anchore

INTRODUCING ANCHORE:

Technology Foundation

Container technology has achieved an incredible amount of momentum over the past couple of years.

There are many reasons why, though it is clear that several characteristics of foundational container technology are immediately attractive: Ease of use, the promise of a substantial decrease in time between development and execution of complex application systems, and an impedance match between modern software engineering processes and the deployment speed and agility associated with container systems.

Surrounding technology, platforms, and models have arisen to provide the community of container users with a rich variety of choice when it comes to deployment platforms and runtime environments, while the ecosystem of publicly available software distributed as containers continues to explode.

Today, adoption of container-based approaches has resulted in dev/ops/sec teams achieving a “time-from-idea-to-production” engineering model that is reduced by weeks or months over previous methods. With a relatively new model (though container technology has been with us for a while, the end-to-end model has been propelled in recent years), adopters of container-based systems are faced with a great deal of choice. They’re also faced with the problem of designing custom mechanisms and processes for integrating containers into existing environments, models, and systems, even as the tools surrounding core container tech continue to evolve rapidly.

CONTAINER DEPLOYMENT CHALLENGES

Tensions have arisen between the benefits of ultra-fast development and the need to effectively manage production application workloads at speed. At the same time, the developer experience that has driven the adoption of container technology must be balanced against the need for organizations to control, evaluate, and secure running production systems.

And as with any new tech, there is also a natural tension between the need for a somewhat stable, reliable system for running production applications and the willingness to adopt new technology as it evolves. We've categorized these tension points, among others, as the most critical areas we intend to target:

TENSION POINTS

1. Flexibility vs Stability: Containers offer developers incredible flexibility when it comes to assembling software in support of applications, while production deployment environments often require more software stability to understand and control what is running.

2. Opacity vs Transparency: Containers explicitly define an executable "unit" composed of a collection of software (in-house and external), which is somewhat opaque. Where operational functions often benefit from more transparency to validate and troubleshoot runtime events.

3. Constant Change vs Known State: The community at large is providing highly functional software as containers at whatever pace is appropriate for their particular software. Where production systems often benefit from a known "state" of running software to ensure an appropriate level of validation, security, and understanding of provenance.

4. Customization vs Standardization: Containers offer application development teams unparalleled ability to choose exactly which environment (third-party software dependencies, OS flavors, and entire third-party services) to base an application implementation upon, leading to a wide variety of actual software that is running at any point in time when many dev/application/systems are running at once. Production systems, however, often strive for more unification of platforms and software versions offered as application development starting points.

5. Distributed vs Centralized: Application development teams are benefiting from not having to stay closely coordinated when it comes to software environments using containers. Where production systems can benefit from the ability to understand the provenance of running applications from a "baseline" to the "final" running system.

Anchore is developing technology and services that aim to address these tensions in a fashion that is non-disruptive for development, operations, and security teams. We developers must be allowed to continue using container technology in a relatively uncoordinated manner (in order to gain the ability to choose the right software environment for their particular app), while at the same time allowing ops to choose and operate the right container deployment system for their use case. Anchore systems are used to create and manage a known, transparent starting point for dev/ops containers, while fully exposing the characteristics of a running container system that are vital for security teams to understand and validate. To achieve this, we have designed a system that is composed of the following components:

ANCHORE TECHNOLOGY

Anchore back-end container validation engine

Anchore front-end user container registry and environment customization UI

Anchore on-premise container registry creation and synchronization tools

Anchore on-premise container inspection, provenance and introspection tools

Anchore report generation a nd notification system

TECHNOLOGY BENEFITS

In combination, users who integrate Anchore technology into their environment can realize the following benefits, which are each targeted at the previously enumerated tension points:

- By using Anchore container registries, users are assured that their organization can start with a fully validated, curated, and comprehensive set of containers that are either directly imported from popular public container hubs, generated by Anchore, or imported from the users themselves.

- Through the Anchore front-end customization UI, users can instruct Anchore to generate the best set of containers for their specific deployment environment, at any point in time. There is no fixed container set, but rather the set is calculated based on the latest stable container set at any point in time, against the user's specified environment(s).

- By interacting with the on-premises container inspection and provenance tools, users can associate running containers with the "baseline" container from which the running system was derived, and can view any deltas for troubleshooting or security analysis.

- Through the report generation and notification system, Anchore can generate a full report of exactly what has been executed against baseline and running containers, and can further proactively notify users when security or functional issues are discovered in the baseline (and, using the provenance tracking, can associate notifications with actual running containers which have been based on known baseline containers).

MOVING FORWARD

Container adoption is strong, with clear benefits across the board for those who have chosen to base their next-generation technology model on the rich ecosystem of tools and techniques in the container space. Natural tensions exist between existing practices and those necessary to fully realize the power of container technology.

Anchore’s goal is to remove barriers and accelerate the ability of organizations to realize container-based infrastructure. We’ll do that by providing a trusted source of containers, a rich customization layer to allow adaptation as surround container systems continue to evolve, and power tooling to enable a high degree of transparency and provenance tracking for running container-based environments.

With Anchore, our users will be able to start with a solid foundation, without re-platforming, and gain the confidence required to achieve the same degree of control and visibility of existing technology. All without sacrificing the benefits that container technology provides.

anchore