# DO CONTAINERS BELONG IN THE CMDB?

Containers have been a white-hot topic in software and infrastructure recently. Gartner has predicted that more than half of global enterprises will be running containers in production environments by 2020; it pegged adoption at less than 20% in 2017. Meanwhile, 451 Research expects the application container market to haul in nearly $2.7 billion in revenues in 2020, up from $762 million in 2016.

Container adoption is increasing because the technology is well-suited for our multi-cloud world. Containers make your workloads more dynamic and are often noted for the portability they enable across different environments. They also make these workloads ephemeral: You can spin up a new instance quickly, use it for as long as you need, and then effectively throw it out. Wash, rinse, repeat.

These dynamic, ephemeral workloads also pose challenges for IT teams, particularly around discovery and service management. Tracking each one of these containerized workloads could quickly become unmanageable. On the flip side, tracking nothing is a recipe for failure; the benefits of containers will get wiped out if you're running them in the dark.

## The fast food of compute

How do you ensure you're maintaining full visibility in your environments in an efficient, scalable manner? What should you be populating and maintaining in your CMDB?

Let's consider an analogy that might help: Think of containers as the technology equivalent of fast food. You order it, the order gets filled almost immediately, you consume it, and you move on. It's fast, it fills a need, and you simply dispose of everything when you're done. There's no cleanup, no dishes, no maintenance required. And it's pretty much always available when you want it again.

The analogy extends to IT service management. One reason why people eat fast food is that they know pretty much exactly what they're going to get when the order arrives.

That's not because they're keeping a record of every cheeseburger they've ever eaten, though. Rather, they remember where and how they ordered it. They remember that they like how one chain prepares and delivers its burger more than another, and they like that it's consistent. A Big Mac is a Big Mac, regardless of when and where you order one.

That's how you should approach containers: Don't try to track every single instance you spin up and down. Instead, track its configuration—how the burger was prepared, so to speak.

## Focus on the details

Instead of the bun, condiments, and cook time, substitute the build, the service this workload is a part of, and the deployment tool used. And track all of this as it changes, too.

Don't think about the container itself. The intricacies and dependencies of the configuration that governs that container are what you want flowing into your CMDB, so that you can track them in their present state as well as when they change.

The dynamic, ephemeral, and scalable nature of containers is part of their growing appeal to enterprise IT. Just remember that you want to track how the burger gets to the table—not the burger itself, or the packaging that gets tossed when it's gone.