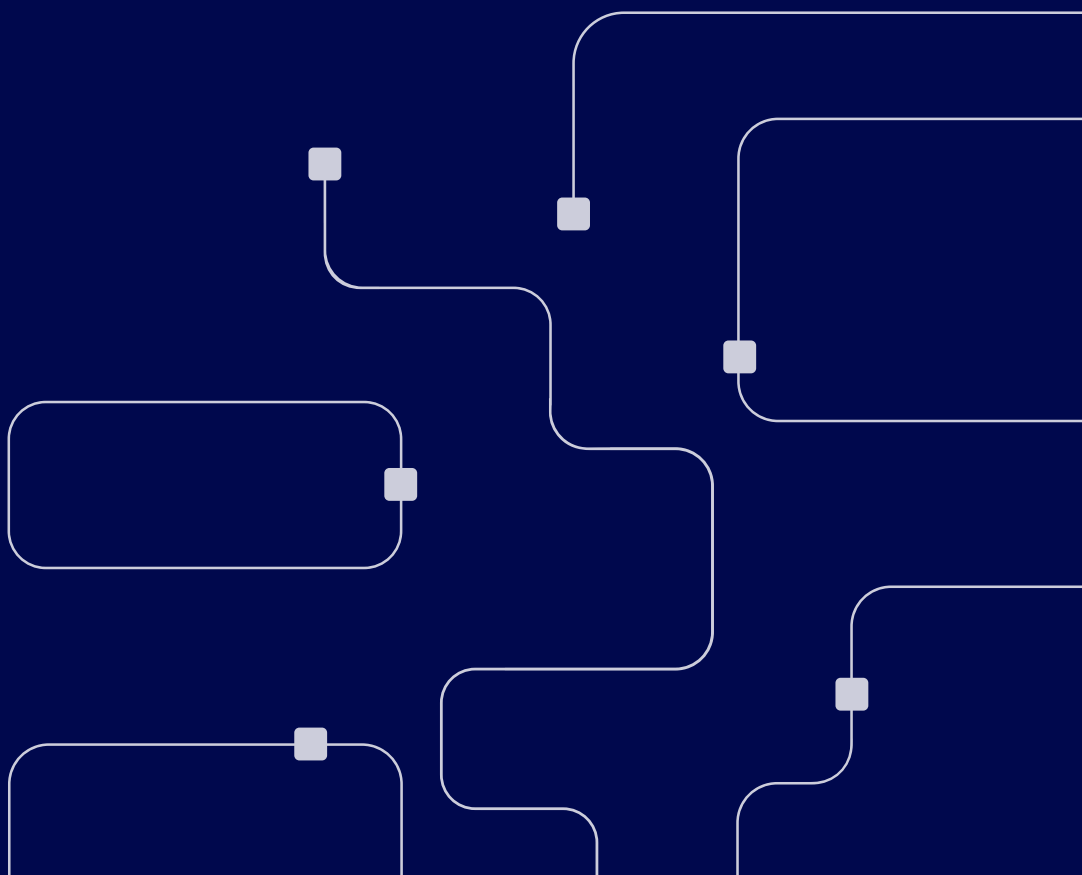
docker

Docker's container development strategy Local + Cloud

Docker's chief product officer discusses its local + cloud strategy to container development.



ADMIN Network & Security

SPECIAL docker

Docker's container development strategy

Local + Cloud

Docker's chief product officer discusses its local + cloud strategy to container development. By Amy Pettle

At DockerCon 2023, Docker announced a local + cloud strategy for container development. We talk to Giri Sreenivas, Chief Product Officer at Docker, about Docker's approach to container development. Giri highlights three new local + cloud products and hints at where this strategy will lead Docker in the future.

The container industry has evolved dramatically over the past 10 years since Docker first launched. How have these changes directed Docker's product development?

Docker's priority has always been the developer. Our mission is to empower developers to focus on building the future. While the container landscape has seen some significant changes in the past decade, this evolution hasn't fundamentally changed our core objective. Instead, it has served as a north star, shaping how we address the evolving needs of the developer community.

Emerging technologies, evolving regulations, and shifting market dynamics all influence the daily lives of developers. We actively listen to how these all play out in the market, which helps us anticipate developers' needs, so we can continue delivering solutions that make their lives easier.

There are some trends I want to touch on that have really helped Docker focus our product development efforts. I think we're all aware of the major architecture shifts from monolithic applications to microservices as well as the rise of Kubernetes and container orchestration. But what I want to highlight are other areas that are very close to Docker: cloud-native development, security and the supply chain, and of course, the developer experience.

We want to meet our customers and users where they are in their development journey. So we focus on building tools that integrate seamlessly with cloud-native services and deployment workflows. Docker has been fundamental to the rise of the cloud-native industry and we'll remain a critical player by investing in tools that give developers flexibility to thrive in any work environment.

Similarly, container environments demand robust security measures, especially as these environments become more complex. Again, we want to meet our users where they are instead of imposing another cumbersome tool on them. So we ask ourselves "how can we make security as easy as possible for developers?" Our tools are designed to catch and solve security issues seamlessly as part of the development process, reducing interruptions and rework.

A question we see a lot from customers is how to help their teams innovate and keep up productivity. We'd like to think we've mastered that at Docker. It's all about a seamless and efficient developer experience. Managers need to reframe how they're thinking about this. It's not only "how do I make developers faster," but also "how can I make my developers happier." Docker prioritizes a seamless and efficient developer experience to empower our users. We believe that happy developers are productive developers, which is why we focus on making our tools intuitive, efficient, and magical.

AI is another great, and relevant, example of this. I'm sure we all see the momentum surrounding generative AI and its potential for developers. But how does this relate specifically to Docker's users? We're seeing

Giri Sreenivas

Giri Sreenivas is the Chief Product Officer at Docker where he leads product, design, growth and data. He has over two decades of experience in engineering and product leadership roles across consumer and enterprise businesses. He is also a two-time venture-backed startup founder with an acquisition and subsequent IPO (Rapid7). Giri is a graduate of the schools of computer science at Stanford University (BS) and Carnegie Mellon University (MS) and currently resides in the Greater Seattle area.

containers used to simplify configuration and development of generative AI solutions. That's why we worked with some partners on our Gen AI stack. It's an easy way to get developers up and running, fast.

Tell us about Docker's local + cloud approach and how it benefits container developers.

We are advocating for a local + cloud approach because we recognize that a single, universal methodology doesn't work for every organization. Developers have a diverse set of needs and should have the flexibility to leverage both local and cloud resources based on their specific requirements.
Instead of forcing developers to adapt to a "cloud-only" or "local-only" environment, Docker embraces a hybrid approach. This allows developers to utilize the familiarity and comfort of local development tools for tasks like code editing and debugging while seamlessly scaling to cloud resources when needed for resource-intensive workloads, collaboration, or deployments.
Our goal is to meet developers where they are with "just enough" cloud to overcome any limitations their current environments pose.

Docker recently announced Docker Scout General Availability (GA). What does Docker Scout do for software supply chain management?

Docker Scout goes beyond just being a security or vulnerability detection tool. It empowers developers to build secure software by ensuring its quality, trustworthiness, and compliance right from the outset.
Imagine building a house. You wouldn't wait until it's finished to check for faulty wiring or leaky pipes, right? That's where Docker Scout comes in for your software development process.
Think of Docker Scout as your construction inspector for software. It proactively helps you identify and fix potential issues in your code, dependencies, and configurations, all while you're still building. This way, you avoid costly rework and delays down the road. Ultimately, Docker Scout helps you build better, more secure software.

What benefits does Docker Scout offer development teams over previous supply chain management solutions?

What differentiates Docker Scout is its ability to meet developers where they are in their workflows. We understand the developer's pain when it comes to solving for security, and we want to make the process as easy and painless for them as possible. We integrate directly into their workflows, so when they're building, they can view where there are issues and solve them right then and there.

Tell us about Docker Build Cloud and its ability to harness the cloud.

Docker Build Cloud is a game changer for developers and development teams who are tired of waiting around for slow builds. It addresses a major pain point by offloading resource-intensive build tasks in the inner loop as well as continuous integration (CI) to the cloud, freeing up local resources and significantly speeding up the build process. We've seen build times reduced by up to 39 times in some cases.
It leverages the scalability and elasticity of the cloud to provide developers with on-demand access to powerful hardware resources. This allows them to build their images quickly and efficiently, even if they don't have access to a high-performance machine locally. Additionally, the cloud-based nature of Docker Build Cloud makes it cost-effective, as developers only pay for the resources they use.
One of the best things about Docker Build Cloud, similar to Docker Scout, is its seamless integration with existing workflows. Developers can use their existing tools and keep their current workflows without any major changes. This makes it easy to adopt and start benefiting from faster builds immediately.

How does Docker Build Cloud differ from alternative build options currently available?

Docker Build Cloud focuses on the crucial inner loop builds where developers need immediate feedback on code changes. This is a significant improvement over traditional build environments that often run on a local developer machine or through emulation (virtual machines), leading to slowness and inconsistency due to varying hardware configurations and multiple layers of touchpoints. Docker Build Cloud avoids these limitations by utilizing powerful cloud instances. This delivers a significantly faster and more consistent build experience.
Docker Build Cloud also integrates seamlessly with existing tools and Docker commands without requiring workflow changes, differing from other CI solutions that often force frequent merges, new command-line interface tools, or unnecessary production-like requirements.

Docker also announced a beta version of a new debugging tool, Docker Debug. How does Docker Debug work?

Docker Debug is really exciting because it enables debugging in containers that don't have an available shell, which is becoming increasingly common with the adoption of slim images. It also enables debugging when `docker exec` is not possible because a container is not running. It removes the friction associated with container debugging, empowering developers to fix issues within their domain of expertise without worrying about container intricacies.

How does Docker Debug differ from traditional container debugging and how does that benefit container developers?

With Docker Debug, anyone, whether a novice or seasoned Docker user, can debug their containers without needing full knowledge of how that container was built and the tools it contains. There's no need to specify a particular configuration or load debug-specific tools into the image. It's native to the Docker toolchain as well so there's no need for the developer to install any additional local tooling.

What other improvements has Docker recently introduced to improve the inner loop (code/build/test/debug) experience for container developers?

We've focused a lot this year on enhancing the inner loop and local development experience, starting with making it easier for developers and development teams to ship secure software with Docker Scout. Through each stage of the inner loop, Scout surfaces vulnerability risk and simplifies how to fix it with simple, one-step remediations.
To improve the experience with containers throughout the entire inner loop, we have also made tremendous advances in performance and stability with Docker Desktop. There are dramatic improvements in startup times, reductions in memory and CPU consumption and improvements, and speedups in networking and file sharing performance.
In addition to Docker Desktop's performance improvements, we have listened closely to our customers and delivered a number of improvements for the deployment and administration of Docker Desktop so it's more readily available and up to date for developers from their IT teams.
We're excited about the direction of `docker init` as we add support for more languages like Python and PHP and accelerate getting more projects running in containers in the code and build phases of the inner loop.
To improve visibility and troubleshooting of container builds, we recently released a Builds view in Docker Desktop.
Lastly, there are some great improvements to Docker Compose with the additions of the `watch` and `include` features. With `watch`, we have made Compose a lot more effective and efficient for front-end developers that are making rapid changes and want to see those reflected quickly in the inner loop. With `include`, we're simplifying the code and build steps of the inner loop by making the Compose configuration more closely reflect the modularization of the application itself.
2023 has been a tremendous year of innovation across Docker with a series of improvements across the inner loop that are benefiting developers.

What is on the horizon in container development tools for Docker?

As we hinted at DockerCon, we believe the future of development is very much a local + cloud hybrid one. This is both about bringing the power of the cloud to the local development experience as well as making it easier to leverage dependencies on the cloud in the local development experience. This will be a common theme across our products in the coming year.
Scout will innovate on the best developer experience for secure software development, leverage trusted content to reduce vulnerability exposure in more strategic ways, and support more integrations to improve end-to-end visibility of risk and the value of remediation across the Software Development Life Cycle (SDLC). We'll further invest in enabling customers to customize policies, remediations, and more, as well as continue improving suggestions for remediations to be more actionable.
Giving time back to developers and development teams to focus on the things that matter is at the heart of our mission, and we're excited about how accelerating builds with the power of the cloud in Docker Build Cloud delivers on this. This fits squarely into our local + cloud theme by bringing the power of the cloud to existing build workflows in the inner loop and CI.
We will continue to invest in simplifying and accelerating the process of containerizing applications with further investments in `docker init`, Docker for AI, and similar initiatives that either automate or abstract the complexity of creating Dockerfiles and Compose files.
Modern software development is a team sport, and it is becoming increasingly vital to facilitate collaboration with containers. Developers today stitch together a variety of hacks and tools to solve the networking and ephemeral environments problems that arise from wanting to share and debug with peers. Stay tuned for some timely innovation here that I'm excited for Docker to get to market in 2024.
We're also excited about our entry into testing in the inner loop with our recent acquisition of AtomicJar. The AtomicJar team has done a phenomenal job cultivating a strong community around Testcontainers and an ecosystem of partners. We will continue to invest here with an eye toward the local + cloud answer for increasingly complex testing needs of developers.
And of course, we can't talk about the future without addressing the watershed moment our industry is going through with generative AI. In addition to all the ways one would expect Docker to leverage AI to accelerate containerization, enable adoption of GenAI stacks, and improve the inner loop experience with containers, we are also thinking deeply about how the SDLC and developer journey will be transformed with AI and how Docker will help accelerate this transformation. ■

The Author
Amy Pettle is an editor for *ADMIN* and *Linux Magazine*. She started out in tech publishing with *C/C++ Users Journal* over 20 years ago and has worked on various Linux New Media publications.