HashiCorp

GitHub | CUSTOMER STORY

Cracking the code to global success

The world's largest developer collaboration platform uses HashiCorp solutions to shore up internal processes and deliver mission-critical functionality faster and at lower cost.

// Infrastructure Enables Innovation

About GitHub

GitHub is the developer company. As the home to more than 65 million developers from across the globe, GitHub is where developers can create, share, and ship the best code possible. GitHub makes it easier to work together, solve challenging problems, and create the world's most important technologies.

FAST FACTS

65+ million developers

96% reduction in load balancing time

Cut time to rebuild load balancing configurations from 30 minutes to 1

400+ unique applications across thousands of nodes

Reduced costs and time spent during onboarding processes

Significant improvements in security posture

> Terraform has helped us create a self-service business model for our development teams...it reduces friction for developers by eliminating the need to track down reviews and approvals from some centralized gate-keeping team."

SCOTT SANDERS,
VP OF INFRASTRUCTURE

Debugging business processes

For GitHub, the world's largest and most advanced developer collaboration platform serving more than 65 million users, provisioning infrastructure to support application development, coordinating microservices across various cloud environments, and keeping all the connections between them safe and secure is a monumental task.

The company's homegrown secrets management system required manual custom configurations and code to connect hundreds of apps and services, all while managing the keys and secrets by hand. Any changes to the static secrets protecting the company's services and applications meant changing and updating that secret in any of the myriad of systems that connected to it.

Other core activities like load balancing servers were similarly and exceptionally manual — and time-consuming — endeavors that forced the team to create software that wrote out static configurations for every host available at a specific moment. If the team wanted to add service capacity or a new service, in some cases they first had to add a new configuration to a given load balancer, which could negatively impact teams across the organization.

"In the past we'd used a range of homegrown systems for everything from secrets management to load balancing that required a ton of custom integration and custom coding across hundreds of services and thousands of nodes," says Scott Sanders, GitHub's vice president of infrastructure. "But as we started adding more applications, more nodes, and more users, that model became untenable. We needed a more efficient, standard, and automated way to support a dynamic and growing user base."

Challenges

Scaling backend infrastructure to meet the demands of a growing user base

Eliminating time-consuming manual infrastructure provisioning, service matching, and secrets management

Enabling developers to stand up infrastructure themselves as needed

Set it and forget it

As GitHub continued to make strategic infrastructure investments to pair with the platform's growth, there was an opportunity to evolve its previous tech stack with something more flexible and agile. Focused on providing a first-class developer experience, the shift would enable a more agnostic solution with support across multiple clouds and platforms.

GitHub adopted HashiCorp Terraform, Consul, and Vault because the team wanted to run best-of-breed open source tools that could help the company scale its backend operations to meet the growing demands of its internal engineering teams and the customers that depend on GitHub's products. GitHub wanted to worry about effectively scaling their product, not spending time on problems that are already solved within the industry.

In particular, the company wanted established, proven tools that featured a range of automation capabilities and could work seamlessly with workloads across Azure, AWS, and Google Cloud.

"We knew we absolutely wanted infrastructure, discovery, and secrets management tools that could basically run themselves, with robust automation features and easy set up and configuration," Sanders explains. "HashiCorp solutions are intuitive, easy to use, and just continue to work on their own after the initial set up, which frees us to focus on higher value strategies and activities."

According to Aaron Brown, one of the company's infrastructure software engineers, GitHub engineers use Terraform modules to make managing both AWS and Azure cloud resources more approachable and simpler for developers. To do this, Terraform provides workflows and configuration primitives that match up with how the team manages its data center and AWS resources. .

Now, when the team stands up infrastructure in its various clouds and its own internal data centers with bare metal and virtualized hosts, team members have the option to own and run their cloud resources or operate in environments managed by the infrastructure team.

"Nearly all of our hosts, including those in the datacenter, are managed by Terraform and built the same way regardless of whether it's Azure, AWS, or another platform," Brown says. "Terraform has helped us create a self-service business model for our development teams, since they can use Terraform infrastructure configuration templates to provision whatever they need. Or they can come to us and let us do it. It reduces friction for developers by eliminating the need to track down reviews and approvals from some centralized gate-keeping team."

Meanwhile, Consul's key value store feature is a fundamental tool for how the team handles onboarding and offboarding services. The platform-agnostic discovery solution, deployed to thousands of hosts, centralizes services registration and automates service discovery and even maintenance status initiation across hosts in GitHub's various clouds and private datacenters to dramatically simplify and accelerate service connectivity.

At the same time, Vault replaced manual secrets management practices and automatically injects secrets — managing certificates and access control — whenever an app is deployed. Vault's public key infrastructure (PKI) is helpful when the company runs dozens of Kubernetes clusters that were previously managed with command line tools. Vault has allowed for dynamic secret generation, giving GitHub the ability to automate operations and efficiently manage hundreds of thousands of secrets requests per day, saving valuable time and resources. Rather than tasking an entire team with bringing the homegrown secrets solution up to industry standard, they can instead work on larger and more strategic projects.

In addition, GitHub's security team uses Vault's certificate management to manage access control, ensuring that only the teams that should have access to a particular set of secrets are able to access them.

> HashiCorp solutions are intuitive, easy to use, and just continue to work on their own after the initial set up, which frees us to focus on higher value strategies and activities."

SCOTT SANDERS,
VP OF INFRASTRUCTURE

Prioritizing security, efficiency, and reliability

Sanders notes that it's important to have a seamless, first-in-class developer experience, especially when onboarding new engineers who are less familiar with internally developed solutions. Because the technology is approachable and the skill set transferable across the industry, the team is able to save costs and time spent bringing developers up to speed. It also helped bolster the company's compliance and security posture by automating protection of key connection points and hypersensitive data, imperative for a company whose platform houses huge volumes of private information and proprietary data.

Adopting HashiCorp solutions has also made big practical improvements in the company's daily operations. "Previously, deploying a new service required a large number of pull requests to separate repositories and frequently required at least one provisioning command for every host," Sanders explains. "With Consul's templates and built-in load balancing configurations, what used to take 30 minutes to work through all the manual steps to rebuild load balancing configurations for a new service now takes less than a minute."

Brown adds that Terraform has provided similar benefits. "Pre-HashiCorp, prototyping and development of new applications or services used to take several days to bring it to the host, load balance it, then inject secrets — just to reach a minimum viable product," he says. "With Terraform, the process now takes less than an hour and when you're tracking hundreds of different internally important services, every minute you can save is a huge bonus and empowers us to be more effective in engineering."

Another benefit that GitHub has observed is Terraform's extensibility. Having the ability to extend Terraform to help them manage infrastructure in their on-premises datacenters using similar workflows and frameworks they already use in a cloud environment is powerful and has enabled them to bring this capability to the rest of GitHub engineering.

Sanders notes that "HashiCorp solutions allow us to look at the vast ecosystem of OSS tools to solve the problems that GitHub has and then compose them into solutions that form the bedrock of our operations and how we build infrastructure here at GitHub."

Outcomes

Standardized on best-of-breed open source solutions with support for multi-cloud environments

Reduced costs and efforts spent onboarding and training developers

Automated service discovery and secrets management across hundreds of services and thousands of nodes

Reduced load balancing configuration time from 30 minutes to under 1 minute

Solution

GitHub uses HashiCorp Terraform, Consul, and Vault to rapidly provision cloud and on-premises virtual infrastructure, automate service discovery across hundreds of services, and streamline secrets management across the various platforms and instances.

GitHub Partners

Git Systems, storage, SRE, compute foundations, DC space
Scott Sanders is VP, engineering at GitHub, leading the Infrastructure and Site Reliability Engineering teams. Since joining in 2013, he's helped establish the global data center and edge network, migrate to Kubernetes, and establish tooling and best-practices that keep GitHub fast and available.

Scott Sanders,
VP of Infrastructure

Aaron Brown is a staff infrastructure engineer at GitHub. Since joining in 2016, he has been responsible for building the infrastructure that hosts GitHub's hundreds of interconnected services. Prior to GitHub, Brown led infrastructure teams in the ed-tech and e-commerce spaces.

Aaron Brown,
Staff Software Eng. Infrastructure

Technology Stack

- **Infrastructure:** Bare-metal & VMs in data centers, AWS, Azure
- **Container Runtime:** Docker
- **Orchestrator:** Kubernetes
- **CI/CD:** GitHub Actions
- **Data Service:** MySQL, ElasticSearch, Kafka, Git
- **Version Control:** GitHub
- **Provisioning:** HashiCorp Terraform
- **Security management:** HashiCorp Vault


HashiCorp