armory

TERRAFORMING WITH SPINNAKER

ARMORY SPINNAKER INTEGRATION

Spinnaker and Terraform are leading open source projects enabling the continuous software delivery and infrastructure-as-code revolutions, respectively. Recently, the engineers at Armory released a Spinnaker integration with Terraform, enabling DevOps teams to extend the power of software pipelines and continuous delivery to infrastructure management. Click here for a demo and webinar on the integration and read on to learn more.

1 Introduction to Terraform

What is Terraform?

Terraform is an open source platform that enables users to safely and predictably create, change, and manage infrastructure. It allows users to codify APIs into declarative configuration files that can be shared among team members and treated as code that can be edited, reviewed, and versioned. Developed by Hashicorp, Terraform integrates with multiple public cloud providers like AWS, Google Cloud, and Microsoft Azure, internal clouds, or hybrid cloud environments for infrastructure delivery and management.

How it Works

Anything with an API can be managed with Terraform. Commands for infrastructure resources are centrally automated from a set of modules, instead of being handled manually from the APIs or consoles of each individual cloud or infrastructure provider. This enables easy standardization of infrastructure management. For example, it is easy to automate management of infrastructure resources like servers, VMs, containers, or load balancers with commands like “build,” “manage,” “delete,” or “update.”

Within Terraform, users define the infrastructure they want to create by issuing a series of commands to create a “plan” for their desired infrastructure. Terraform keeps track of both the planned (desired) infrastructure and the current infrastructure state in a state file. When you issue a Terraform plan, or something changes in your live infrastructure, Terraform compares the state file to the new desired state. It determines what needs to happen and which API needs to be called upon in order to converge upon the desired or planned state and automates the commands to achieve that state.

Benefits of Using Terraform

- Infrastructure code can be used repeatedly and efficiently, in a pattern across the organization.
- Reducing the number of manual processes reduces the likelihood of human error.
- Terraform commands are written in human-readable, declarative language, making it easy for teams to understand and agree on approaches and best practices.
- Infrastructure can be put through the same code review process as application code.
- Enables easy debugging and rollbacks.
- Introduces safety into rollouts with execution windows.
- Quantifies the state of your infrastructure.

2 Infrastructure as Code: Overview and Benefits

What Is Infrastructure as Code (IAC)?

More companies are adopting continuous integration and continuous delivery (CI/CD) to address the complexities of delivering software in hybrid environments with multiple deployment targets. The concept of "infrastructure as code" (IAC) is a corresponding shift in the approach "to managing IT infrastructure for the age of cloud, microservices, and continuous delivery," where the systems and devices that comprise infrastructure are approached and treated as if they themselves were software. This enables organizations to automate the process of provisioning infrastructure using tools and approaches that have been proven to work in other areas of IT and software delivery. Examples include testing and version-controlling infrastructure code as well as checking it in alongside the application code it is supporting.

Benefits of IAC

- Prevents configuration drift by ensuring all deployments are standardized.
- Enables automation of infrastructure deployment in a repeatable, consistent manner.
- Quick, consistent configuration and deployment of infrastructure components.
- Rapid and repeatable configuration of development, staging, and production environments.
- Automated configurations reduce the likelihood of errors in infrastructure and application deployments.
- Mitigates risk. Automation and consistency protect organizations against loss of institutional knowledge when key team members leave, and repeatability is ensured through version control and testing.
- Secures efficiency and cost savings by enabling faster time-to-market.
- Ensures accuracy and repeatability by applying the same testing stringency to infrastructure code as to application code.
- Infrastructure documentation will always be current: documentation doesn't have to be manually recorded when the code itself is the documentation.
- Enables collaboration across disparate teams.

3 Terraform and Spinnaker: The Combined Benefits of Infrastructure as Code and Continuous Delivery

4 Ways to Think About Infrastructure and Software Delivery

Continuous integration and continuous delivery (CI/CD) are an approach to software delivery with a set of best practices for releasing software with higher velocity and confidence. To understand where Terraform fits into CD, it's important to understand the nuances of infrastructure within the context of CD — specifically static, dynamic, shared, and single-user infrastructure.

STATIC INFRASTRUCTURE

- Changes infrequently: A few times a month to every few months.
- Stable resources: Provision once, reconfigure occasionally. It remains largely unchanged.
- Static infrastructure is consumed by dynamic infrastructure.
- *Examples: Caches, databases.*

SHARED INFRASTRUCTURE

- Tends to have high user base spread across an organization.
- Customers of shared infrastructure are frequently application developers building applications that actual customers are using.
- Shared infrastructure is critical: If developers can't access or use it, they can't make changes to the applications or single-user infrastructure they need.
- *Examples: Kubernetes, elastic search, logging or monitoring tools.*

DYNAMIC INFRASTRUCTURE

- Changes constantly: Can change multiple times per day, especially when organizations practice CD.
- Dynamic infrastructure tends to be rolled out across multiple environments and is usually staged and orchestrated.
- Dynamic infrastructure consumes static infrastructure.
- *Examples: Infrastructure for hosting applications; read/write objects from/to databases.*

SINGLE-USER INFRASTRUCTURE

- Tends to be used by a single application, user, or team.
- Outages or mistakes are quickly repairable.
- Single-user infrastructure consumes both dynamic and static infrastructure.
- *Examples: Caches, databases, buckets.*

Consistent Delivery Requirements No Matter the Infrastructure Type

Each of the infrastructure types varies in the type of user or application that accesses it, what it supports, and how often it changes. However, there are almost identical delivery requirements for all infrastructure types in any organization: safety, consistency, and reliability.

Before continuous delivery, infrastructure and application updates were rolled out in "time windows" when users were ideally asleep or were not likely to be using the resource or application. For example, applications might be taken offline between 10pm and 2am for updating. As expectations increase for always-on infrastructure and applications, there is a desire to avoid having update windows with any associated downtime. To achieve this goal, companies are turning to continuous delivery so that updates can be rolled out with as much automation and safety as possible, without needing specific update or rollout windows. Code updates can be delivered as they are ready, not when users are less likely to be consuming the application.

Why Do Organizations Turn to CD to Roll Out Applications and Infrastructure?

With CD, changes are always rolled out the same way, and increased automation means fewer errors as the "surface" for errors is reduced. Rollout windows with application or infrastructure downtime are replaced with constantly updated code. CD also represents a "paved road" for infrastructure: a philosophy originating from Netflix (where Spinnaker also originated), providing a set of applications and tools that make it easy to launch a new application using well-established patterns. This is similar to the guard rails that Spinnaker provides, where developers can quickly code and innovate without breaking organizational best practices, controls, or restrictions.

Terraform + Spinnaker (1+1 = 3)

Organizations that use both Terraform and Spinnaker experience more than just the combined benefits of both. Running Terraform infrastructure code through the same code review process as application code review ensures a solid vetting and successful delivery, as infrastructure is being checked for correctness and completeness at every step along the way.

Additionally, deploying infrastructure continuously alongside associated applications enables organizations to understand how they affect each other in a "band of context." For example, in an organization that does not practice integrated DevOps, an applications engineer would deploy an application while another team member deployed the related infrastructure. Neither would have visibility into each other's code unless (or until) something went wrong. With Spinnaker and Terraform, a DevOps practitioner rolls out the application and infrastructure side-by-side, understanding changes happening to both and how they might affect each other. By maintaining the band of context, there is a higher likelihood of smooth deployments and rapid corrections in the case of issues.

The result? More safety, reduced confusion, and increased delivery velocity with a tighter feedback loop that allows for small, iterative changes.

4 Armory Spinnaker/Terraform Integration: Details

Armory built its Armory Spinnaker/Terraform integration to solve a clear customer need: the ability to roll out key and spare infrastructure alongside applications, with the safety and reliability provided by Spinnaker.

What Does the Integration Enable Today?

- Manage pipelines with Terraform and update them with Spinnaker, within the same pipeline manager.
- Provision a full environment with Terraform.
- Deliver infrastructure as code with Terraform and deploy applications alongside it, within the same Spinnaker pipelines.
- Teams who have onboarded Spinnaker can create Terraform infrastructure pipelines exactly the same way they create application pipelines. There is a low barrier to adoption and a short, simple learning curve.
- Teams can use the same Spinnaker artifacts and repositories.
- Parameterize your pipelines (e.g., RBAC).
- Orchestrate testing (via Jenkins stages).
- Visualize your infrastructure rollout and understand the state of your infrastructure at-a-glance.
- Manage execution windows within Spinnaker pipelines so that preset changes are paused until execution windows are reached.
- Maintain the infrastructure/application "band of context."
- Provision, test, and revise infrastructure in all stages: staging, testing, and production.
- Establish guardrails for both application code and infrastructure management with Spinnaker.

"If Terraform supports it, our Armory Spinnaker integration supports it as well."

Integration Roadmap Items

Armory's goal is to empower more complex scenarios as it continues to support and improve the current integration. These are just a few of the items in the integration roadmap:

1) Capture Terraform outputs and use them in downstream stages
    - Create an output and inject it into something else. (Example: create a database and host it.)
2) Apply a plan from an upstream stage
    - Save a plan from a previous stage, and apply only that.
3) Integrate with cloud driver accounts
    - Configure a cloud driver in Spinnaker to integrate with AWS, GCP, etc. Pull credentials to communicate with the APIs and make it easier to set up credentials for Terraform.
4) Build backwards compatibility between Armory's Terraform version and others
    - Select which version of Terraform you want to run. This will make the integration more adaptable, so your team doesn't have to upgrade to match the Terraform version available on the Armory integration.

More details and a demo of the integration can be found here.

5 Why Armory?

Armory is a founding member of the Continuous Delivery Foundation and a noted thought leader in CI/CD, building Spinnaker integrations into other products, and contributing back to the Spinnaker open source community.

The Armory platform is Spinnaker at enterprise scale. We provide compelling features and extensions including:

- Terraform and other third-party integrations
- DinghyTM for managing Spinnaker pipelines via source code
- SDLC Policy Engine for improved compliance, security, and end-to-end auditability
- Armory Halyard for improved installation, release, change, and configuration management
- Increased availability and guaranteed per-app uptime SLAs
- Reporting for actionable insights and deep analytics

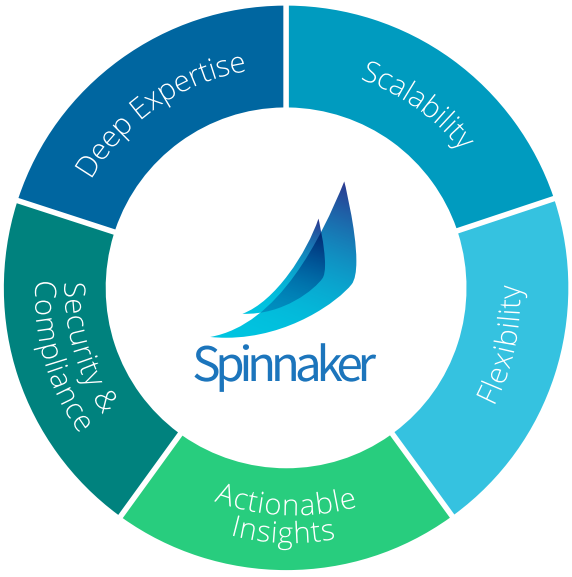
In addition, Armory provides deep domain expertise and 24/7 service and support to our customers. Our services extend from installation to operationalization to training and ongoing access to our solutions architects and Spinnaker experts. We also provide customers the option of a managed Spinnaker instance run in their data center.

DEEP EXPERTISE

- Operating and Extending Spinnaker
- Services, Training, and Support
- Guaranteed Uptime SLA

SECURITY & COMPLIANCE

- End-to-End Auditability
- Air-Gapped Environments
- SLA Management
- FedRAMP & ISO



ACTIONABLE INSIGHTS

- Development Analytics
- Developer Productivity

SCALABILITY

- Manage Spinnaker Pipelines via SourceCode
- 1-Click Service Bootstrap
- High-Availability Configuration

FLEXIBILITY

- 3rd Party & Custom Integrations

Jira Software

DATADOG

Terraform

New Relic.

Vault

Contact Armory today for a complimentary assessment of your software delivery practices and learn more about how your organization can benefit from fast software delivery.

CONTACT US TO LEARN MORE

Armory Is Spinnaker at Enterprise Scale

Armory helps software teams ship better software, faster. Our platform is powered by Spinnaker, the de facto standard in cloud-native continuous software delivery developed and open sourced by Netflix and Google. Armory brings the power of Spinnaker to your business, coupled with enterprise-grade availability, powerful feature extensions, and 24x7 expert support and services.

Armory is headquartered in San Mateo, CA, and is backed by leading investors including Insight Partners, Crosslink Capital, Bain Capital Ventures, YCombinator, and others.

armory

Armory Inc. 100 S. Ellsworth Ave., 9th floor, San Mateo, CA 94401
1-888-222-3370 | info@armory.io | www.armory.io