

LIVE WEBINAR

# Codifying your cloud security at scale with Terraform Cloud and Bridgecrew

Wednesday, August 26, 8AM PT / 11AM ET / 3PM GMT



**bridgecrew**

Guy Eisenkot  
Co-founder, VP Product



**HashiCorp**

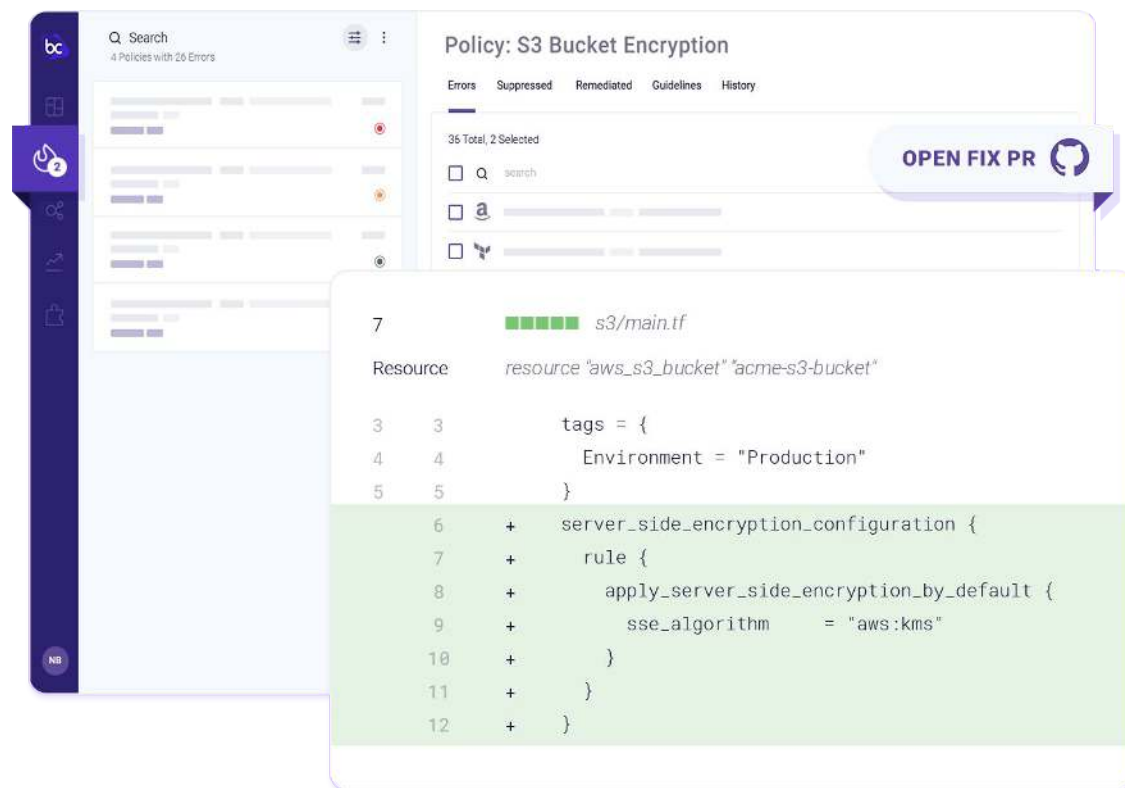
Kerim Satirli  
Senior Developer Advocate

# Agenda

- INTRODUCTIONS** Intro to Bridgecrew  
Intro to HashiCorp
- TERRAFORM** How Terraform works  
Terraform best practices and benefits
- IAC SECURITY** Developer security 101  
IaC security best practices  
Common IaC misconfigurations
- DEMO** Setting up Bridgecrew  
Workflow via GitHub Actions  
Terraform Cloud Runs
- Q&A** Leave your questions in the Zoom panel

# bridgecrew

# Bridgecrew: Automating and codifying cloud security



## FIND CLOUD MISCONFIGS

Both infrastructure-as-code and cloud resources



## FIX ISSUES IN CODE, WITH CODE

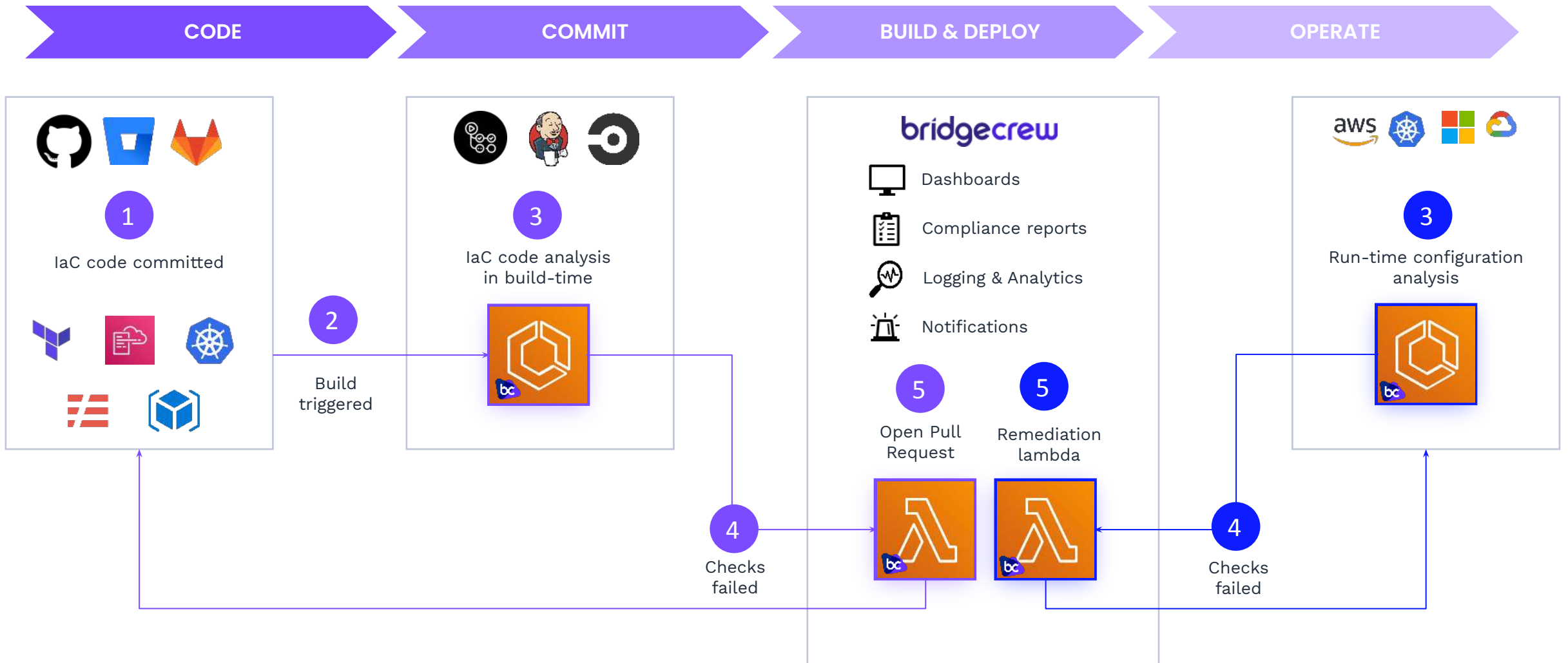
Merge-ready pull requests and automated playbooks



## PREVENT ISSUES FROM BEING DEPLOYED

Enforce policy-as-code in all config modules via CI/CD

# How Bridgecrew works





# HashiCorp

# Terraform – Benefits

## **CODIFY INFRASTRUCTURE**

Provision, manage, and version infrastructure and service components

## **INCREASE VISIBILITY**

Separate plan and apply steps for more predictable changes

## **CONSISTENT WORKFLOW**

Reproducible deployments across different environments and providers

# Terraform Cloud – Benefits

## COLLABORATIVE WORKFLOW

Work with your team to review and iterate on infrastructure

## AUTOMATED WORKFLOW

VCS connections and full API support for in-depth integrations

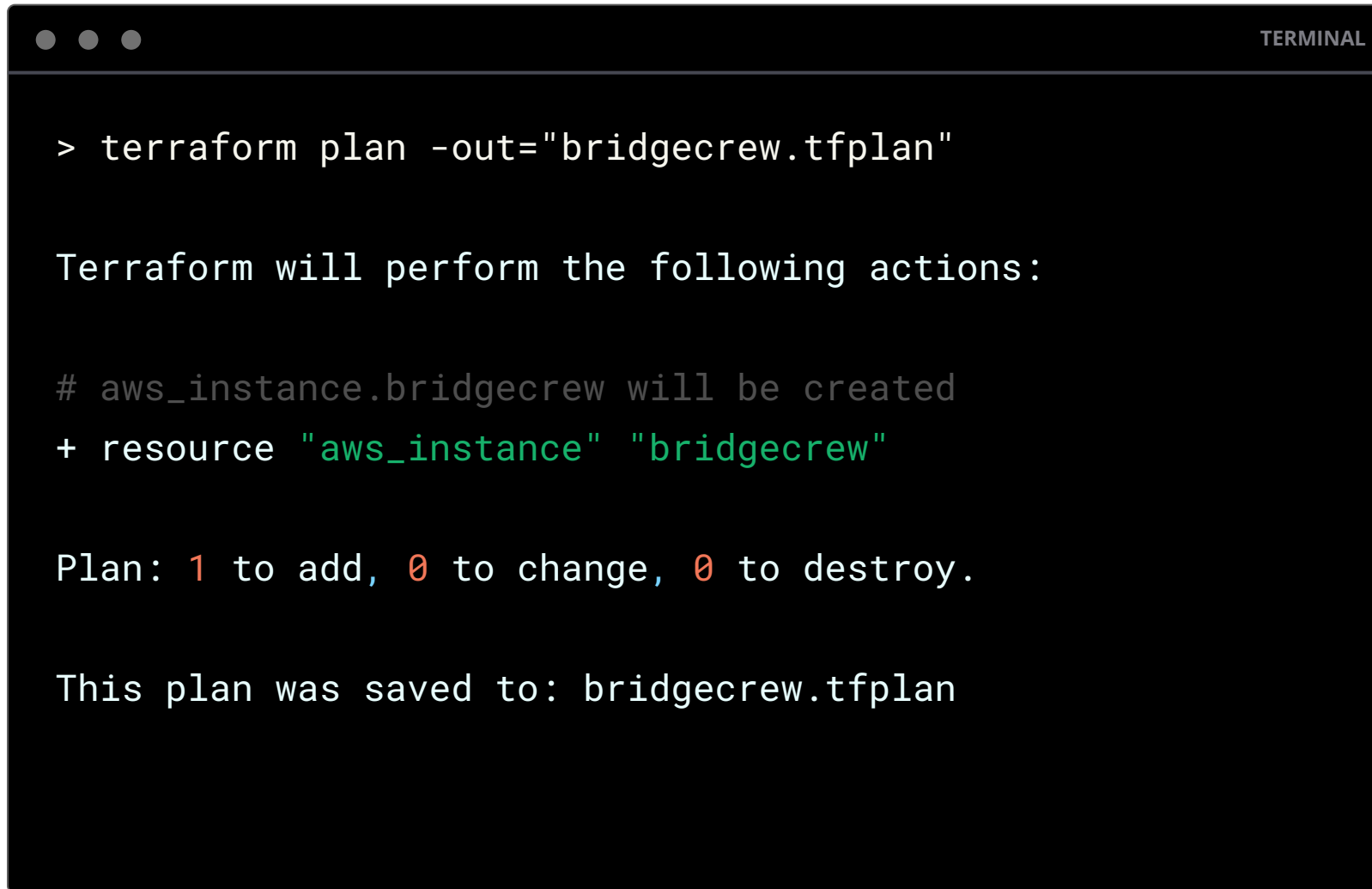


# Terraform – How it works



```
resource "aws_instance" "bridgecrew" {  
    ami            = var.ami_id  
    instance_type  = "t3.large"  
    availability_zone = "us-west-1"  
}
```

# Terraform – How it works

A terminal window with a dark background and light text. The title bar at the top right says "TERMINAL". The terminal shows the command to run a Terraform plan, followed by a summary of actions to be performed, a detailed plan for creating an AWS instance, and the final plan file name.

```
> terraform plan -out="bridgecrew.tfplan"

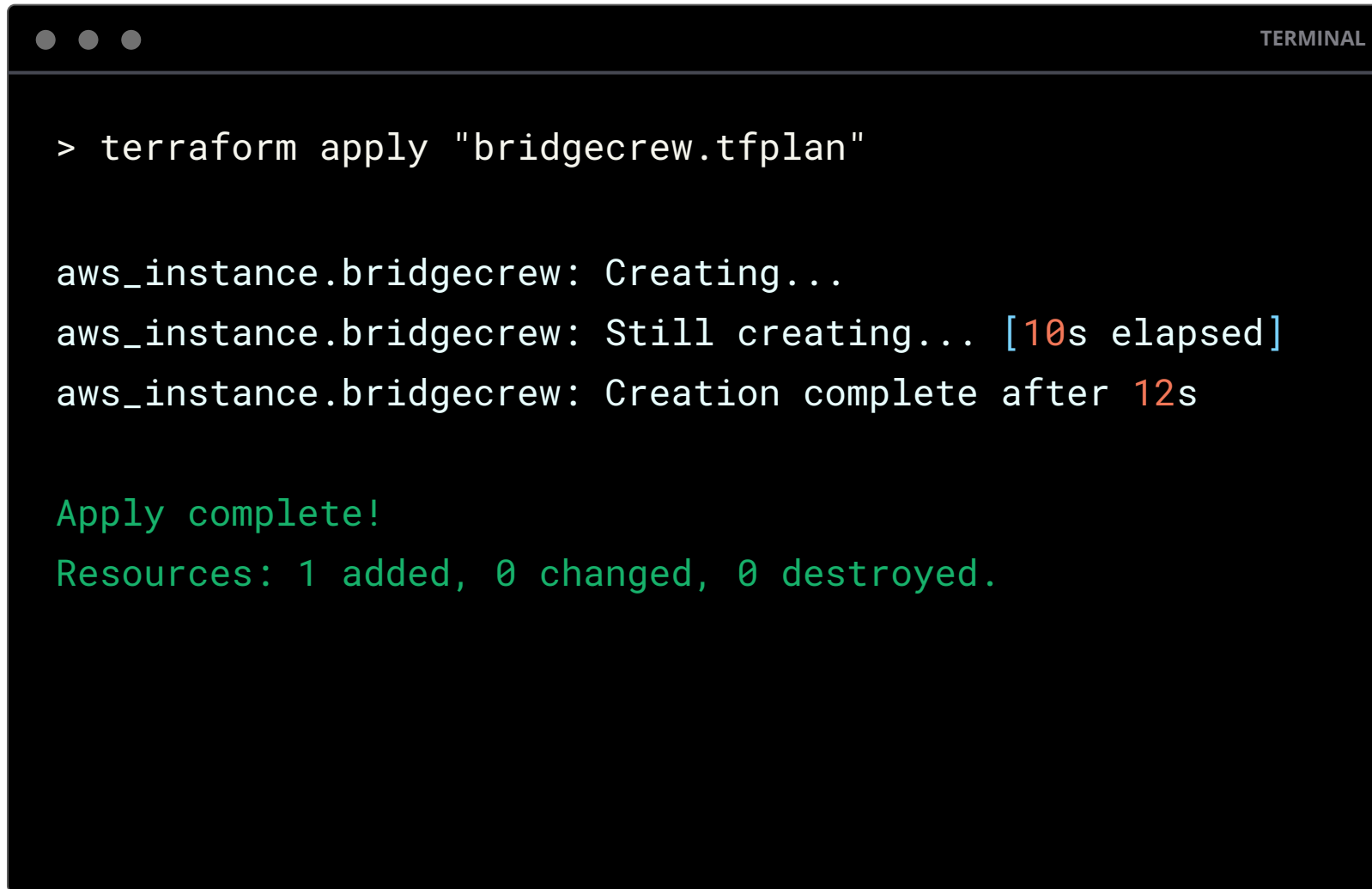
Terraform will perform the following actions:

# aws_instance.bridgecrew will be created
+ resource "aws_instance" "bridgecrew"

Plan: 1 to add, 0 to change, 0 to destroy.

This plan was saved to: bridgecrew.tfplan
```

# Terraform – How it works

A terminal window with a dark background and light gray window controls (three dots) in the top-left corner. The title bar in the top-right corner reads "TERMINAL". The terminal displays the output of a Terraform apply command. The first line is a prompt followed by the command. Subsequent lines show the progress of creating an AWS instance, including a progress bar and elapsed time. The final lines indicate the successful completion of the apply operation and the state of the resources.

```
> terraform apply "bridgecrew.tfplan"

aws_instance.bridgecrew: Creating...
aws_instance.bridgecrew: Still creating... [10s elapsed]
aws_instance.bridgecrew: Creation complete after 12s

Apply complete!
Resources: 1 added, 0 changed, 0 destroyed.
```

# Developer security 101



CODE

COMMIT

BUILD & DEPLOY

OPERATE

Change tracking

Secure images

Asset inventory

Secure IAM access

Code analysis

Package sourcing

Compliance assurance

Network segmentation

Reproducibility

Backups

Secret vaulting

Data protection

# Infrastructure-as-code security best practices

## AUTOMED GOVERNANCE

Enforce policies as early as possible through automation

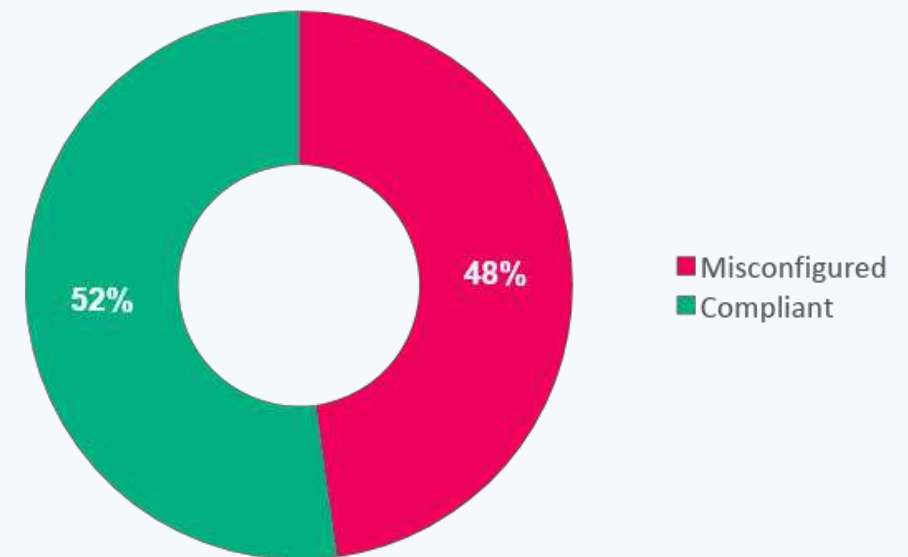
## CONSISTENT GOVERNANCE IN CODE

Policy-as-code provides a common language

## CONTINUOUS WORKFLOW

Embedded into code review processes and CI/CD workflow

### OPEN SOURCE TERRAFORM AWS MODULES



# Insecure code example

## Azure Kubernetes Engine cluster

```
1 resource azurerm_kubernetes_cluster "k8s_cluster" {
2   dns_prefix      = "terragoat-${var.environment}"
3   location        = var.location
4   name            = "terragoat-aks-${var.environment}"
5   resource_group_name = azurerm_resource_group.example.name
6   identity {
7     type = "SystemAssigned"
8   }
9   default_node_pool {
10    name     = "default"
11    vm_size  = "Standard_D2_v2"
12    node_count = 2
13  }
14  addon_profile {
15    oms_agent {
16      enabled = false
17    }
18    kube_dashboard {
19      enabled = true
20    }
21  }
22  role_based_access_control {
23    enabled = false
24  }
25 }
```

Ensure Kube Dashboard is disabled

Ensure RBAC is enabled on AKS clusters

Ensure AKS cluster has Network Policy configured

Ensure AKS has an API Server Authorized IP Ranges enabled

Ensure AKS logging to Azure Monitoring is Configured



[github.com/bridgecrewio/terragoat](https://github.com/bridgecrewio/terragoat)



DEMO

# Additional Resources

## BRIDGECREW CLOUD

[bridgecrew.cloud](https://bridgecrew.cloud)

## BRIDGECREW GITHUB ACTION

code: [github.com/bridgecrewio/bridgecrew-action](https://github.com/bridgecrewio/bridgecrew-action)

blog: [bridgecrew.io/blog/github-integrations](https://bridgecrew.io/blog/github-integrations)

## BRIDGECREW TERRAFORM TUTORIAL

blog: [bridgecrew.io/blog/terraform-tutorial](https://bridgecrew.io/blog/terraform-tutorial)

## TERRAFORM CLOUD

[hashi.co/tf-cloud-bc](https://hashi.co/tf-cloud-bc)