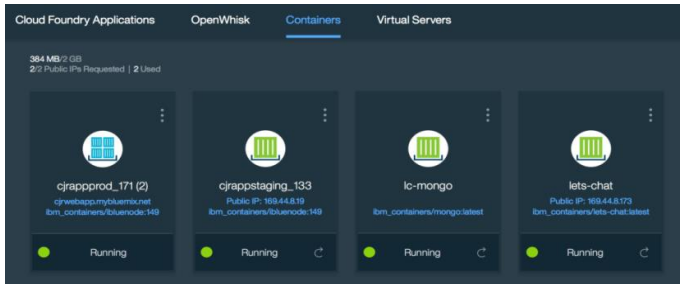
Overlay Networking

Providing a private network for each tenant that includes a private non-routed IP address for each container, and so eliminating the need for port mapping, overlay networking is another feature of the IBM Bluemix Container Service runtime environment.



Step 4 – *Run*
(continued)

In a microservices architecture, overlay networking enables each component to communicate directly with another using private networking instead of publicly routed IP addresses. For the components that require external connectivity, the user can bind a routable IP to a single container or define a hostname that governs access to a container group.



Persistent storage

While containers are ephemeral, applications require a method for storing persistent data. IBM Bluemix Container Service provides the ability to bind storage volumes to a single container or a container group. This service manages the lifecycle of those volumes as containers are stopped and started, ensuring availability of critical data.

API-accessible services ecosystem

Because not all workloads will not run inside a container, IBM Bluemix Container Service provides API access to other sources, allowing apps running inside a container to leverage any of the more than 150 IBM and third-party services, such as IBM Watson™, Analytics and Internet of Things.

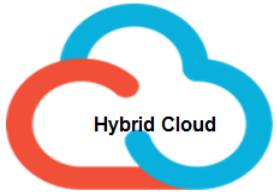
Step 5 - *Maintain*

Using open source technology from the Elasticsearch, Logstash, and Kibana (ELK) stack, IBM Bluemix Container Service integrates monitoring and logging seamlessly into the user experience, giving real-time information about failures, clues to de-bugging, and a path to making a change that repeats steps in the container lifecycle, as needed.

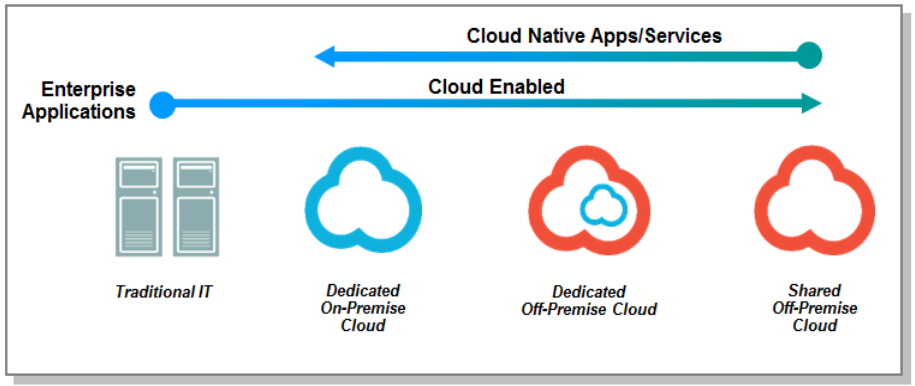
Regardless of the Docker image source, and without installing agents, IBM Bluemix Container Service consolidates the logs and monitoring data in the Bluemix console for all deployed containers. Visibility into CPU, memory and network utilization begins as soon as a user deploys a new container. These metrics are valuable in evaluating if the current container size is appropriate for running the application. And a user may also customize their Grafana charts to showcase the most relevant monitoring characteristics.

Hybrid cloud

Providing a true hybrid cloud platform requires giving users the freedom to choose and change their environments, data and services. The Open by Design™ program, reflected in IBM Bluemix®, optimizes flexibility by leveraging innovations in ongoing open-source projects such as OpenStack, Docker, Cloud Foundry, Node.js, and OpenWhisk. Yet all of that is irrelevant to enterprise users without including the tools necessary to help improve monitoring, control and security wherever possible. IBM provides the means for IT teams to determine when and where to securely run their mission critical enterprise applications, allowing the lines of business to focus on their products and markets.



IBM defines hybrid cloud as the secure consumption of services from two or more sources, including private cloud, public cloud, or traditional IT



Trademarks and notes