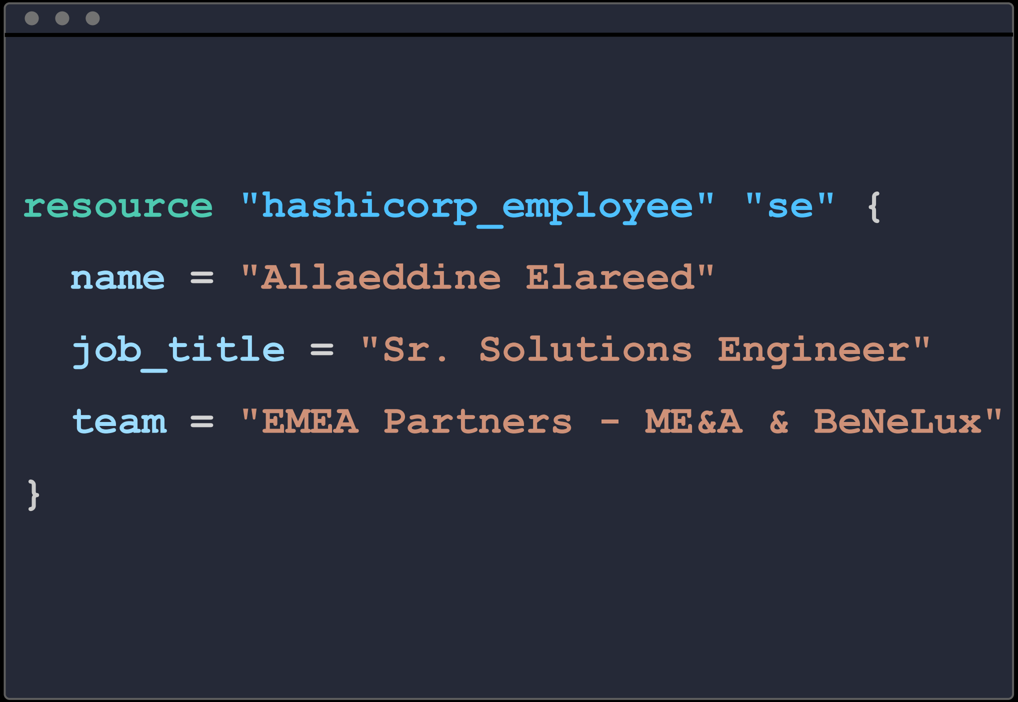
HashiCorp

Cloud-Native Infrastructure as Code

Terraform Cloud vs. Terraform CE DIY Approach

Allaeddine Elareed
Sr. Solutions Engineer



```
resource "hashicorp_employee" "se" {
  name = "Allaeddine Elareed"
  job_title = "Sr. Solutions Engineer"
  team = "EMEA Partners - ME&A & BeNeLux"
}
```

HashiCorp

Terraform

Agenda

Terraform History 01

Why IaC matters? 02

Terraform CE DIY 03

Terraform Cloud 04

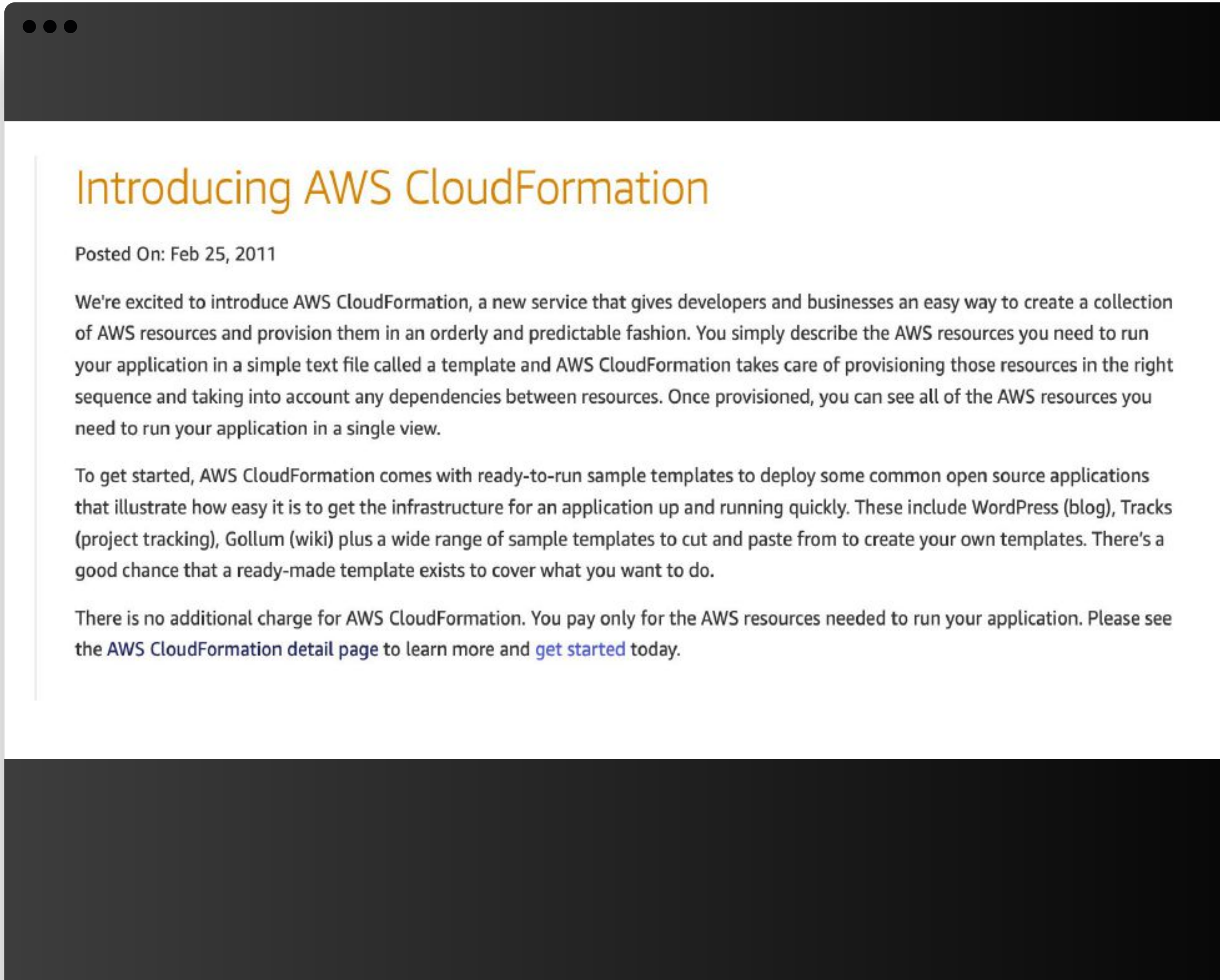
01

Terraform History

The evolution of Terraform

TF History

2011 AWS introduced CF



Introducing AWS CloudFormation

Posted On: Feb 25, 2011

We're excited to introduce AWS CloudFormation, a new service that gives developers and businesses an easy way to create a collection of AWS resources and provision them in an orderly and predictable fashion. You simply describe the AWS resources you need to run your application in a simple text file called a template and AWS CloudFormation takes care of provisioning those resources in the right sequence and taking into account any dependencies between resources. Once provisioned, you can see all of the AWS resources you need to run your application in a single view.

To get started, AWS CloudFormation comes with ready-to-run sample templates to deploy some common open source applications that illustrate how easy it is to get the infrastructure for an application up and running quickly. These include WordPress (blog), Tracks (project tracking), Gollum (wiki) plus a wide range of sample templates to cut and paste from to create your own templates. There's a good chance that a ready-made template exists to cover what you want to do.

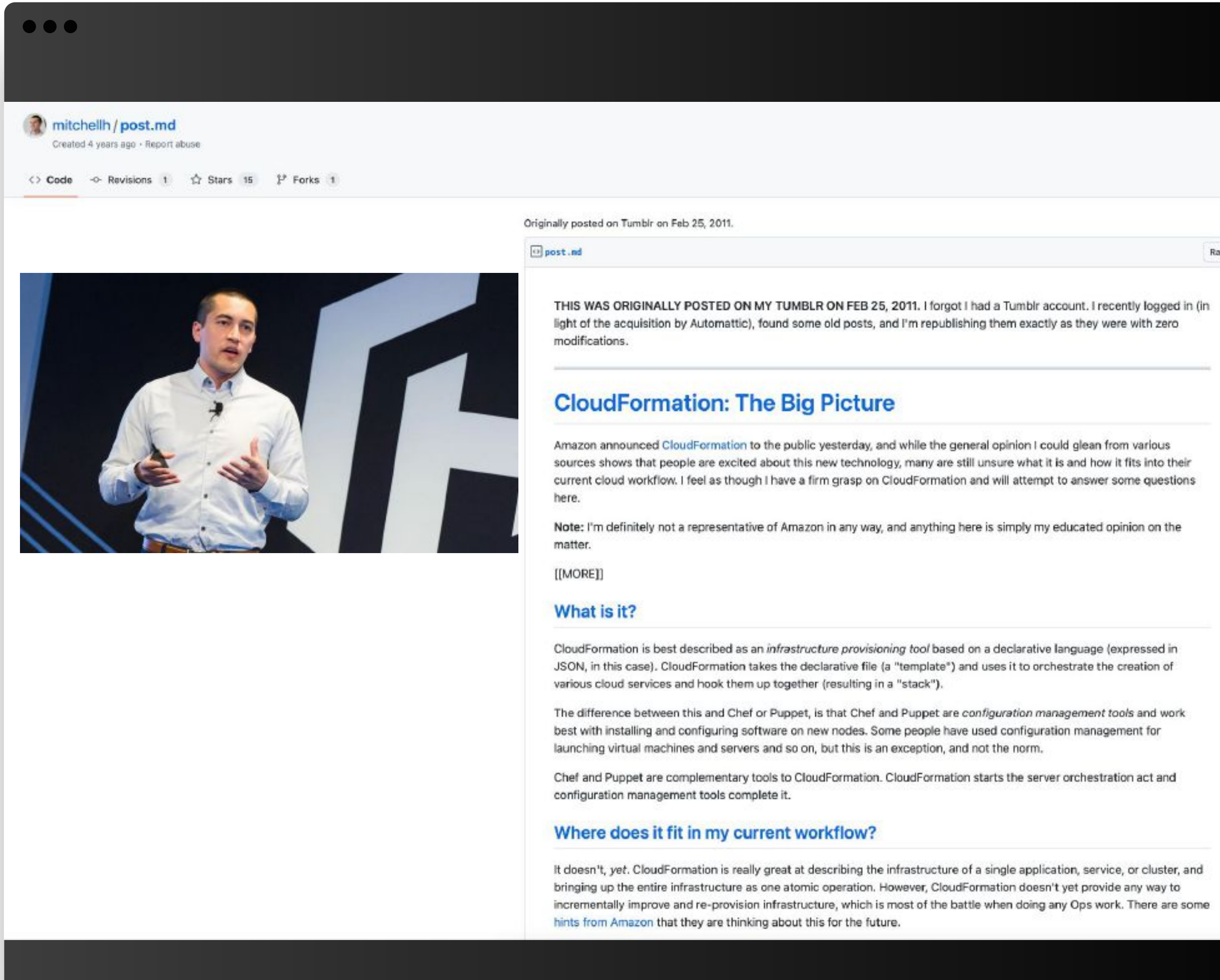
There is no additional charge for AWS CloudFormation. You pay only for the AWS resources needed to run your application. Please see the AWS CloudFormation detail page to learn more and get started today.

TF History

2011 AWS introduced CF

Need for an open source, cloud-agnostic solution

Jul 2014 released TF v. 0.1



Originally posted on Tumblr on Feb 25, 2011.

THIS WAS ORIGINALLY POSTED ON MY TUMBLR ON FEB 25, 2011. I forgot I had a Tumblr account. I recently logged in (in light of the acquisition by Automattic), found some old posts, and I'm republishing them exactly as they were with zero modifications.

CloudFormation: The Big Picture

Amazon announced CloudFormation to the public yesterday, and while the general opinion I could glean from various sources shows that people are excited about this new technology, many are still unsure what it is and how it fits into their current cloud workflow. I feel as though I have a firm grasp on CloudFormation and will attempt to answer some questions here.

Note: I'm definitely not a representative of Amazon in any way, and anything here is simply my educated opinion on the matter.

[[MORE]]

What is it?

CloudFormation is best described as an *infrastructure provisioning tool* based on a declarative language (expressed in JSON, in this case). CloudFormation takes the declarative file (a "template") and uses it to orchestrate the creation of various cloud services and hook them up together (resulting in a "stack").

The difference between this and Chef or Puppet, is that Chef and Puppet are *configuration management tools* and work best with installing and configuring software on new nodes. Some people have used configuration management for launching virtual machines and servers and so on, but this is an exception, and not the norm.

Chef and Puppet are complementary tools to CloudFormation. CloudFormation starts the server orchestration act and configuration management tools complete it.

Where does it fit in my current workflow?

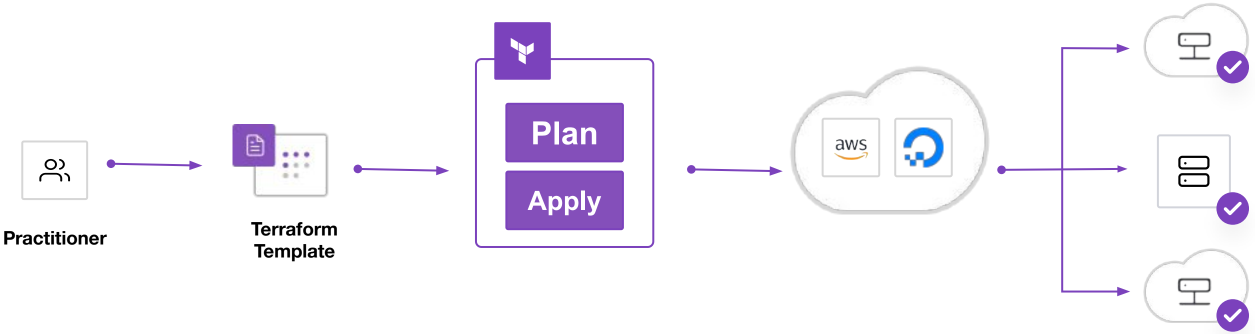
It doesn't, *yet*. CloudFormation is really great at describing the infrastructure of a single application, service, or cluster, and bringing up the entire infrastructure as one atomic operation. However, CloudFormation doesn't yet provide any way to incrementally improve and re-provision infrastructure, which is most of the battle when doing any Ops work. There are some hints from Amazon that they are thinking about this for the future.

TF History

2011 AWS introduced CF

Need for an open source, cloud-agnostic solution

Jul 2014 released TF v. 0.1
2014 - 2016 The hard time



TF History

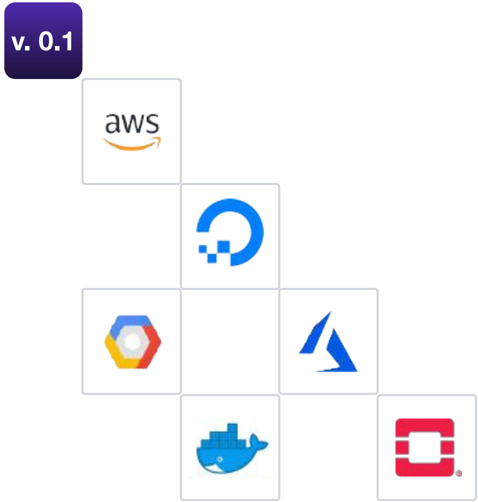
2011 AWS introduced CF

Need for an open source, cloud-agnostic solution

Jul 2014 released TF v. 0.1

2014 - 2016 The hard time

2017 - The year of Terraform



TF History

2011 AWS introduced CF

Need for an open source, cloud-agnostic solution

Jul 2014 released TF v. 0.1

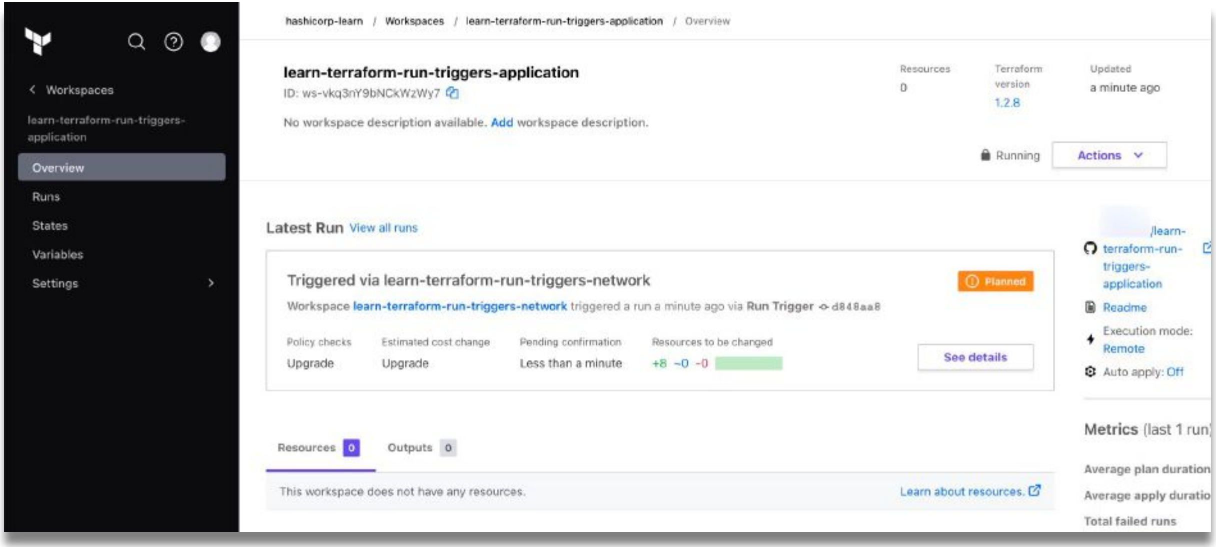
2014 - 2016 The hard time

2017 - The year of Terraform

2018 - 2020 TFC / TFE

Terraform Cloud

Terraform Enterprise



hashicorp-learn / Workspaces / learn-terraform-run-triggers-application / Overview

learn-terraform-run-triggers-application

ID: ws-vkq3nY9bNCkWzWy7

No workspace description available. Add workspace description.

Resources: 0

Terraform version: 1.2.8

Updated: a minute ago

Running

Actions

Latest Run View all runs

Triggered via learn-terraform-run-triggers-network

Planned

Workspace learn-terraform-run-triggers-network triggered a run a minute ago via Run Trigger d848aa8

Policy checks	Estimated cost change	Pending confirmation	Resources to be changed
Upgrade	Upgrade	Less than a minute	+8 ~0 -0

See details

Resources 0 | Outputs 0

This workspace does not have any resources. Learn about resources.

/learn-terraform-run-triggers-application

Readme

Execution mode: Remote

Auto apply: Off

Metrics (last 1 run)

Average plan duration

Average apply duratio

Total failed runs

TF History

2011 AWS introduced CF

Need for an open source, cloud-agnostic solution

Jul 2014 released TF v. 0.1
2014 - 2016 The hard time

2017 - The year of Terraform

2018 - 2020 TFC / TFE

Today

3,000+
Providers

12,000+
Modules

20+
Run task partners

250M+
Downloads*

30K+
Certified users

2,500+
Customers

**Source: Downloads as of FY2022, Terraform customers as of 3Q FY23*

02

Why IaC matters?

Shifting to cloud infrastructure

DEDICATED INFRASTRUCTURE

Traditional Datacenter

“Static”

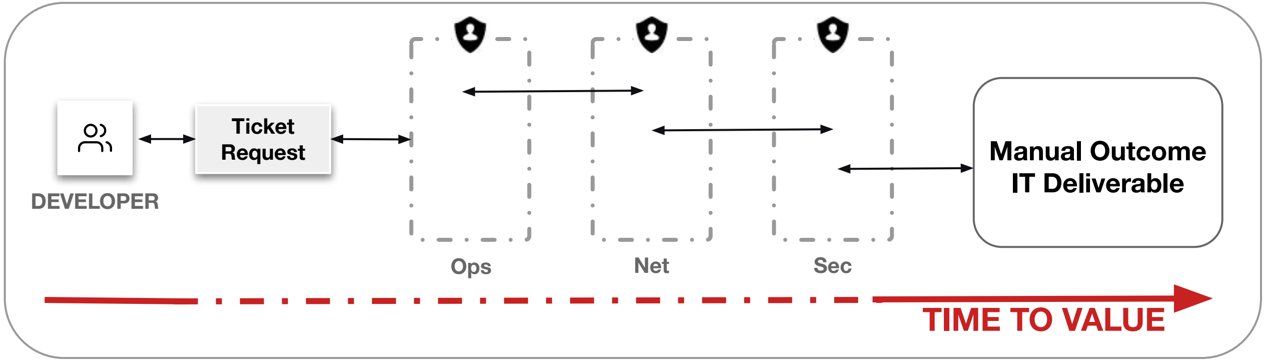
ITIL & Tickets

The Iron Age

Basic ITIL Workflow:

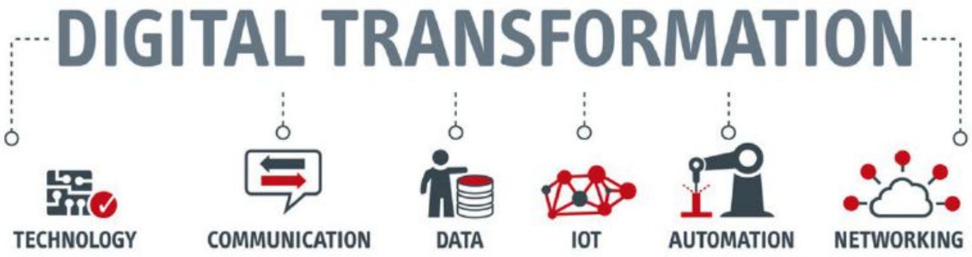
Gatekeeping process through silos

- Waterfall Model (1st, 2nd, 3rd...)
- Engineering Approach (Big Bang)
- Static and long lasting outcomes



Traditional **manual** and **static** Workflows

“ClickOps” & Imperative infrastructure driven



Means pressure on IT

- **Time to market** - doing things faster
- Increase **productivity** and lower **cost**
- **Security** & **governance**

Gartner predicts that by **2026**, **75%** of organizations will adopt a **digital transformation** model predicated on **cloud** as the **fundamental underlying platform**.

https://www.gartner.com/en/newsroom/press-releases/2023-04-19-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-reach-nearly-600-billion-in-2023

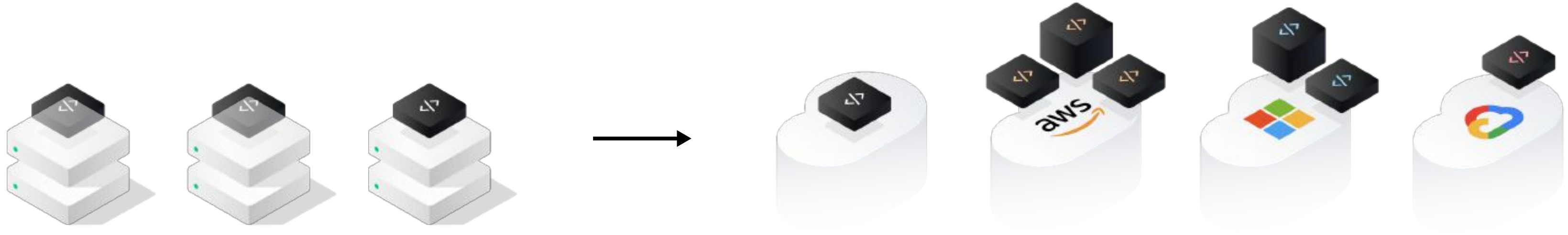
Shifting to cloud infrastructure

PRIVATE CLOUD

Systems of Record

PUBLIC CLOUD

Systems of Engagement



Traditional Datacenter

“Static”

ITIL & Tickets

Modern Datacenter

“Dynamic”

Self-Service & APIs

Infrastructure as Code

Challenge Solution Results

Manual provisioning

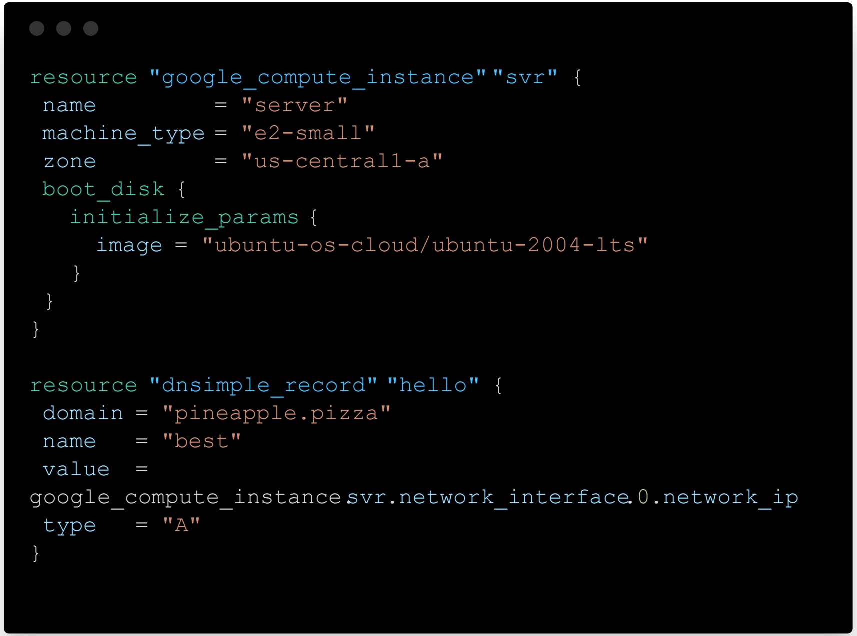
Manual provisioning through point-and-click GUIs or custom scripts is slow, error-prone, inefficient, and difficult to use at scale.

Infrastructure as Code

Challenge | **Solution** | Results

With HashiCorp Configuration Language (HCL), infrastructure and services from any provider can be provisioned in a codified, secure, and automated fashion.

- HashiCorp Configuration Language (HCL) is human readable and machine executable
- Declarative, Turing-complete language
- Codify, version, and collaborate on infrastructure



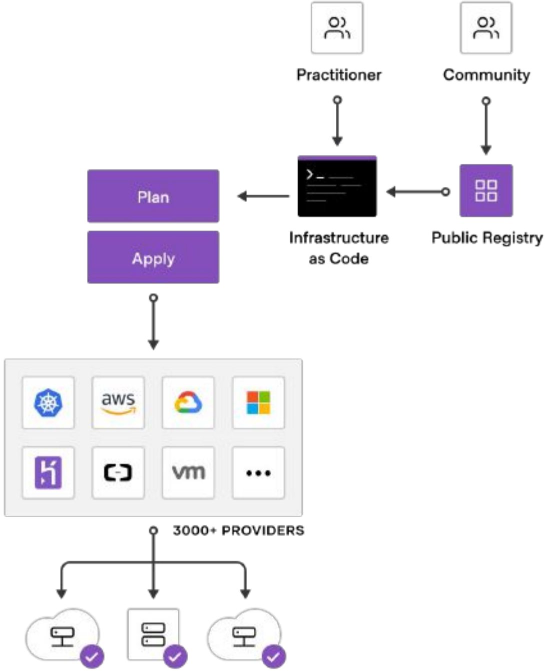
```
resource "google_compute_instance" "svr" {
 name         = "server"
 machine_type = "e2-small"
 zone         = "us-central1-a"
 boot_disk {
   initialize_params {
     image = "ubuntu-os-cloud/ubuntu-2004-lts"
   }
 }
}

resource "dnsimple_record" "hello" {
 domain = "pineapple.pizza"
 name   = "best"
 value  =
google_compute_instance.svr.network_interface.0.network_ip
 type   = "A"
}
```

Infrastructure as Code

Challenge | **Solution** | Results

Compose, collaborate, reuse

- Use version control and automation to reduce human error and failed builds
- Adopt Terraform infrastructure as code and policy as code to automate everything
- Provider plugins allow rapid creation and support for any infrastructure

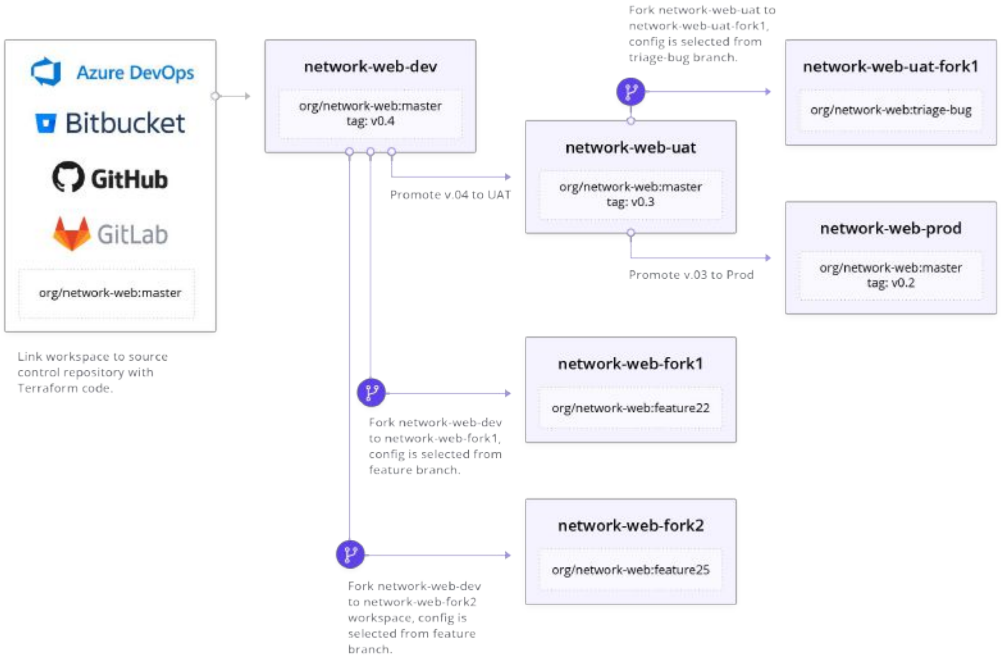


Infrastructure as Code

Challenge Solution Results

Benefits of infrastructure as code

- Versioning
- Collaboration
- Promotion
- Reuse



Azure DevOps

Bitbucket

GitHub

GitLab

org/network-web:master

Link workspace to source control repository with Terraform code.

network-web-dev

org/network-web:master
tag: v0.4

Promote v.04 to UAT

network-web-uat

org/network-web:master
tag: v0.3

Fork network-web-uat to network-web-uat-fork1, config is selected from triage-bug branch.

network-web-uat-fork1

org/network-web:triage-bug

Promote v.03 to Prod

network-web-prod

org/network-web:master
tag: v0.2

Fork network-web-dev to network-web-fork1, config is selected from feature branch.

network-web-fork1

org/network-web:feature22

Fork network-web-dev to network-web-fork2 workspace, config is selected from feature branch.

network-web-fork2

org/network-web:feature25

Infrastructure as Code

Challenge Solution **Results**

Increase Agility

With infrastructure as code, the manual effort involved in provisioning infrastructure is significantly reduced. Code can be reused and modified as many times as necessary.

Reduce Risk

Minimize manual, error-prone work and reuse known-working and known-secure infrastructure configurations across an organization.

Reduce Cost

By defining proper infrastructure footprints in modules, teams provision the infrastructure they need without wasteful and costly over-provisioning.

Infrastructure as Code is one of the cornerstones of DevOps.

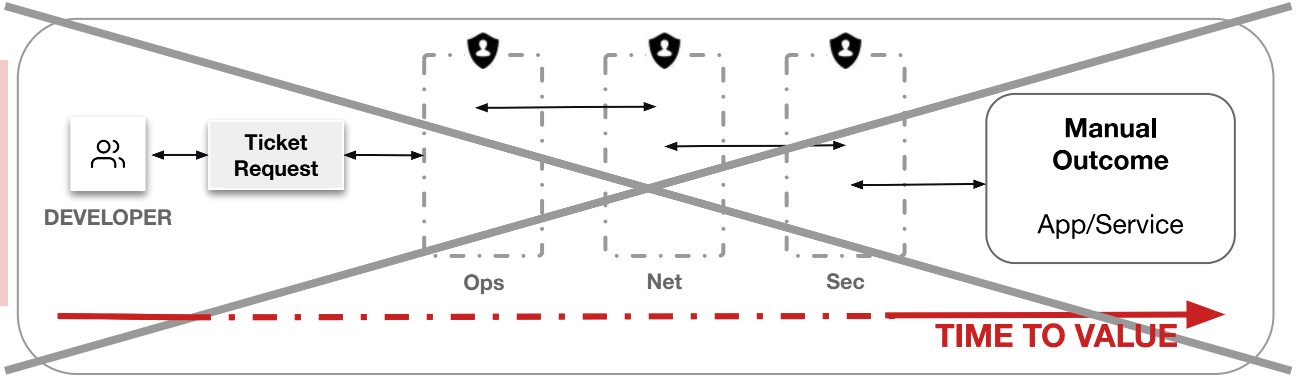
It is the “A” in “CAMS”: Culture, Automation, Measurement, and Sharing.

The Cloud Age

Basic ITIL Workflow:

Gatekeeping process through silos

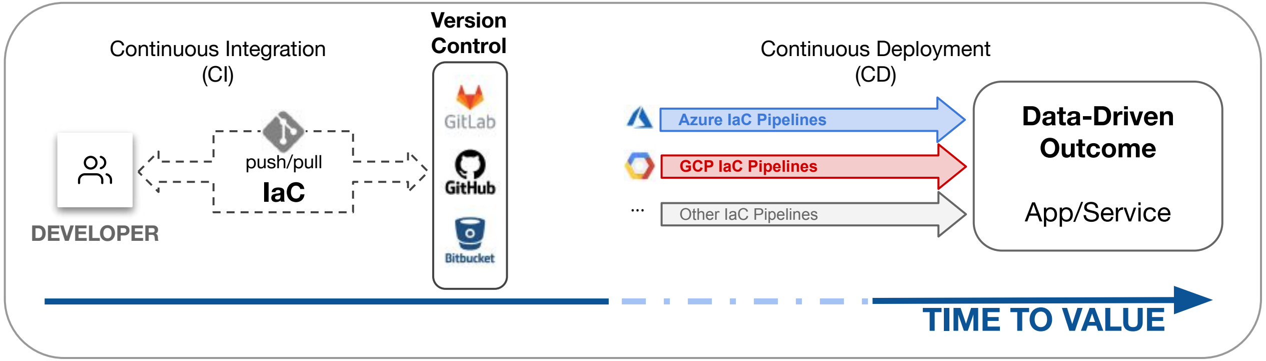
- Waterfall Model (1st, 2nd, 3rd...)
- Engineering Approach (Big Bang)
- Static and long lasting outcomes



DevOps Workflow:

CI/CD process through stages (dev...prod)

- IaC / Data driven (non-consecutive)
- MVP Approach (small chunks)
- Dynamic and short lived cycles

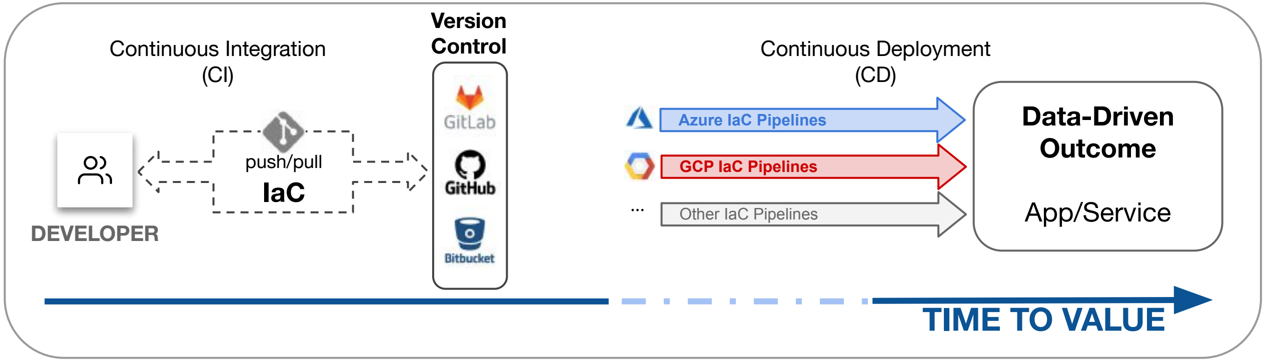


A single IaC platform

DevOps Workflow:

CI/CD process through stages (dev...prod)

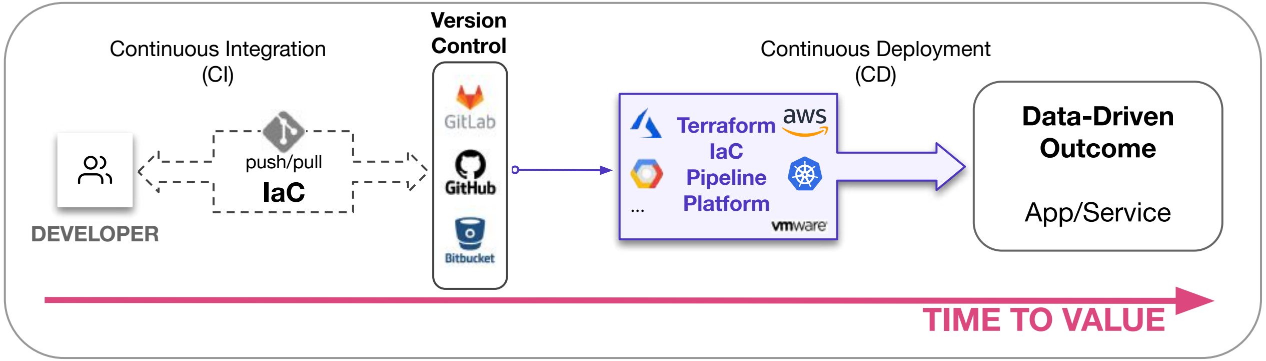
- **IaC / Data driven (non-consecutive)**
- **MVP Approach (small chunks)**
- **Dynamic and short lived cycles**



Terraform Workflow:

Unified process through a single platform

- **Workflow over technology**
- **Immutable infrastructure**
- **Learn once; use everywhere**

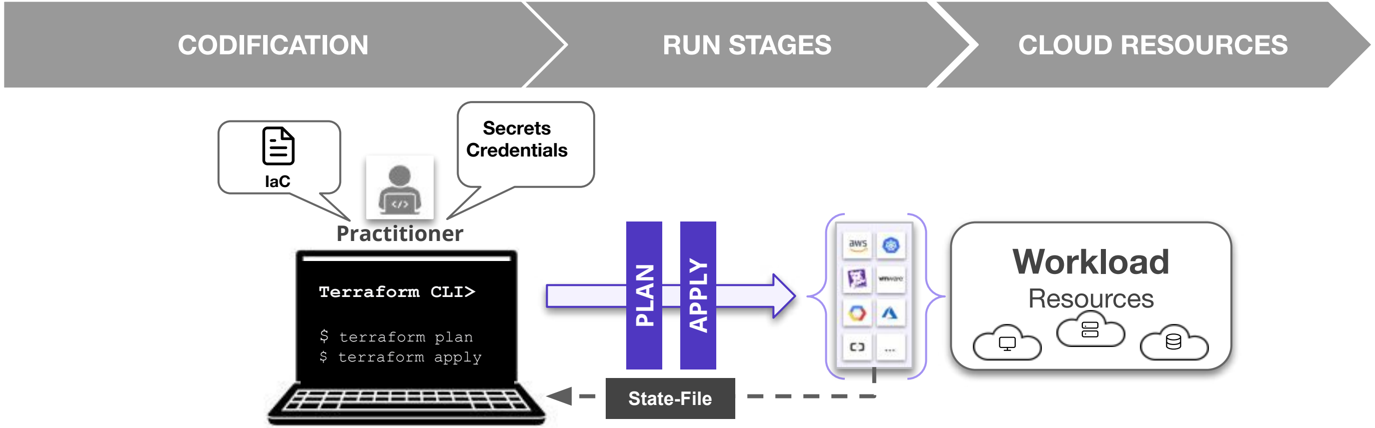


03

Terraform CE DIY

Playing Terraform on “Hard Mode”

Terraform CLI



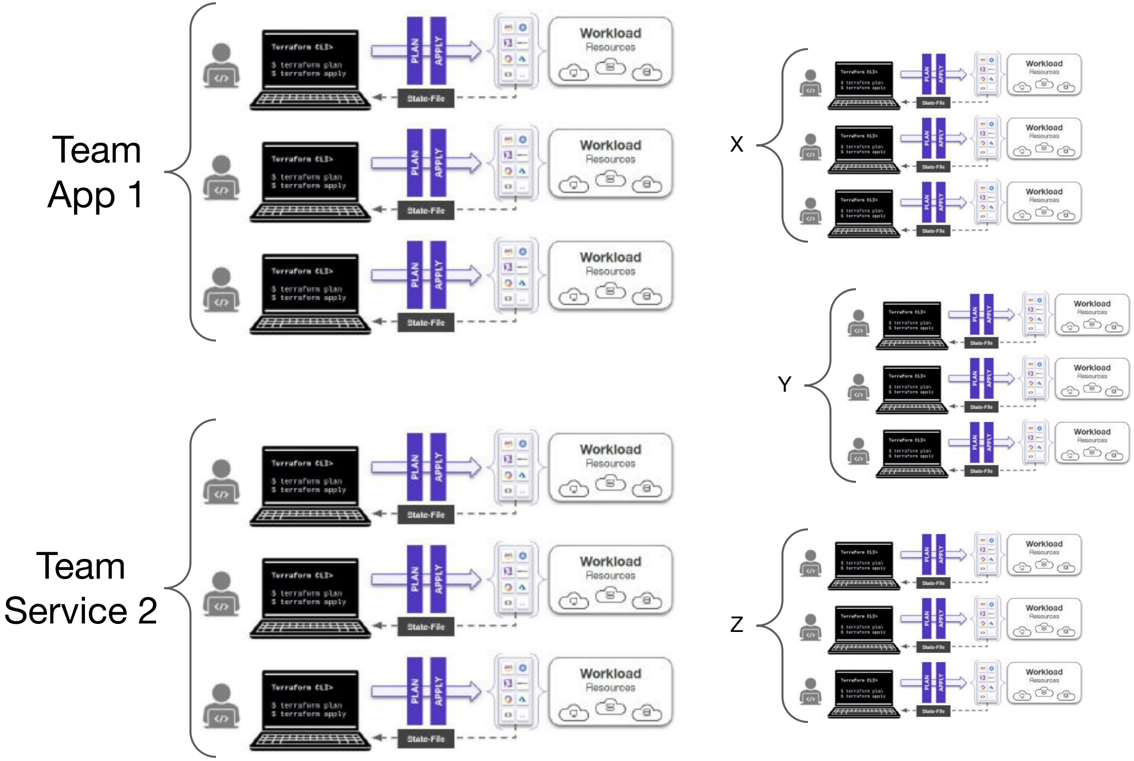
```
Terraform CLI>

$ terraform plan
$ terraform apply
```

- Terraform CLI is our open source command line tool that can be downloaded from our website.

- It is a single binary installed on your local machine that is capable to execute IaC written in HCL.

- It is made for single practitioners to abstract any workload.

TF CE/CLI at scale



Enterprise concerns:

- Sprawl of credentials

- Sprawl of State-Files

- no governance & control

- no collaboration features

- no audit-trail

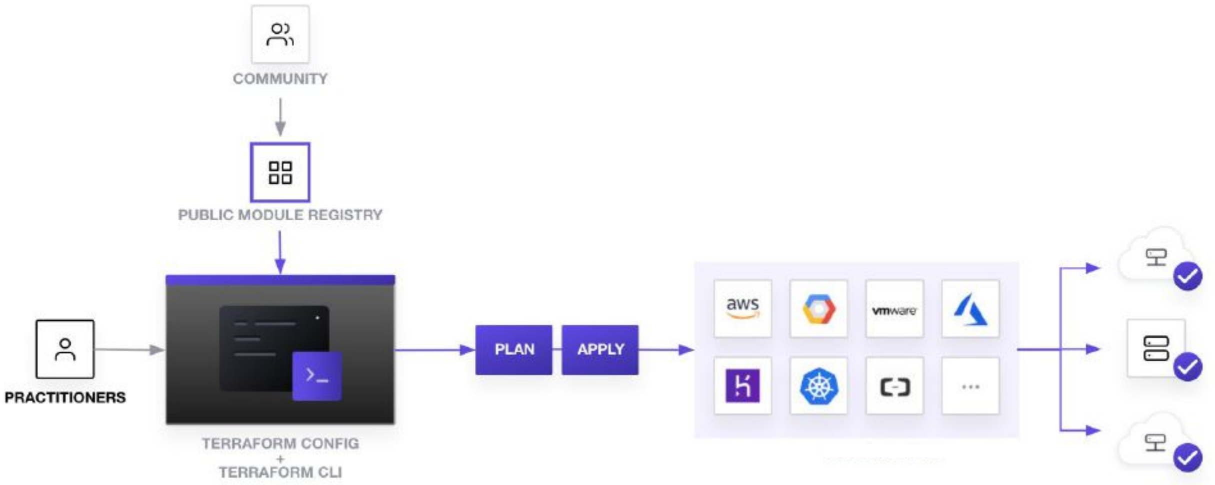
- no RBAC, no SSO

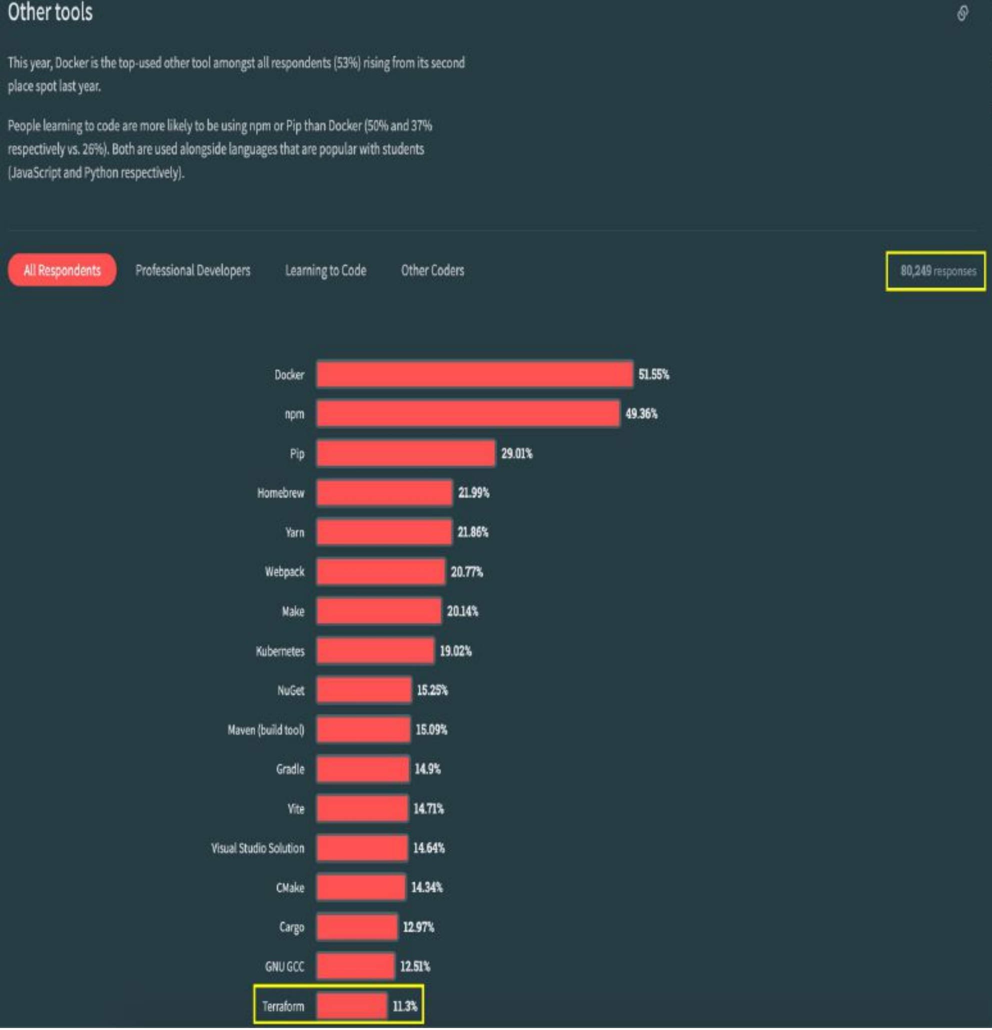
- no integration into existing ecosystems

…

Terraform CE Workflow

- Custom **scaffolding** for any automation (CI/CD tools)
- Manual **state file** management
- Expanded logging and auditing plane

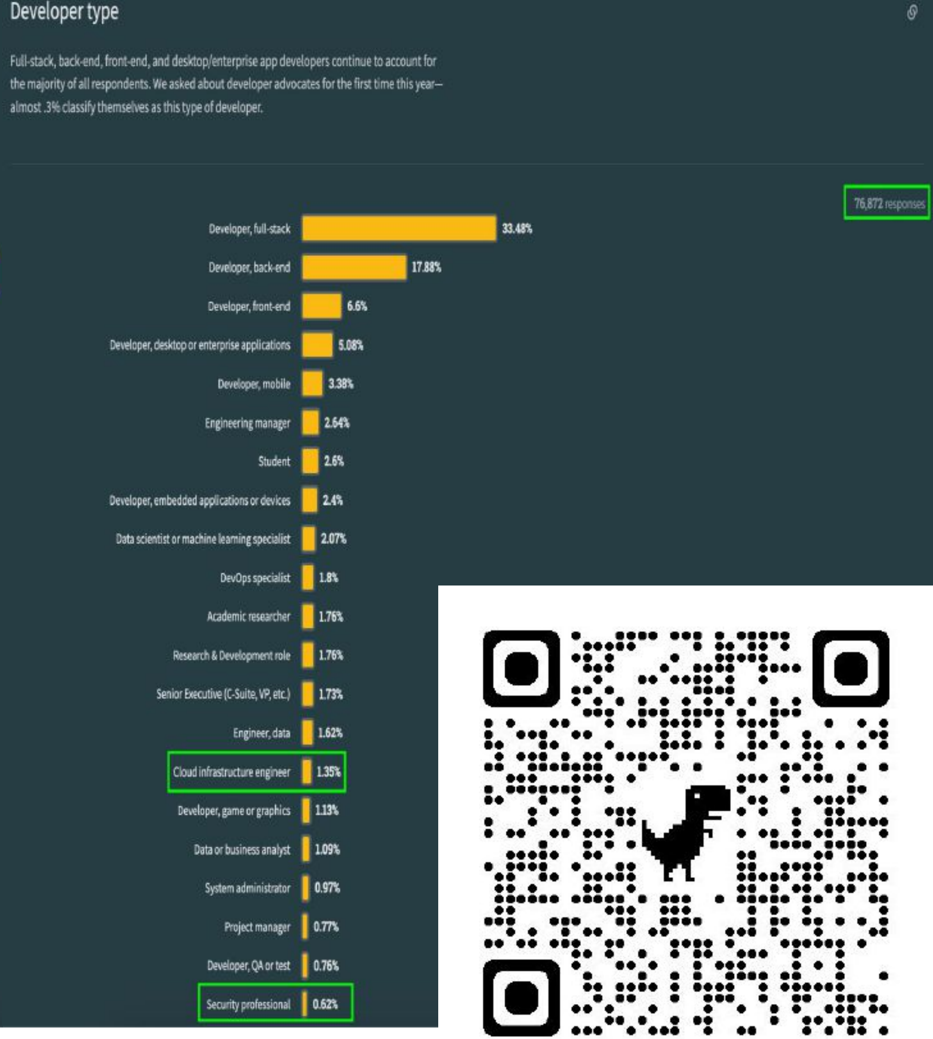




Other tools

This year, Docker is the top-used other tool amongst all respondents (53%) rising from its second place spot last year.

People learning to code are more likely to be using npm or Pip than Docker (50% and 37% respectively vs. 26%). Both are used alongside languages that are popular with students (JavaScript and Python respectively).



Developer type

Full-stack, back-end, front-end, and desktop/enterprise app developers continue to account for the majority of all respondents. We asked about developer advocates for the first time this year—almost .3% classify themselves as this type of developer.

76,872 responses
Developer, full-stack 33.48%
Developer, back-end 17.88%
Developer, front-end 6.6%
Developer, desktop or enterprise applications 5.08%
Developer, mobile 3.38%
Engineering manager 2.64%
Student 2.6%
Developer, embedded applications or devices 2.4%
Data scientist or machine learning specialist 2.07%
DevOps specialist 1.8%
Academic researcher 1.76%
Research & Development role 1.76%
Senior Executive (C-Suite, VP, etc.) 1.73%
Engineer, data 1.62%
Cloud infrastructure engineer 1.35%
Developer, game or graphics 1.13%
Data or business analyst 1.09%
System administrator 0.97%
Project manager 0.77%
Developer, QA or test 0.76%
Security professional 0.62%

Stack Overflow Developer Survey 2023 - https://survey.stackoverflow.co/2023/

The big picture

7 numbers to remember

56%

Boosted cloud spending in the last year, despite macroeconomic uncertainty

92%

Of high-cloud-maturity organizations say multi-cloud is working, or is expected to within a year

53%

Of high-maturity organizations are using multi-cloud to save money

74%

Of high-maturity companies say multi-cloud helps them attract, motivate, and retain talent

#1

Rank of skills shortages as a multi-cloud barrier

#1

Rank of security as a multi-cloud driver

92%

Of organizations are adopting, standardizing, or scaling platform teams

https://www.hashicorp.com/state-of-the-cloud

So, what it takes?

Collaborative IaC

Protecting and updating state files

Reporting, how IaC performs when applied.

Standardization across teams, avoid duplication of efforts

Pre-built and well-documented Workflows

Compliance & Mgmt.

Governance & Security

Audit Logging

Role-based Access

State File, Variable Encryption & Creds Mgmt.

Cost Tracking

Drift Detection

Continuous Validation

Self-Service Infra.

Standardized Environments to automate provisioning

Provisioning Compliance and Control

No-code option for less experienced users

R&D + Support

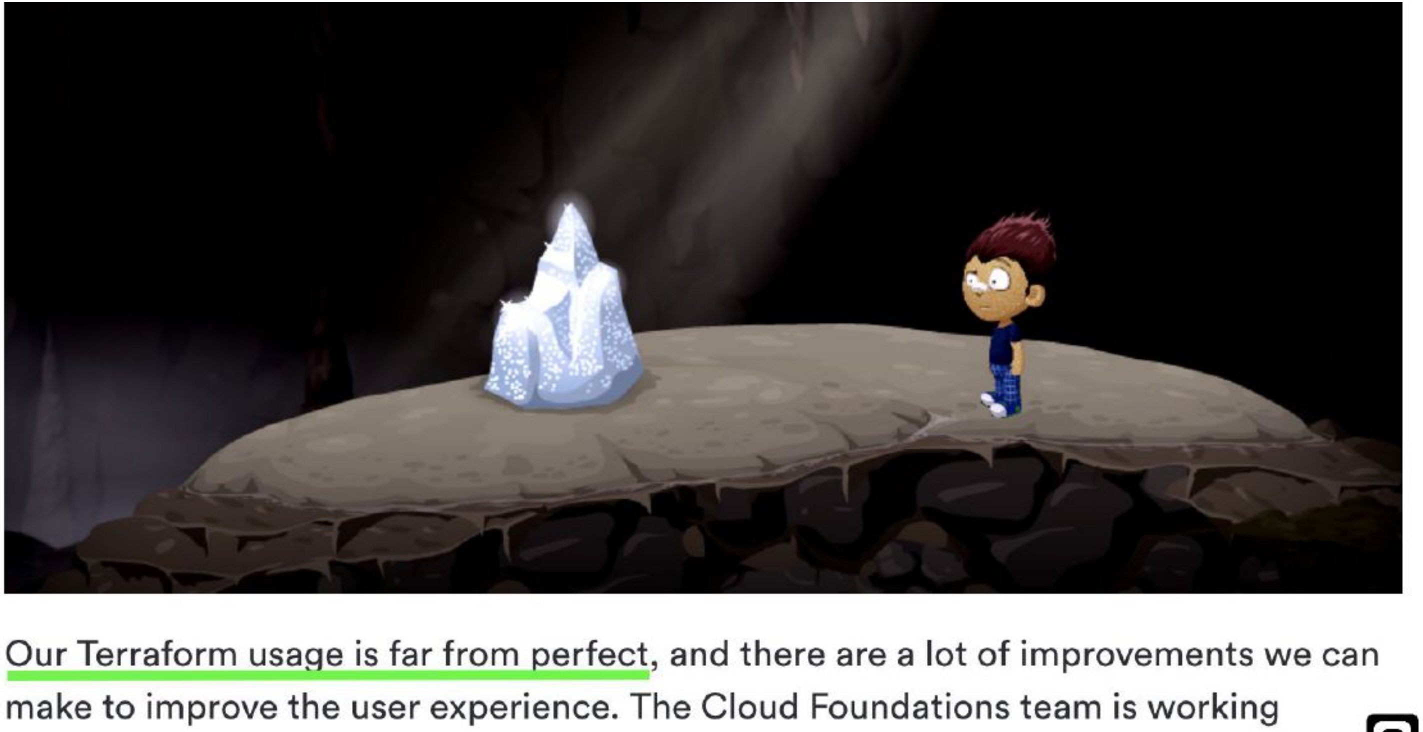
Architectural design - Strategies for workflow, approval gates, isolation, etc

Product releases

Bug fixes & upgrades

Operational/hosting costs

Availability



Our Terraform usage is far from perfect, and there are a lot of improvements we can make to improve the user experience. The Cloud Foundations team is working

https://slack.engineering/how-we-use-terraform-at-slack/

04

Terraform Cloud

Enabling Platform Team Capabilities

Standardize workflow, manage infrastructure lifecycle, operate at scale

Unified Workflow Management

RBAC | Remote State Storage | Registry | No Code Workflow

Policy & Security

Sentinel, OPA Policy | Run Tasks | Enforcement

Visibility & Optimization

Workspace Mgmt | Drift Detection | Continuous Validation | Alerts | Audit Logs | Roll-back/forward

Reliability & Scale

Managed HA | Self-Managed HA | Self-Hosted Agents

Governance, Risk, & Compliance

SOC Compliance | 24×7 Support

Integrations & API

Okta, Splunk, Waypoint, ServiceNow, HCP Packer

Dynamic Provider Credentials

Challenge Solution

Credential management at scale

Managing static, long-lived credentials in Terraform Cloud causes operational complexity for platform teams and introduces security risks.

Dynamic Provider Credentials

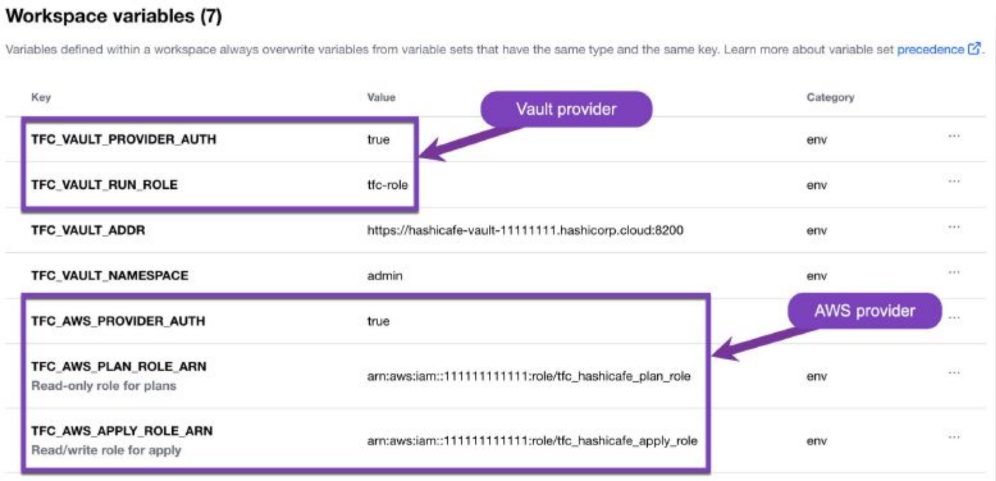
Challenge Solution

Native just-in-time (JIT) provider authentication

A native solution for JIT access using Terraform workload identity and provider OIDC support.

Configure dynamic credential injection via workspace or project-level variables for:

- Vault
- AWS
- Azure
- Google Cloud



Workspace variables (7)

Variables defined within a workspace always overwrite variables from variable sets that have the same type and the same key. Learn more about variable set precedence.

Key	Value	Category	
TFC_VAULT_PROVIDER_AUTH	true	env	···
TFC_VAULT_RUN_ROLE	tfc-role	env	···
TFC_VAULT_ADDR	https://hashicafe-vault-11111111.hashicorp.cloud:8200	env	···
TFC_VAULT_NAMESPACE	admin	env	···
TFC_AWS_PROVIDER_AUTH	true		···
TFC_AWS_PLAN_ROLE_ARN Read-only role for plans	arn:aws:iam::111111111111:role/tfc_hashicafe_plan_role	env	···
TFC_AWS_APPLY_ROLE_ARN Read/write role for apply	arn:aws:iam::111111111111:role/tfc_hashicafe_apply_role	env	···

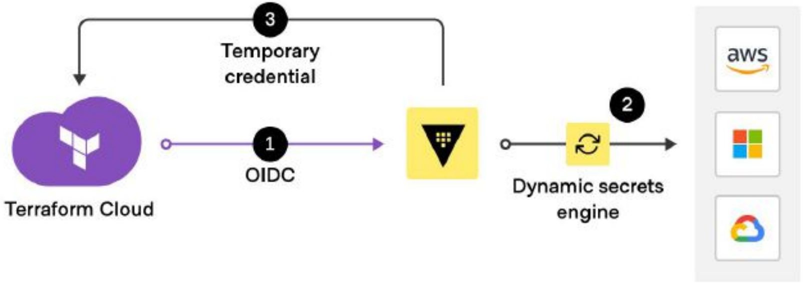
Vault-Backed Dynamic Credentials

Challenge Solution

Centralized credential management

Combine the power of dynamic provider credentials and Vault dynamic secrets engines.

- Authenticate Terraform runs to Vault using JWT/OIDC auth method
- Vault generates temporary cloud credentials for AWS, Azure, or Google Cloud
- Secrets are injected into the Terraform agent environment for use with providers
- Credentials are revoked immediately after each run phase



** No inbound OIDC connectivity from cloud providers required for Terraform Enterprise

Team Management

Challenge Solution

Role-based access control (RBAC)

To maintain proper security posture, organizations should provide access to configurations and provisioning only as needed by team members.

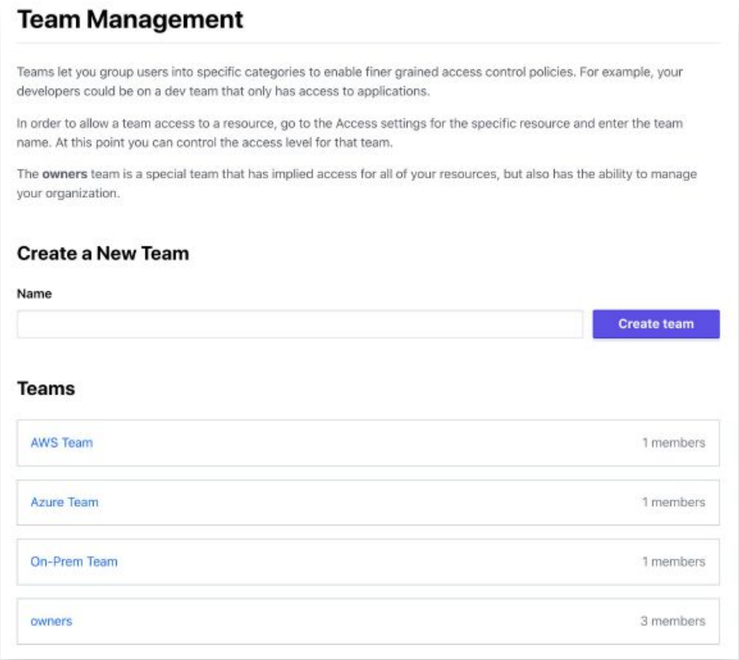
Team Management

Challenge Solution

Role-based access control

Group users into teams for access to projects and workspaces to achieve “least privilege”.

- Full role-based access control
- Assign users to one or more teams
- Granular management of team permissions



Team Management

Teams let you group users into specific categories to enable finer grained access control policies. For example, your developers could be on a dev team that only has access to applications.

In order to allow a team access to a resource, go to the Access settings for the specific resource and enter the team name. At this point you can control the access level for that team.

The **owners** team is a special team that has implied access for all of your resources, but also has the ability to manage your organization.

Create a New Team

Name

Teams

AWS Team	1 members
Azure Team	1 members
On-Prem Team	1 members
owners	3 members

Modules

Challenge Solution

Non-standardized provisioning

Without templated infrastructure as code, operators spend time manually fulfilling infrastructure requests, or developers provision cloud resources for their applications without oversight or guardrails.

Modules

Challenge

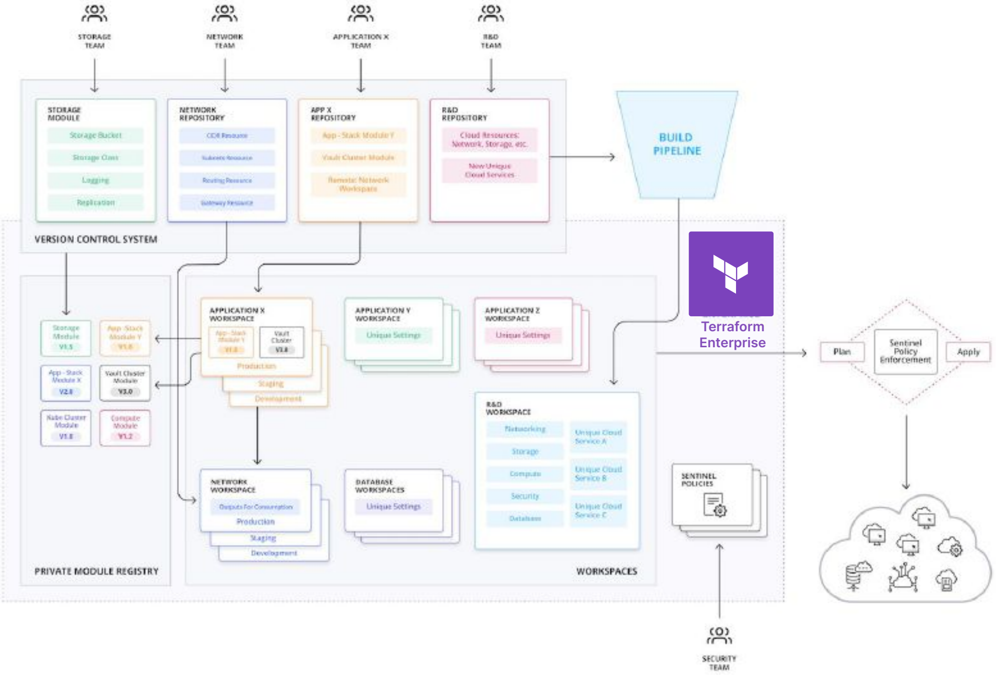
Solution

By creating reusable modules, operations teams empower their organization to efficiently provision approved, secured, and standardized infrastructure.

- Reusable, templated infrastructure as code
- Create interfaces with input and output variables

Producer / Consumer Workflow

- Producers create modules and publish to a registry for discovery
- Consumers explore the registry to create infrastructure as needed for applications



Policy as Code

Challenge   Solution

Guardrails around multi-cloud provisioning

Rapid provisioning opens up tremendous possibility, but organizations need to maintain security, compliance, and prevent over-provisioning.

Policy as Code

Challenge Solution

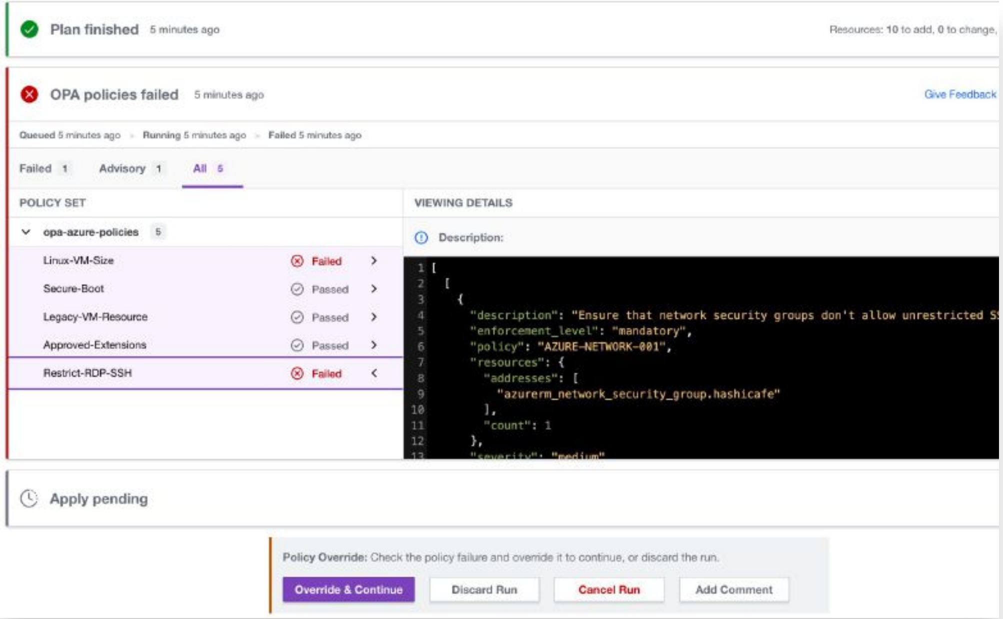
Native Open Policy Agent support

Leverage existing OPA skills

- Add Rego policies to your provisioning workflow with a first-class integration
- Coexist alongside Sentinel policies

Enforcement levels

- Advisory: Warning when a policy fails
- Mandatory: Block provisioning when a policy fails
- Allow or prevent overrides at the policy set level



Plan finished 5 minutes ago

Resources: 10 to add, 0 to change,

OPA policies failed 5 minutes ago

Give Feedback

Queued 5 minutes ago › Running 5 minutes ago › Failed 5 minutes ago

Failed 1 Advisory 1 All 5

POLICY SET	
opa-azure-policies 5	
Linux-VM-Size	Failed
Secure-Boot	Passed
Legacy-VM-Resource	Passed
Approved-Extensions	Passed
Restrict-RDP-SSH	Failed

VIEWING DETAILS

Description:

```
[
  [
    {
      "description": "Ensure that network security groups don't allow unrestricted S
      "enforcement_level": "mandatory",
      "policy": "AZURE-NETWORK-001",
      "resources": {
        "addresses": [
          "azurerm_network_security_group.hashicafe"
        ],
        "count": 1
      },
      "severity": "medium"
```

Apply pending

Policy Override: Check the policy failure and override it to continue, or discard the run.

Cost Estimation

Challenge Solution

Visibility and impact of infrastructure cost

Cloud presents a decentralized purchasing model and gives everyone the ability to spend money for the company.

A challenge is enabling the practitioners deploying the infrastructure changes to understand the financial impact of the changes they're applying.

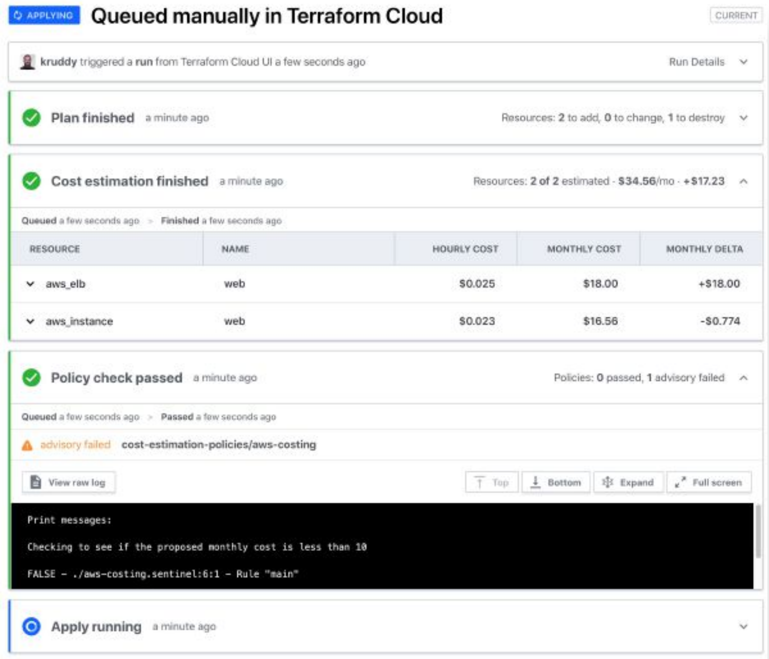
Cost Estimation

Challenge | **Solution**

Cloud cost management

Approaches for managing cost in a self-service operational model:

- Visibility: cost estimation before provisioning
- Management: set & enforce Sentinel policies
- Optimization: business changes to optimize cost long term



Queued manually in Terraform Cloud

kruddy triggered a **run** from Terraform Cloud UI a few seconds ago

✓ **Plan finished** a minute ago — Resources: **2** to add, **0** to change, **1** to destroy

✓ **Cost estimation finished** a minute ago — Resources: **2 of 2** estimated · **$34.56**/mo · **+$17.23**

Queued a few seconds ago > **Finished** a few seconds ago

RESOURCE	NAME	HOURLY COST	MONTHLY COST	MONTHLY DELTA
aws_elb	web	$0.025	$18.00	+$18.00
aws_instance	web	$0.023	$16.56	-$0.774

✓ **Policy check passed** a minute ago — Policies: **0** passed, **1** advisory failed

Queued a few seconds ago > **Passed** a few seconds ago

⚠ advisory failed **cost-estimation-policies/aws-costing**

```
Print messages:

Checking to see if the proposed monthly cost is less than 10

FALSE - ./aws-costing.sentinel:6:1 - Rule "main"
```

○ **Apply running** a minute ago

Ephemeral Workspaces

Challenge Solution

Clean up temporary resources

Many dev/test workflows require temporary infrastructure. But these resources are often left running long after they are needed, incurring unnecessary costs.

Ephemeral Workspaces (Beta)

Challenge Solution

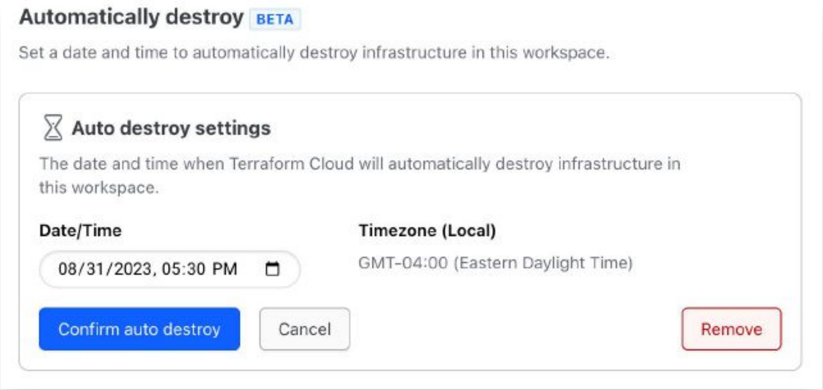
Automatic resource destruction

Set a time-to-live for a workspace to auto-initiate a destroy run.

Configure reminder and completion notifications.

Use cases include:

- Development sandboxes
- Automated test pipelines
- Demo or classroom lab environments



Automatically destroy BETA

Set a date and time to automatically destroy infrastructure in this workspace.

Auto destroy settings

The date and time when Terraform Cloud will automatically destroy infrastructure in this workspace.

Date/Time: 08/31/2023, 05:30 PM

Timezone (Local): GMT-04:00 (Eastern Daylight Time)

CI/CD Pipeline Templates

Challenge Solution

Integrating into existing pipelines

Many organizations want to interact with Terraform through existing CI/CD tools, but building and maintaining custom workflows is challenging.

CI/CD Pipeline Templates

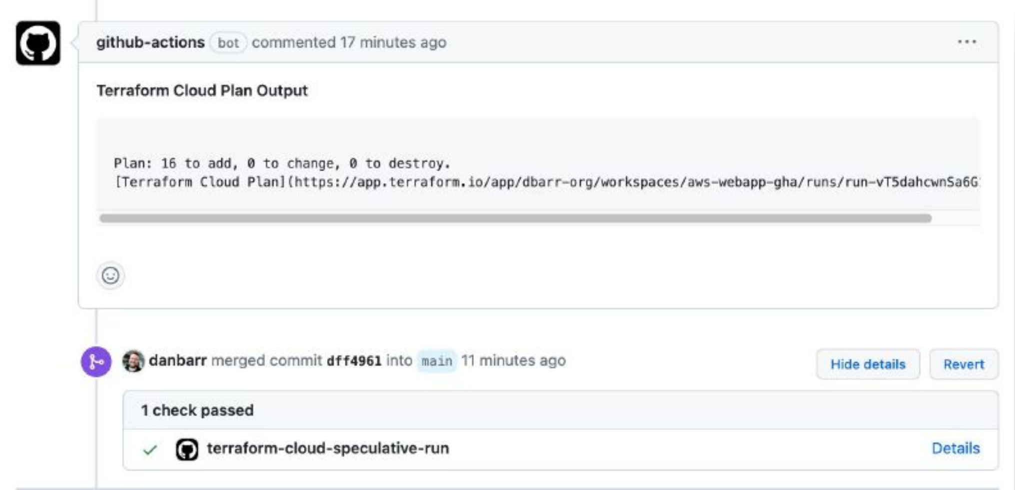
Challenge   **Solution**

Get up and running quickly with Terraform Cloud and Enterprise.

Integrate with existing CI/CD pipelines to minimize process changes:

- Containerized tool that implements common API functions
- Predefined templates for GitHub Actions and GitLab CI/CD
- Apply as a prescriptive workflow or integrate actions into existing pipelines

github.com/hashicorp/tfc-workflows-tooling



github-actions bot commented 17 minutes ago

Terraform Cloud Plan Output

```
Plan: 16 to add, 0 to change, 0 to destroy.
[Terraform Cloud Plan](https://app.terraform.io/app/dbarr-org/workspaces/aws-webapp-gha/runs/run-vT5dahcwnSa6G
```

danbarr merged commit `dff4961` into `main` 11 minutes ago

1 check passed

✓ terraform-cloud-speculative-run

Kubernetes Operator

Challenge Solution

Automate Terraform Cloud from Kubernetes

Organizations or teams that are heavily invested in Kubernetes want to automate the provisioning of infrastructure from the Kubernetes control plane.

Kubernetes Operator v2 (Beta)

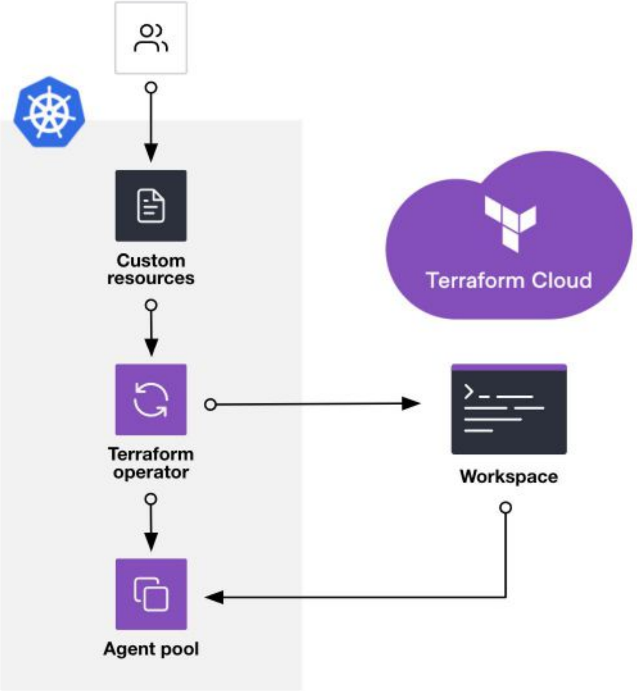
Challenge Solution

Manage Terraform Cloud and provision infrastructure using Kubernetes custom resources:

- `AgentPool` manages Terraform Cloud agent pools with auto-scaling support
- `Workspace` manages Terraform Cloud workspaces
- `Module` implements API-driven run workflows to provision infrastructure

Metrics for each controller are exposed in standard Prometheus format.

github.com/hashicorp/terraform-cloud-operator



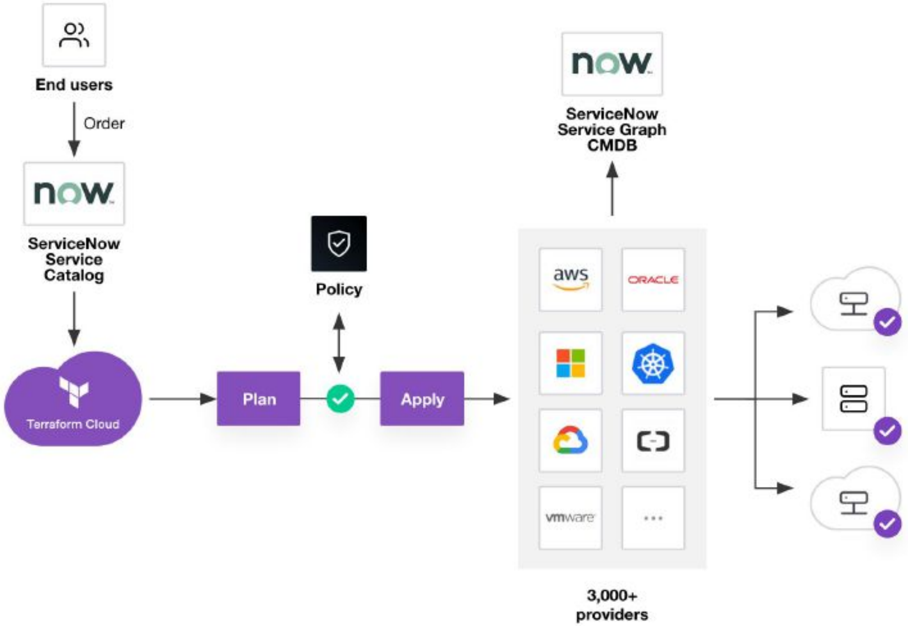
ServiceNow Integration

Challenge Solution

Service management integration

Organizations with ServiceNow want to enable self-service infrastructure for end users while still maintaining their infrastructure as code approach for multi-cloud compliance and management.

For ServiceNow Service Graph customers, the Configuration Management Database (CMDB) is the source of truth for infrastructure visibility.



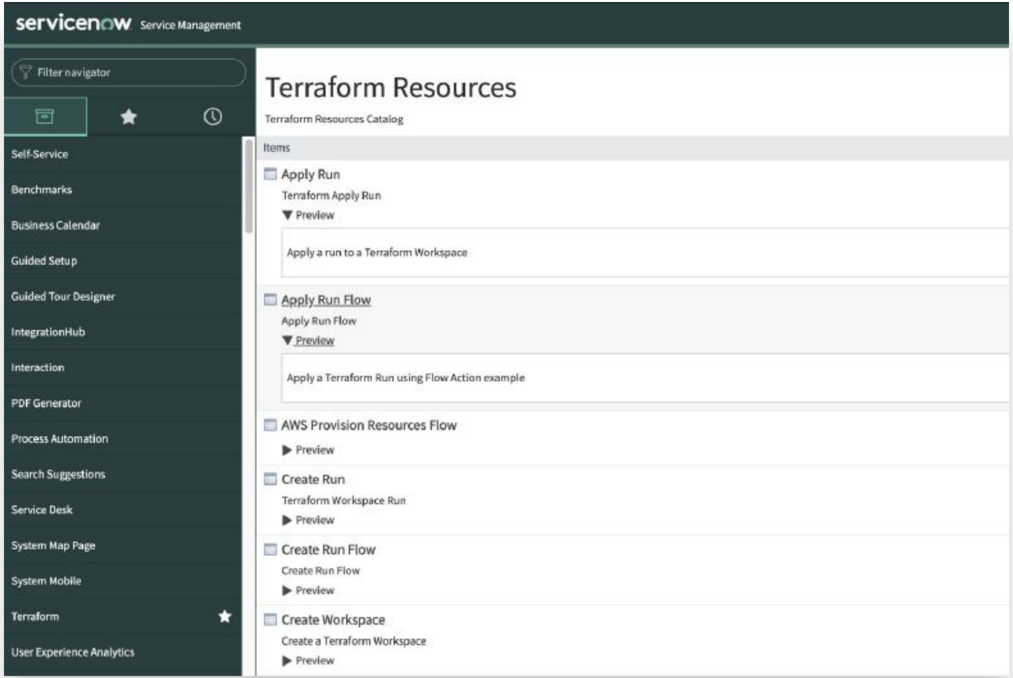
ServiceNow Integration – Service Catalog

Challenge Solution

Self-service infrastructure

Provision resources via the ServiceNow Service Catalog

- Familiar request workflow for users
- Customize via input variables
- Creates workspace and initiates a VCS-driven workflow
- Self-service destruction to clean up resources



Terraform Resources

Terraform Resources Catalog

Items

□ Apply Run
Terraform Apply Run
▼ Preview
Apply a run to a Terraform Workspace

□ Apply Run Flow
Apply Run Flow
▼ Preview
Apply a Terraform Run using Flow Action example

□ AWS Provision Resources Flow
▶ Preview

□ Create Run
Terraform Workspace Run
▶ Preview

□ Create Run Flow
Create Run Flow
▶ Preview

□ Create Workspace
Create a Terraform Workspace
▶ Preview

No-Code Provisioning

Challenge Solution

Getting up and running with Terraform

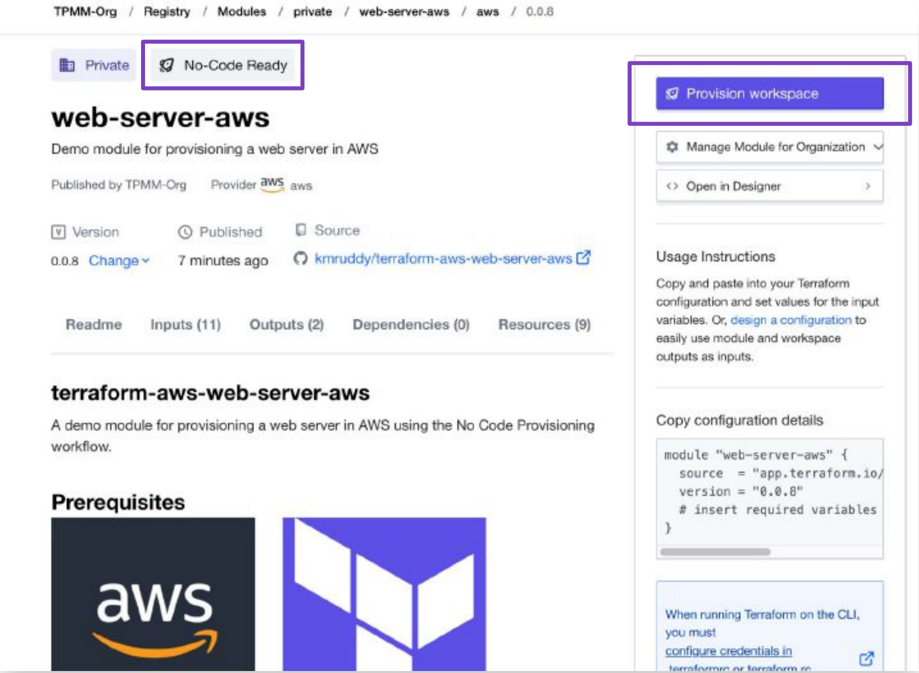
Skills shortage issues have been ranked as the top multi-cloud barrier for organizations. Provisioning something immediately useful with Terraform requires knowledge of infrastructure, as well as familiarity and comfort with HCL code, both of which create a barrier to adoption.

No-Code Provisioning

Challenge Solution

Deploy cloud resources using Terraform, without learning HCL

Teams can spend less time defining configurations and rebuilding the wheel, and spend more time building off the work of others and supporting the business.



web-server-aws

Demo module for provisioning a web server in AWS

Published by TPMM-Org Provider aws aws

Version 0.0.8 Change

Published 7 minutes ago

Source kmruddy/terraform-aws-web-server-aws

terraform-aws-web-server-aws

A demo module for provisioning a web server in AWS using the No Code Provisioning workflow.

Prerequisites

Usage Instructions

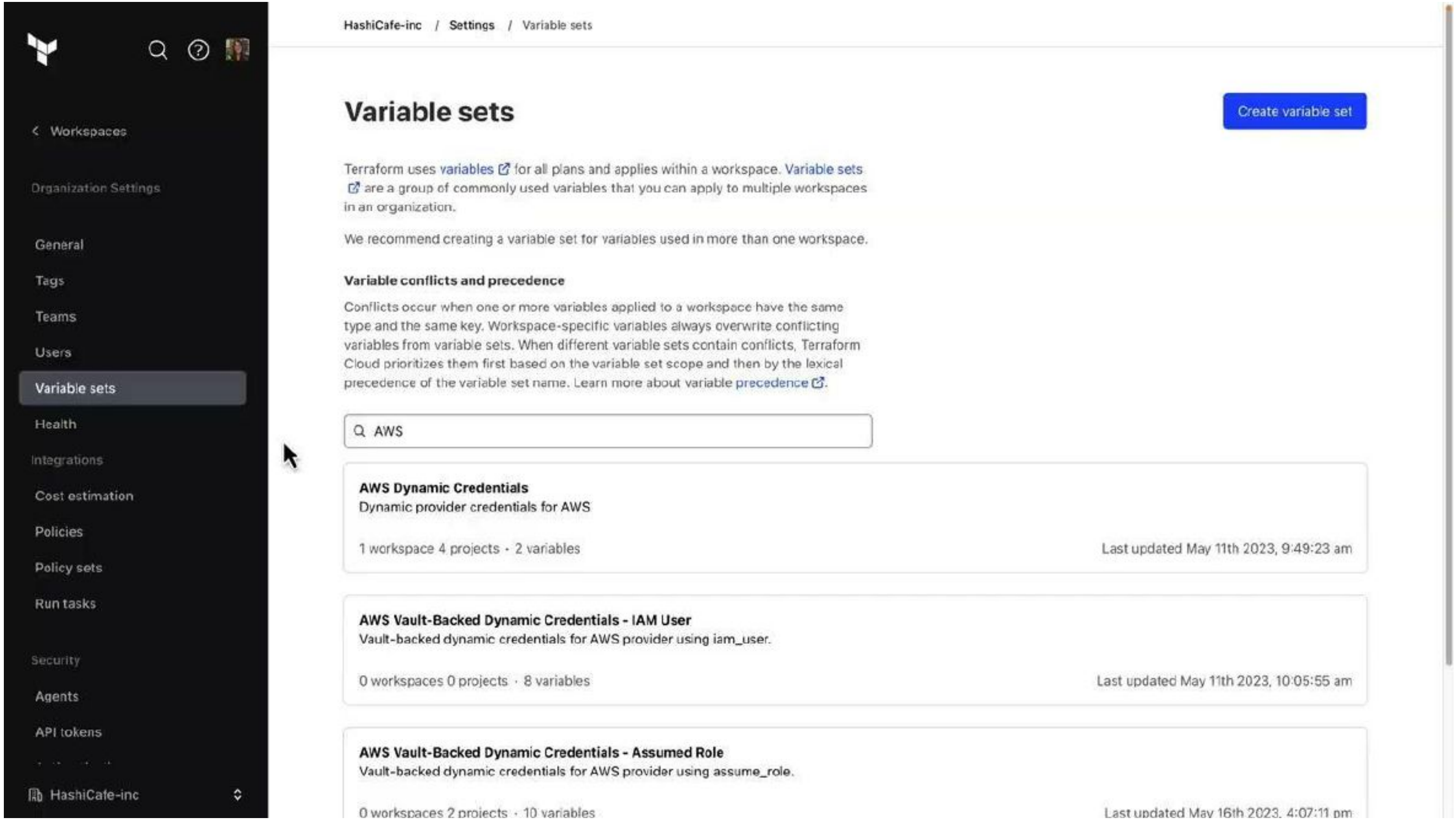
Copy and paste into your Terraform configuration and set values for the input variables. Or, design a configuration to easily use module and workspace outputs as inputs.

Copy configuration details

```
module "web-server-aws" {
  source  = "app.terraform.io/
  version = "0.0.8"
  # insert required variables
}
```

When running Terraform on the CLI, you must configure credentials in terraformrc or terraform.rc

No-Code & dynamic credentials



Variable sets

Terraform uses variables for all plans and applies within a workspace. Variable sets are a group of commonly used variables that you can apply to multiple workspaces in an organization.

We recommend creating a variable set for variables used in more than one workspace.

Variable conflicts and precedence

Conflicts occur when one or more variables applied to a workspace have the same type and the same key. Workspace-specific variables always overwrite conflicting variables from variable sets. When different variable sets contain conflicts, Terraform Cloud prioritizes them first based on the variable set scope and then by the lexical precedence of the variable set name. Learn more about variable precedence.

AWS

AWS Dynamic Credentials
Dynamic provider credentials for AWS

1 workspace 4 projects · 2 variables — Last updated May 11th 2023, 9:49:23 am

AWS Vault-Backed Dynamic Credentials - IAM User
Vault-backed dynamic credentials for AWS provider using iam_user.

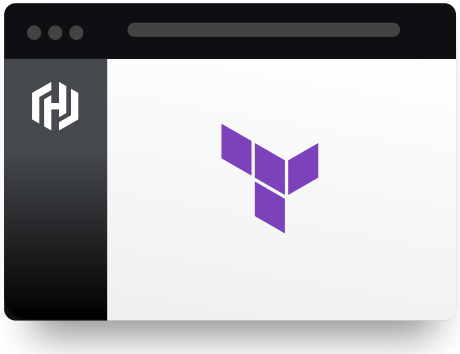
0 workspaces 0 projects · 8 variables — Last updated May 11th 2023, 10:05:55 am

AWS Vault-Backed Dynamic Credentials - Assumed Role
Vault-backed dynamic credentials for AWS provider using assume_role.

0 workspaces 2 projects · 10 variables — Last updated May 16th 2023, 4:07:11 pm

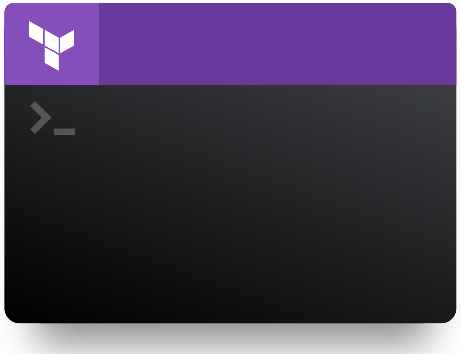
Solutions to Fit Your Needs

Fully Managed



Terraform Cloud

Self-managed



Terraform Enterprise

https://app.terraform.io/

Thank you

hello@hashicorp.com

HashiCorp

Unlock the Cloud Operating Model