

The importance
of monitoring
containers



The container achilles heel

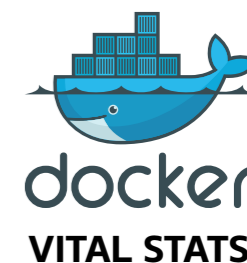
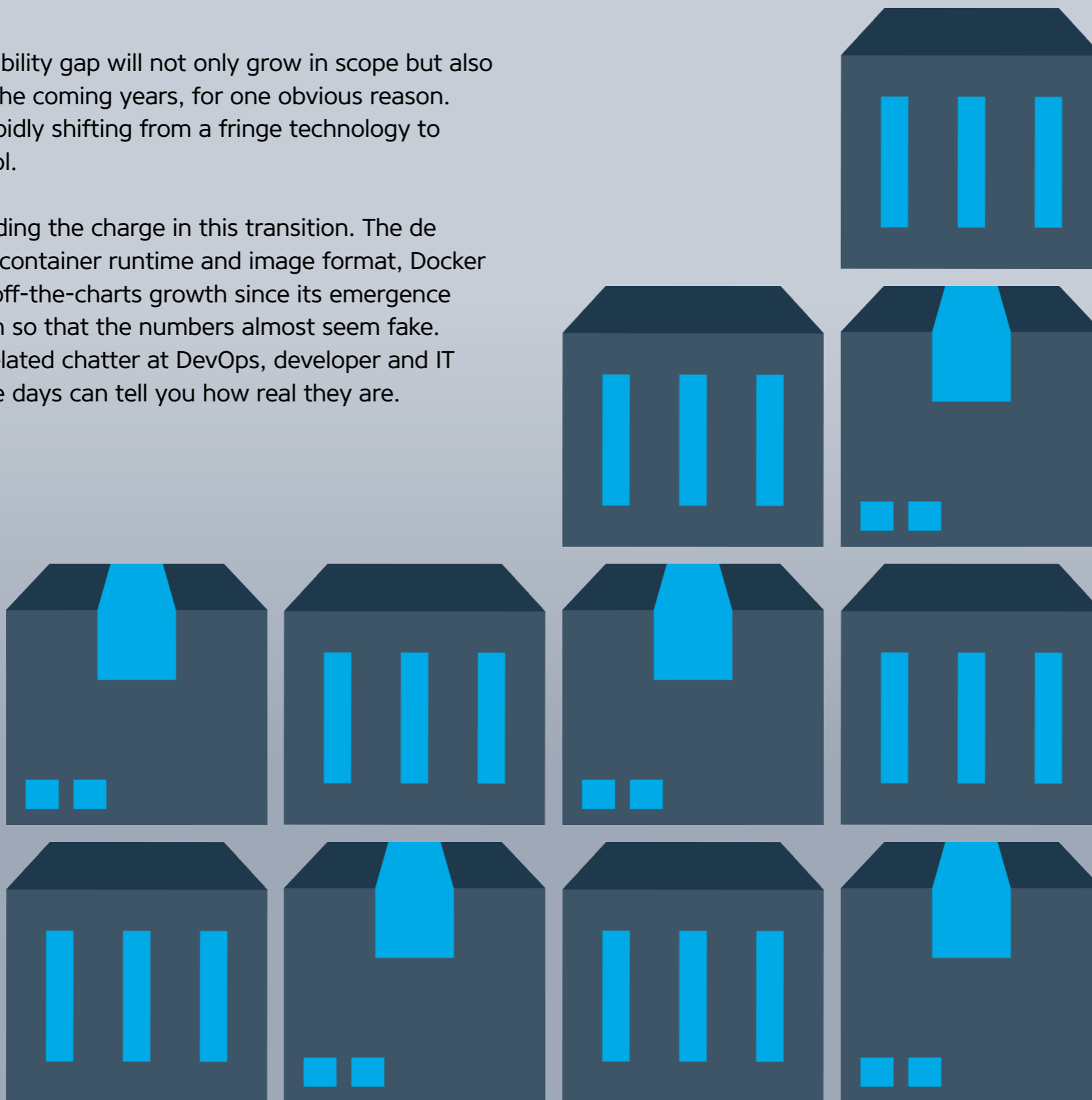
As the containerization market skyrockets, with DevOps and continuous delivery as its jet fuel, organizations are trading one set of problems for another. With container platform technology like Docker, they gain a lightweight, immutable vessel for applications that is a snap to provision.

But on the other hand, they can stand to lose a lot of visibility into what occurs within these containers. Most traditional IT monitoring applications still fail to offer a good way to track metrics related to containers or the applications they hold within. As a result, innovative organizations that depend on containers to move through their DevOps journeys tend to develop huge operational blind spots in their test, dev and demo environments.

Why container visibility matters

This container visibility gap will not only grow in scope but also in importance in the coming years, for one obvious reason. Containers are rapidly shifting from a fringe technology to mainstream IT tool.

And Docker is leading the charge in this transition. The de facto standard in container runtime and image format, Docker has experienced off-the-charts growth since its emergence in 2013—so much so that the numbers almost seem fake. But the Docker-related chatter at DevOps, developer and IT conferences these days can tell you how real they are.



TOTAL DOWNLOADS

- **320 million** Docker containers downloaded
- **18,082% jump in downloads** in the past year



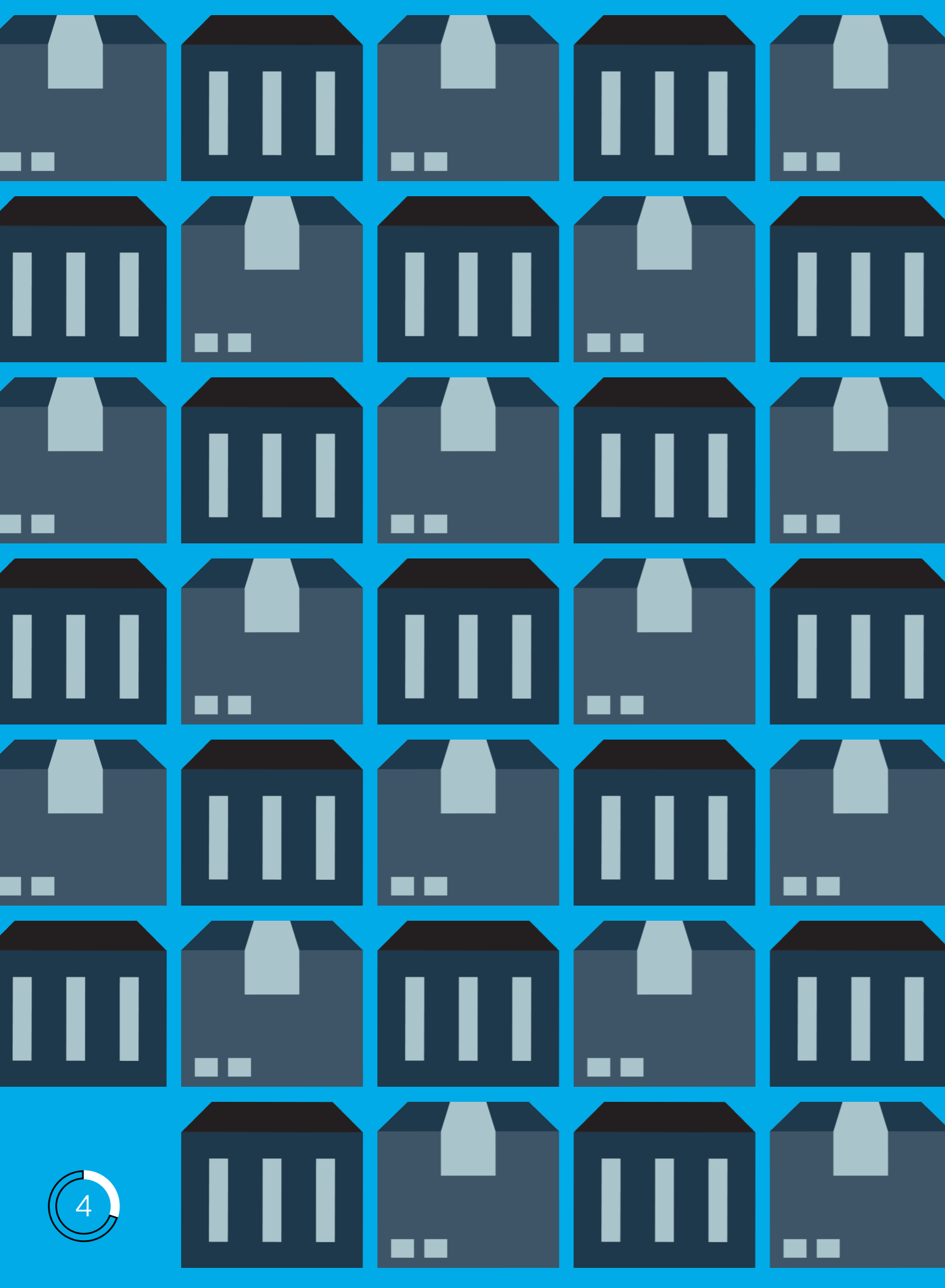
COMMUNITY

- **More than 3 million developers** downloading Docker containers
- **10,000 organizations and workgroups** set up on DockerHub
- **183% increase in contributors** to Docker project in the past year



ECOSYSTEM

- **Over 100,000** Dockerized applications
- **Over 33,000 GitHub projects** with Docker in the name
- **515% growth** in projects on GitHub in the past year

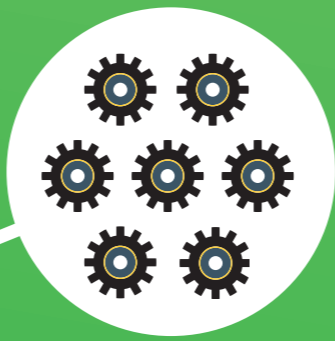


Container sprawl is coming

One of the big drivers for containers' rapid rise in popularity is the ease with which they can be spun up and down. As a result, container use has rapidly risen not only in the number of organizations depending on them, but also the volume of containers used within each individual organization.

It's a great indication of the potential of containerization in IT, but it's also the sign of an inbound issue for IT. Many technologists already struggle to deal with the problem of VM sprawl. Container sprawl is going to heap even more misery on top of VM sprawl.

That's because even though containers share similar traits to VMs, they're much more ephemeral and scale far more rapidly.

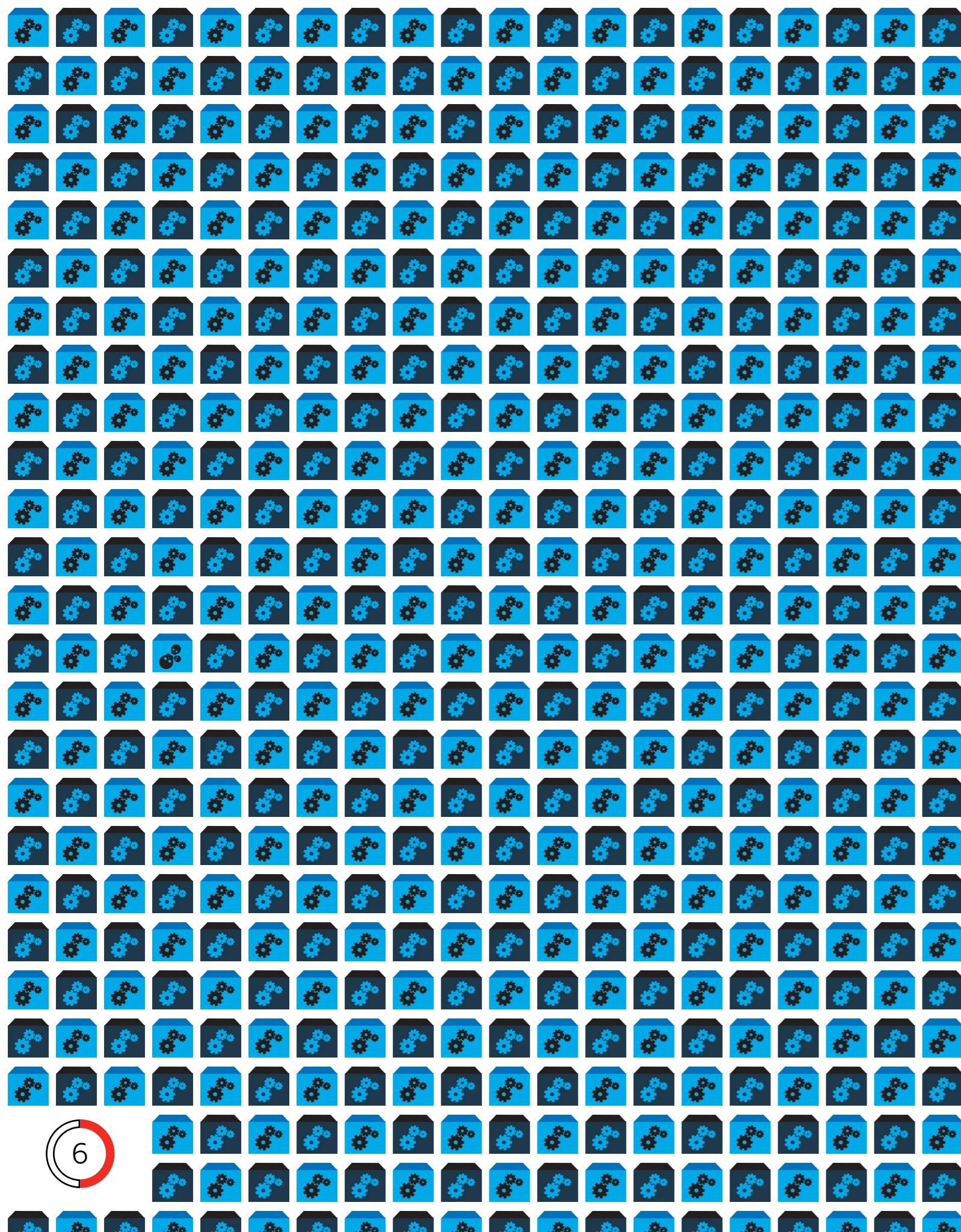


According to analysts at IDC, containers can accommodate as many as **four to six times the number of application instances** as a hypervisor running on the same server.

As some experts explain, this just compounds the sprawl problem.

“Consider what happens to these issues when containers enter into the mix. Not only are all the VM issues still there, but they’re now potentially compounded. Inventories that were already difficult to keep current because of VM sprawl might now have to accommodate containers, too. For example, any given VM could contain potentially dozens of individual containers. Issues arising from unexpected migration of VM images might be made significantly worse when the containers running on them can be relocated with a few keystrokes.”

– Ed Moyle, director of emerging business and technology for ISACA



Docker downside

Clearly, the more that container sprawl takes hold of IT environments, the bigger the monitoring black hole organizations will start to face. If Docker users fail to address this issue, they'll find themselves engaged in a game of diminishing returns with containers. Most organizations bring on Docker to aid efforts in continuous delivery and to ultimately achieve better application performance and customer experience. But a lack of effective container monitoring may actually produce the opposite end results.

“Dockerized” applications are portable, can be run anywhere and are an ideal building block for development using the microservices approach. But they also introduce another level of complexity to the application architecture. Now organizations must be able to not only track application performance metrics, but also measurables around container performance, as well as analytics around how business transactions behave as they move across multiple Dockerized applications.

What you lose without container monitoring

Without an effective means of monitoring containers, Docker users stand to lose ground in three important areas:



PERFORMANCE

An inability to inspect and analyze metrics about the Docker container itself and the applications contained within makes it next to impossible to manage performance within a microservices architecture. Developers will lack visibility into Dockerized application code to deal with latency issues and ops will be unable to isolate infrastructure problems that could be pegged through CPU, memory and network data.



CONTEXTUAL KNOWLEDGE

Docker itself has already gone a long way toward improving monitoring by offering up a feed of key performance metrics about its containers for use through its Docker Monitoring Extension. But if organizations can't find a way to link that to what's happening in applications within containers and across the entire application infrastructure, then they lose context necessary to find and fix systemic problems.



CUSTOMER SATISFACTION

Without the appropriate tools to deal with the previous two areas, organizations will be ill-equipped to identify troubles in applications before they impact customers. In this day and age where end users have endless choices of technology, they've grown to be unforgiving when it comes to software and digital services.

What to look for in a monitoring solution

Ultimately, application and infrastructure monitoring should be able to fold Dockerized applications into an end-to-end view of the architecture that's easy to analyze.

“Since containers may rapidly spin up and down in automated deployment and scaling scenarios, the new challenge for the monitoring agent is to keep track of and react to these lifecycle events without introducing complexity for the user,” explains Peter Salvatore, a member of the technical alliances team at Docker.

This means that organizations need to look for a monitoring solution that can trace interactions from the user through all of the application service calls and infrastructure, including processes running inside containers.

Ultimately, a monitoring platform should be able to treat Docker containers just as any other application or infrastructure component in the continuous delivery tool chain.

A good solution should be able to give insight into:

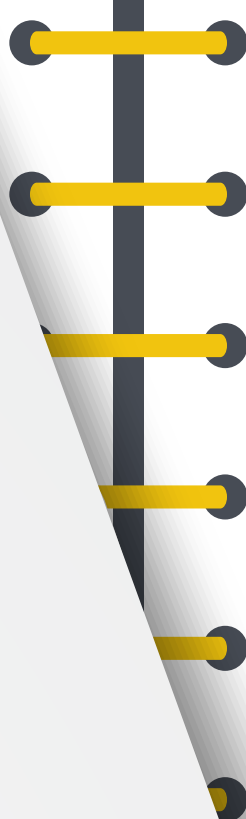
How are applications connected to the container?

How do container metrics relate to application workloads?

What are the dependencies between Dockerized applications and the rest of the infrastructure?

Rules for container and microservices monitoring

As organizations evaluate technology and processes to decide how they'll monitor containers, they should keep some goals in mind. According to Adrian Cockcroft, the ultimate goal of container and microservices monitoring is to be able to “understand and communicate common microservice failure patterns.” Known best for his successes as the cloud architect for Netflix, Cockcroft now works as a technology fellow for Battery Ventures. In a recent presentation, he brought forward six rules to live by for monitoring containers and microservices:



RULE #1: Spend more time working on code that analyzes the meaning of metrics than code that collects, moves, stores and displays metrics.

RULE #2: Metric to display latency needs to be less than human attention span (~10s).

RULE #3: Validate that your measurement system has enough accuracy and precision. Collect histograms of response time.

RULE #4: Monitoring systems need to be more available and scalable than the systems being monitored.

RULE #5: Optimize for distributed, ephemeral, cloud native, containerized microservices.

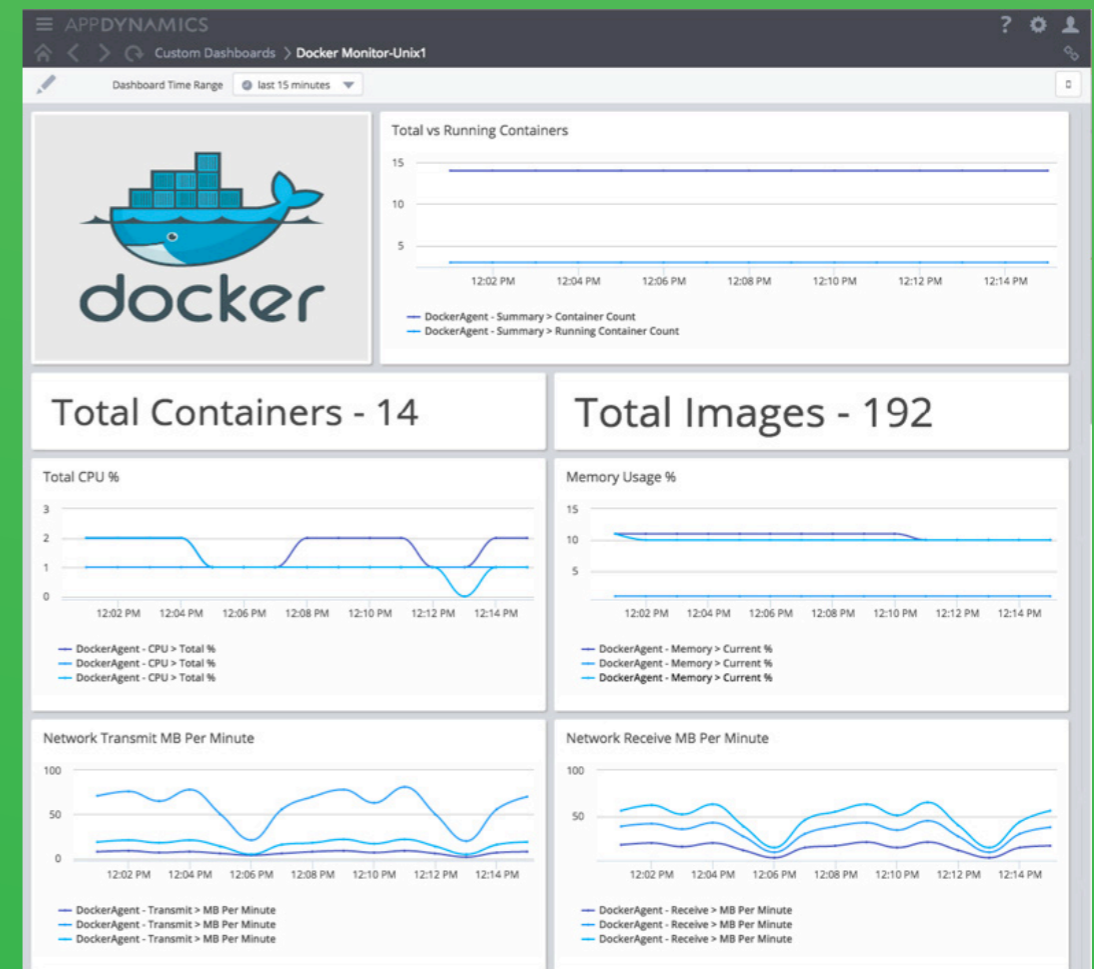
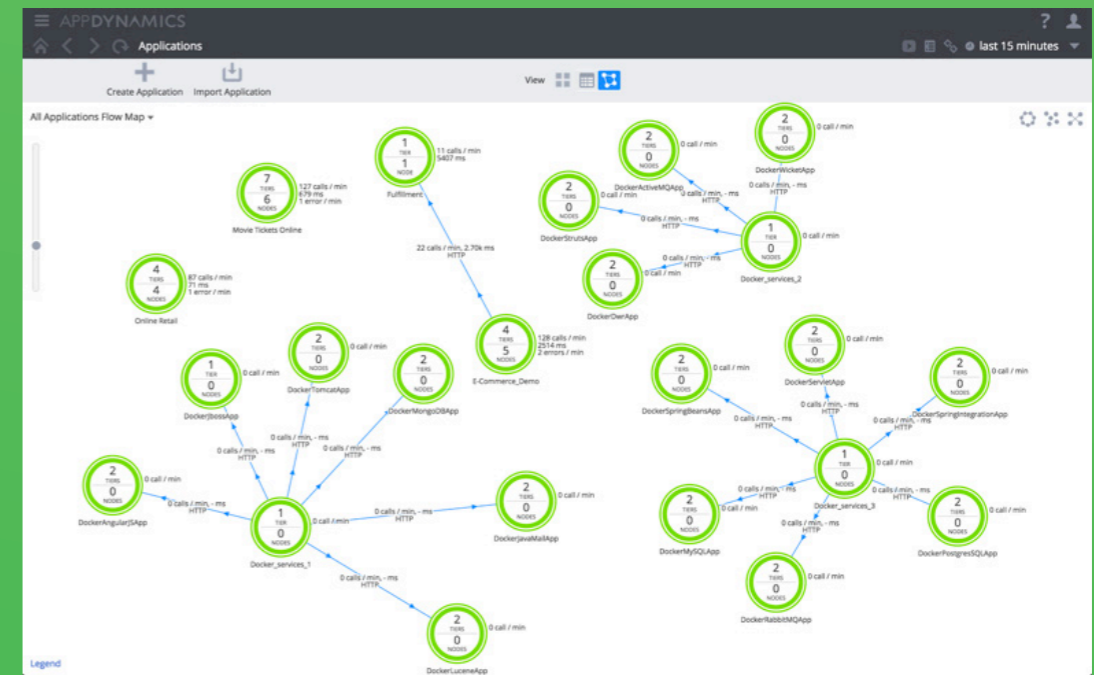
RULE #6: Fit metrics to models to understand relationships.

AppDynamics working with Docker

The AppDynamics Docker Monitoring Extension builds off of the momentum App Dynamics has already built through its unified monitoring approach. Working through the Docker Remote API, the AppDynamics Docker Monitoring Extension offers visibility into important Docker container metrics and delivers them through an out-of-the-box dashboard:

- Total number of containers
- CPU usage
- Memory usage
- Network traffic

More importantly, these metrics are correlated with the metrics from the applications running in the container to get a view of overall performance metrics as compared to Docker performance metrics. When paired with the AppDynamics core functionalities such as dynamic baselining, health rules, policies and actions, the platform delivers a cohesive view that fully accounts for the newly Dockerized world.



APPDYNAMICS

Interested in monitoring your Docker containers with AppDynamics?
Check out a free trial today at appdynamics.com/freetrial.