vyomlabs An EMERGYS Company

bmc

Solution White Paper

Five Best Practices for Improving Docker® Lifecycle Management

Ensure comprehensive management of Docker environments with BMC solutions

Table of Contents

Executive Summary

Developers are embracing Docker to meet the demands of the digital enterprise. This open platform for developing, shipping, and running software helps them accelerate and streamline the process of building next-generation distributed applications. While Docker tools deliver significant benefits across the enterprise, they can also add additional complexity and management challenges. Docker is relatively new technology, and most existing IT operations management tools are inadequate for managing Docker environments.

Docker is extremely powerful, advanced technology and managing it can become very complex, very quickly, especially for use cases that involve multi-host, mission-critical production systems. Successfully moving "Dockerized" applications into production containers requires the right management tools to ensure:

- Security
- Governance
- Automation
- Orchestration

This paper explores the challenges associated with managing Docker environments and the need for enterprise solutions that support effective management and deployment of Docker containers. Enterprise organizations need to be able to manage containers as part of a complete business service, and successfully address container sprawl, compliance, and governance issues. BMC provides solutions that meet this need with streamlined, comprehensive Docker lifecycle management that supports and enables best practices.

INTRODUCTION

Organizations are challenged to differentiate and lead by delivering new and innovative products and services. If they don't digitally transform their business, they are susceptible to disruption by smaller, more nimble competitors. In this software driven world, change and innovation run through the hands of developers.

Enter Docker, an open source technology that is rapidly gaining acceptance among developers. **Docker makes it easy for developers to package and deploy applications to different environments quickly.** A developer can program applications on a laptop, capture them as a Docker image, and then deploy Docker containers to different environments. Docker containers wrap up a piece of software in a complete file system that contains everything it needs to run: code, runtime, system tools, system libraries—anything you can install on a server. The implication is that the application or service will run the same, regardless of the environment. This is an important benefit at a time when companies are moving to hybrid clouds and want application portability.

While Docker containers deliver significant benefits across the enterprise, they can also add complexity and unique, new management challenges that cannot be addressed by traditional IT operations management tools.

A LOOK AT THE CONTAINER LIFECYCLE

To better understand the management challenges associated with Docker containers, it's important to explore their typical lifecycle. For simplicity, the lifecycle is divided into three stages, as shown in Figure 1.

1. **Development stage -** Developers quickly create and deploy Docker containers that include build artifacts such as application code and libraries. They then test the applications, fix bugs, add features or enhancements, build new Docker images, and deploy them into new containers. This iterative process continues until the product is deemed to meet the minimum viable product specifications outlined in the Agile development sprint.
2. **Application release stage -** Release managers coordinate the automation of application environments that include Docker building, testing, and deployment pipelines. Usually a release engineer is responsible for managing application releases that consist of Docker containers.
3. **IT operations stage -** The containers are deployed into production and maintained for performance and availability until they are decommissioned. This is the stage where meeting the final three challenges—orchestration and governance, security, and monitoring of containers—becomes critical.

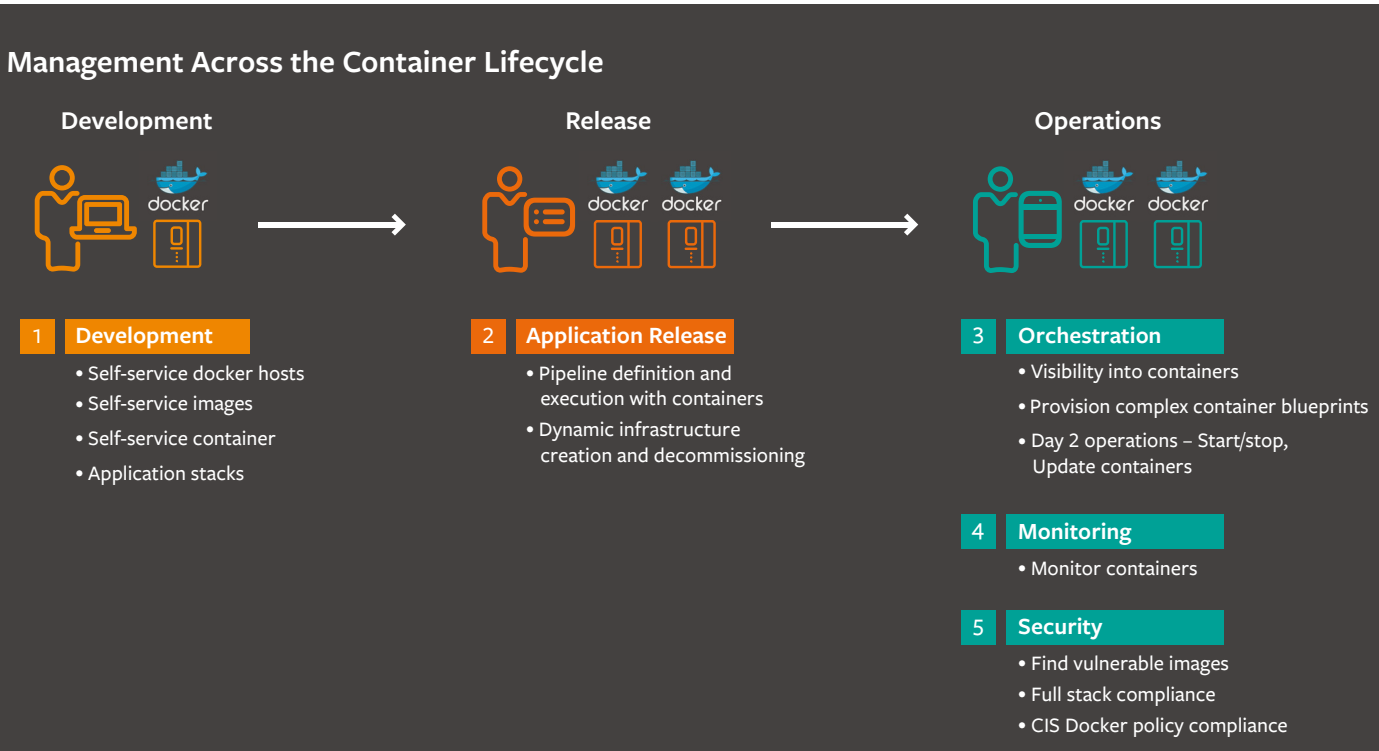

FIGURE 1: Management challenges in each lifecycle stage of Docker containers

TOP FIVE CONTAINER MANAGEMENT CHALLENGES

In order to realize the full potential of the benefits that Docker can provide, enterprise organizations need solutions designed to address these five Docker container management challenges:

1. **Lack of control –** Developers need self-service to quickly create, deploy, and test application containers, while the operations team needs control and governance to prevent container sprawl, resource overconsumption, and increased infrastructure costs. Without proper governance, environments can balloon out of control: running containers that are unauthorized, have vulnerabilities, or are no longer being used but consume valuable resources.
2. **Disconnected release cycle, fragmented processes** – As development changes increase in frequency, release coordination is necessary to maintain quality and security, and make smooth transitions through the development stages. Lack of integrated processes can negatively impact time to delivery and release quality, and even allow applications to reach production without being properly vetted.
3. **Complexity of scaling containers –** Virtualized or cloud infrastructure is not going away, and will continue to co-exist with Docker infrastructure, such as Docker hosts or Docker cluster managers (e.g. Swarm or Kubernetes) for the foreseeable future. Deploying complete applications that span across Docker and other infrastructures requires more-advanced capabilities for orchestrating deployments and managing running environments.
4. **Vulnerability protection and compliance –** Because they include parts of operating systems, Docker containers may incorporate vulnerabilities such as Heartbleed and Ghost. Protecting a Docker environment requires security at the host layer, the container, and the images. Updating immutable containers creates a new management paradigm that may shift tasks from operations to development.
5. **Unique monitoring requirements –** Docker environments require special monitoring capabilities, for example, API-level integration with Docker, and instrumentation that is built into the Docker image. These requirements are beyond the scope of most traditional monitoring tools.

Dockerfile: Instructions that tell the image builder (e.g., Jenkins) what the image should look like.

Image: The components of a Docker container at rest. They are stored in a registry and instantiated in a container at runtime.

Container: Applications and their components (code, system tools, runtime, etc.) are instantiated in a container at runtime.

Docker Engine (Host): Lightweight runtime that pulls images and creates and runs containers. It can be installed on physical, virtual, or cloud hosts.

Docker Swarm: Docker tool for managing a cluster of Docker containers.

Registry (Hub): Repository for storing Docker images, available on premises or as a cloud service.

HOW BMC CAN HELP

BMC offers solutions that enable **your enterprise to holistically manage Docker containers** across all lifecycle stages to successfully overcome common management challenges and achieve optimal benefits. People in all job roles—from developers and application release managers to IT operations staff—can manage Docker containers as services, rather than individually, and understand how each container impacts the other containers functionality.

These BMC solutions include:

- **Release Lifecycle Management** (RLM) for coordinating application releases from development through production
- **Cloud Lifecycle Management** (CLM) for deploying Docker hosts and containers into cloud environments
- **BladeLogic Server Automation** (BSA) for managing configuration and security
- **TrueSight Pulse** for Docker monitoring
- **TrueSight Operations Management** for insights and analytics

FIVE CONTAINER MANAGEMENT BEST PRACTICES

1. Combine Speed with Control (Development Stage)

Cloud Lifecycle Management addresses both self-service and governance. This solution provides a self-service catalog, as shown in Figure 2, where developers can easily request a Docker infrastructure, such as a Docker host or swarm, with a single click. The self-service catalog also makes it easier for developers to quickly identify and select existing images rather than create new, duplicate images that add to the sprawl problem. The operations team can retain control by setting quotas on containers or hosts, as well as automating the reclamation of unused containers.

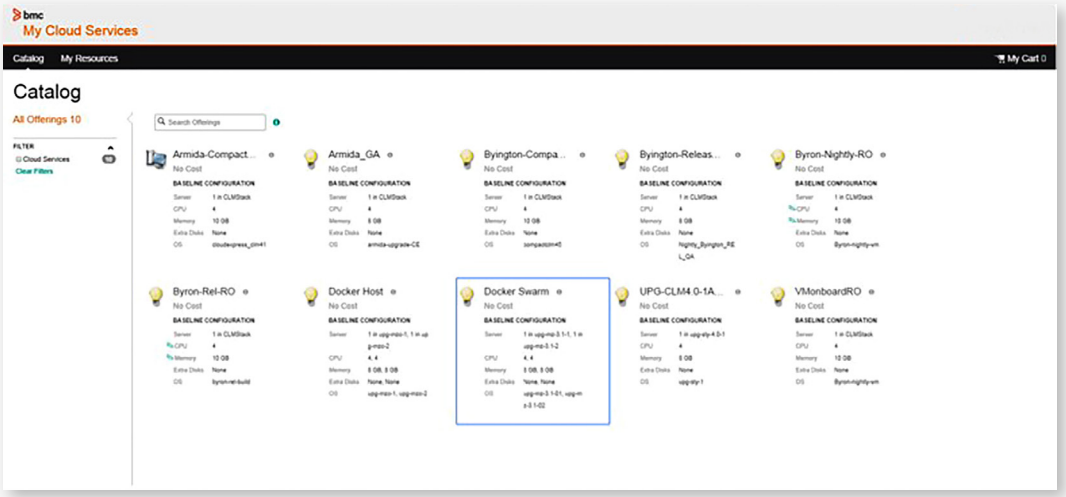

FIGURE 2: A Cloud Lifecycle Management self-service catalog for requesting Docker hosts, swarms, and containers.

A common problem encountered by early adopters is that containers are provisioned into the infrastructure and then left running even though they aren't being used. Reclamation is an important step in the container lifecycle to prevent overconsumption of server resources and potential security exposures. Container management tools should include capabilities that help identify and reclaim unused containers.

An advanced use case featuring multiple Docker application stacks can be defined using **Cloud Lifecycle Management**. A simple version of a multi-tier application stack is shown in Figure 3. This application could span physical, virtual, or cloud servers and be part of a continuous delivery pipeline, so that as new builds are produced, Docker images are created and automatically deployed into mixed environments for testing.

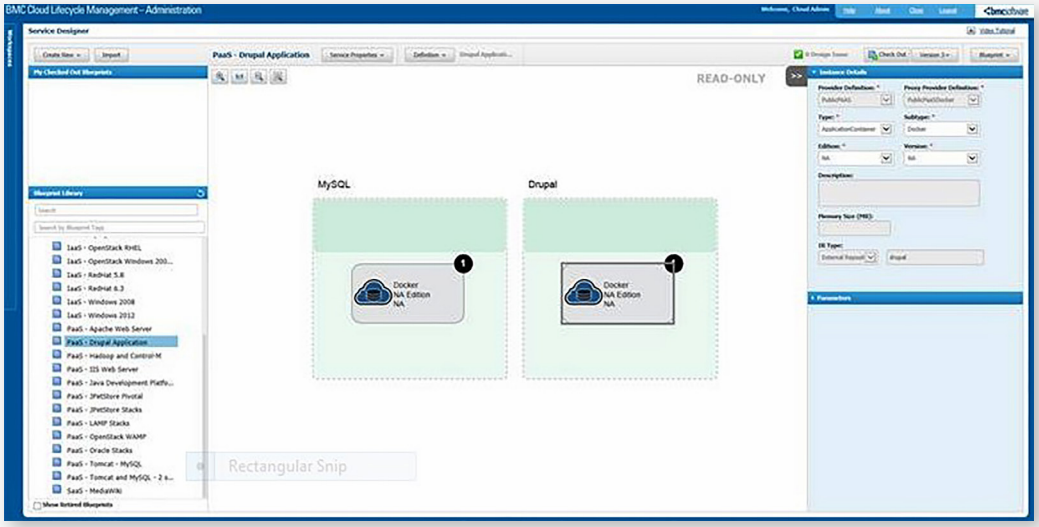

FIGURE 3: Multi-container application stacks defined in Cloud Lifecycle Management

2. Coordinate the Continuous Integration/Continuous Delivery (CI/CD) Pipeline (Release Stage)

Release Lifecycle Management not only helps govern the lifecycle of a Docker image through the various stages, it also helps coordinate application updates managed by multiple release teams across different environments to deliver quality releases successfully.

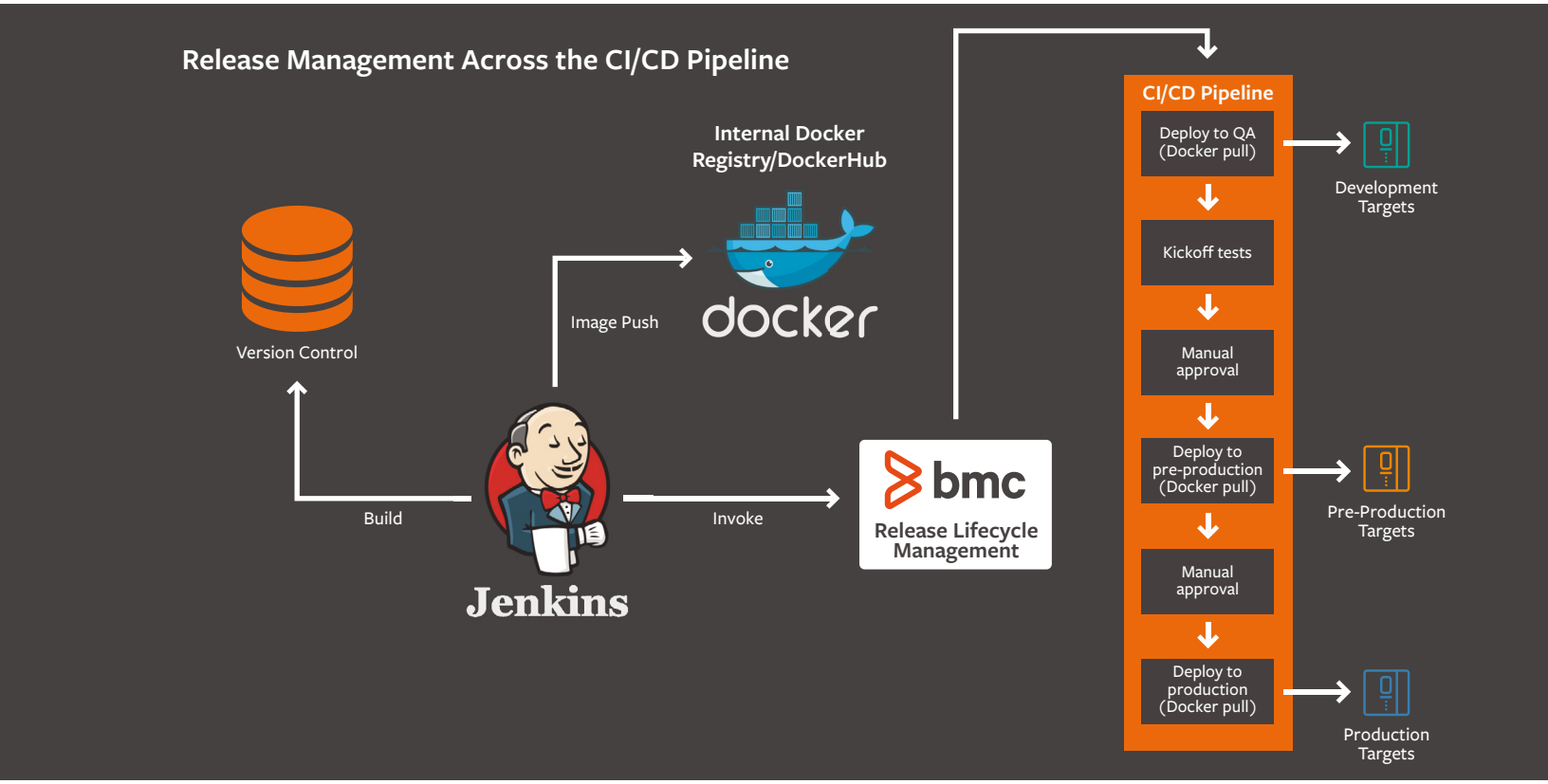

FIGURE 4: Release Lifecycle Management CI/CD pipeline

Figure 4 depicts a simple deployment pipeline where applications/configurations/environment updates move from QA to pre-production and finally to production. More complex deployment pipelines that include multiple Docker registries can also be built and managed. In these scenarios, **Release Lifecycle Management** can coordinate the movement of images between the build tools (e.g., Jenkins) and Docker registries into the various environments.

3. Streamline Orchestration and Governance of Container Infrastructure (Operations Stage)

New Docker-based cloud applications will likely integrate with applications and middleware running on virtualized or cloud systems. **Cloud Lifecycle Management** allows IT operations to deploy and manage these hybrid environments and enable specific placement policies for different types of workloads, such as Docker containers and virtual machines (VMs).

The demands of the digital enterprise call for a new suite of cloud-native and micro-services applications that require orchestration and management of multiple containers. **Cloud Lifecycle Management** enables enterprises to easily and declaratively define, deploy, orchestrate, manage, and govern multi-container hybrid workloads. Its capabilities extend beyond initial provisioning to include starting, stopping, updating, and scaling of containers, and monitoring performance.

With BMC solutions, you can:

- Automatically provision a Docker infrastructure (e.g., Docker engine/host, Docker Swarm) through a self-service portal
- Create and manage Docker infrastructure, such as a farm of Docker hosts, alongside traditional virtualization infrastructure, including workload placement policies to align business and IT goals and manage risk
- Define and deploy applications as a collection of containers and then manage ongoing operations such as starting, stopping, updating, and scaling of containers, and monitoring performance

4. Provide Security Across the Full Stack (Operations Stage)

Vulnerability testing of containers can be conducted with **BladeLogic Server Automation** tools, which perform Security Contained Automation Protocol (SCAP as defined at scap.nist.gov) testing of Docker images to **ensure that containers do not have any critical security vulnerabilities.**

To provide full-stack security, enterprises need controls for Docker infrastructure hosts, clusters, and Docker applications. **BladeLogic Server Automation** provides sample compliance policies from this specification for use in building and validating full-stack Docker compliance. Additional custom security policies can be created to prevent, for example, developers from using Docker images from non-trusted public registries. Figure 5 illustrates the status of a compliance check on containers.

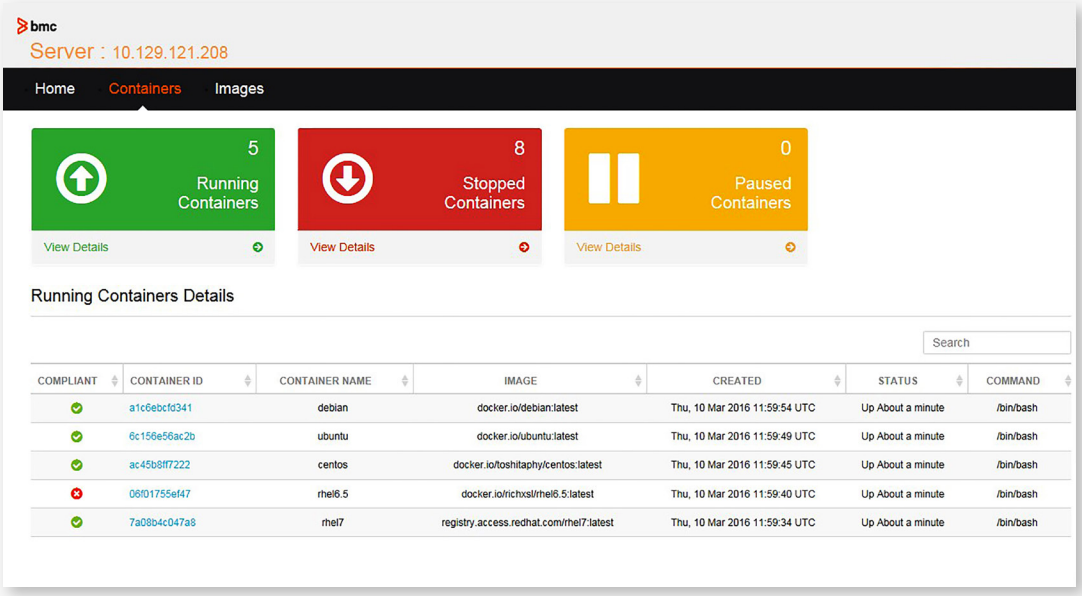

FIGURE 5: Sample status of container compliance from BladeLogic Server Automation

The development team must also ensure that it meets the needs of the security and IT operations teams, which involves validating container security early in the lifecycle. The servers used for development and testing can be managed by **BladeLogic Server Automation**, and containers that are running can be scanned for any vulnerabilities. These steps help **prevent vulnerable containers from flowing into production.**

5. Container Discovery, Visibility, and Performance Monitoring (Operations Stage)

TrueSight Operations Management and **TrueSight Pulse** provide discovery, visibility, and performance monitoring of Docker environments. These solutions provide full-stack monitoring of hosts, containers, and applications. They can discover containers and continuously collect performance metrics for each one, such as status (up/down), CPU and memory utilization, and network I/O. These metrics can be visualized in a customizable dashboard with trending and live feeds (Figure 6).

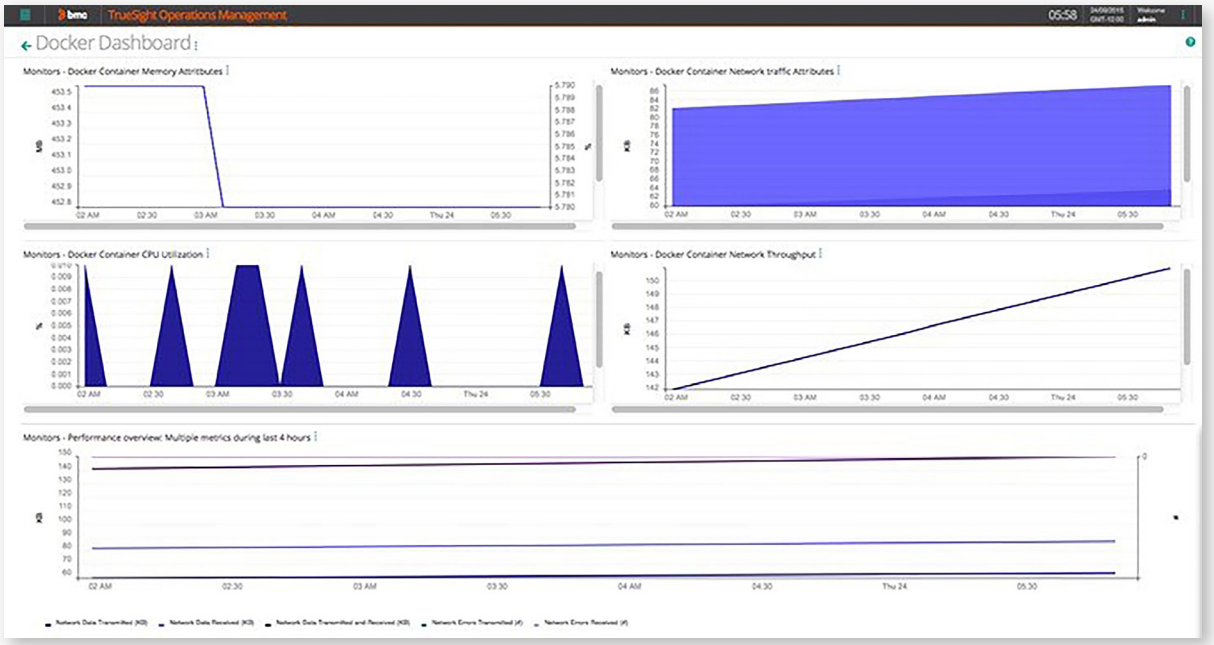

FIGURE 6: TrueSight Operations Management dashboard with customizable dashlets

The architecture for Docker monitoring with **TrueSight** is agentless. Once monitoring is set up in the containers, **TrueSight** automatically indicates when resources are overloaded so mitigation actions can be taken.

CONCLUSION

Speed is essential in the digital enterprise, and Docker helps developers innovate more quickly. However, to take full advantage of Docker benefits, organizations need the right management tools. BMC solutions enable you to effectively manage the complete Docker container lifecycle and ensure enterprise-readiness for both development and production environments—including provisioning, application release, container orchestration and governance, security, and monitoring.

With BMC solutions, enterprises can unlock the business value of Docker and securely manage the process of building next-generation enterprise and IT infrastructure.

FOR MORE INFORMATION

To learn more about how your developer organization can benefit from solutions to manage the full Docker lifestyle, please visit:

Cloud Lifecycle Management:
bmc.com/it-solutions/cloud-lifecycle-management

Release Lifecycle Management:
bmc.com/it-solutions/release-lifecycle-management

TrueSight:
bmc.com/it-solutions/truesight

475809