# Applying Policy as Code

**Ned Bellavance**
HashiCorp Ambassador

@ned1313 nedinthecloud.com

# Overview

**HashiCorp Sentinel**

**Application process**

**Globomantics scenario**

# HashiCorp Sentinel

**Policy as code**

**Version control**

**Native policy language**

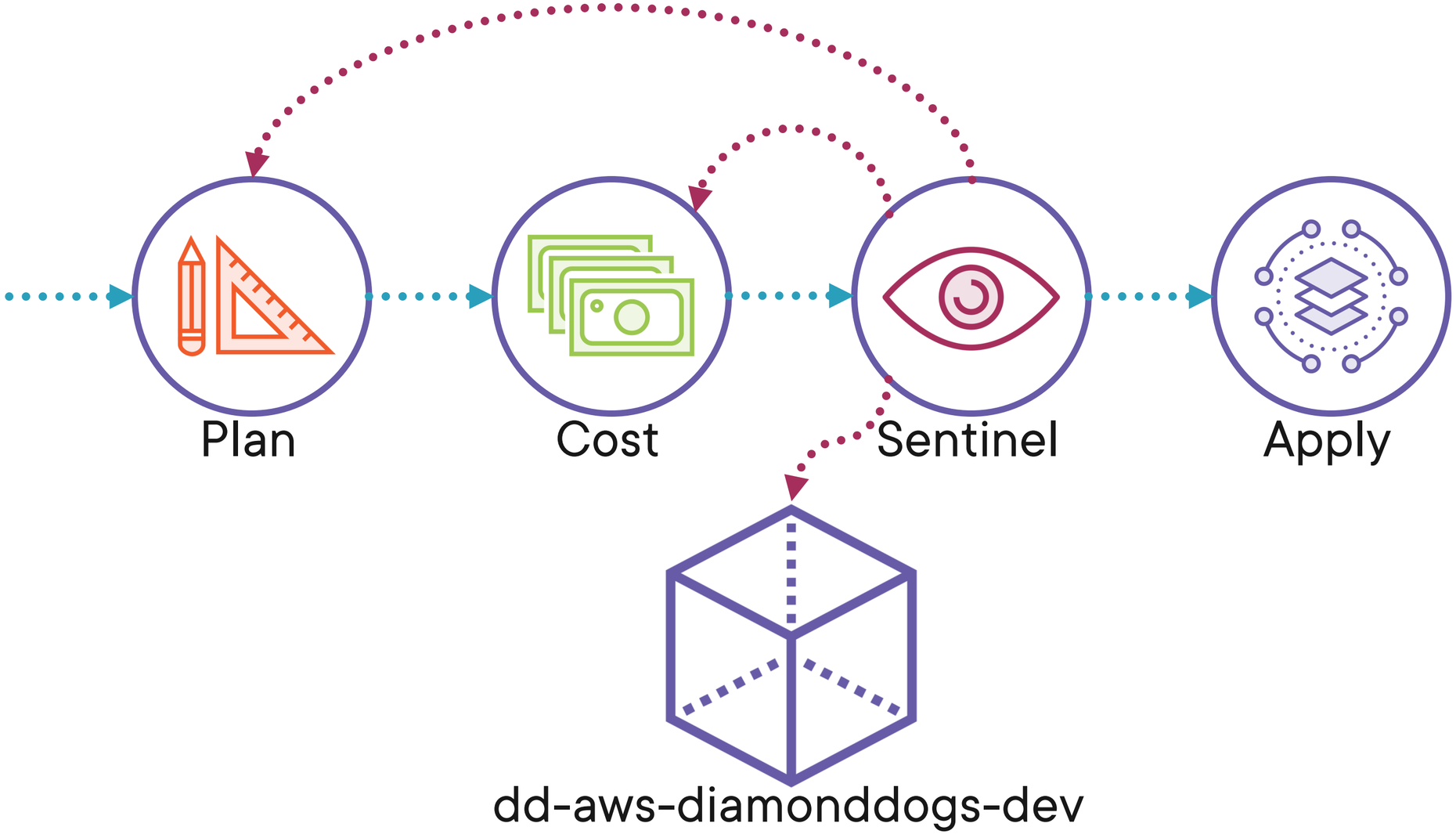
**Fine grained and conditional**

**Import data sources**

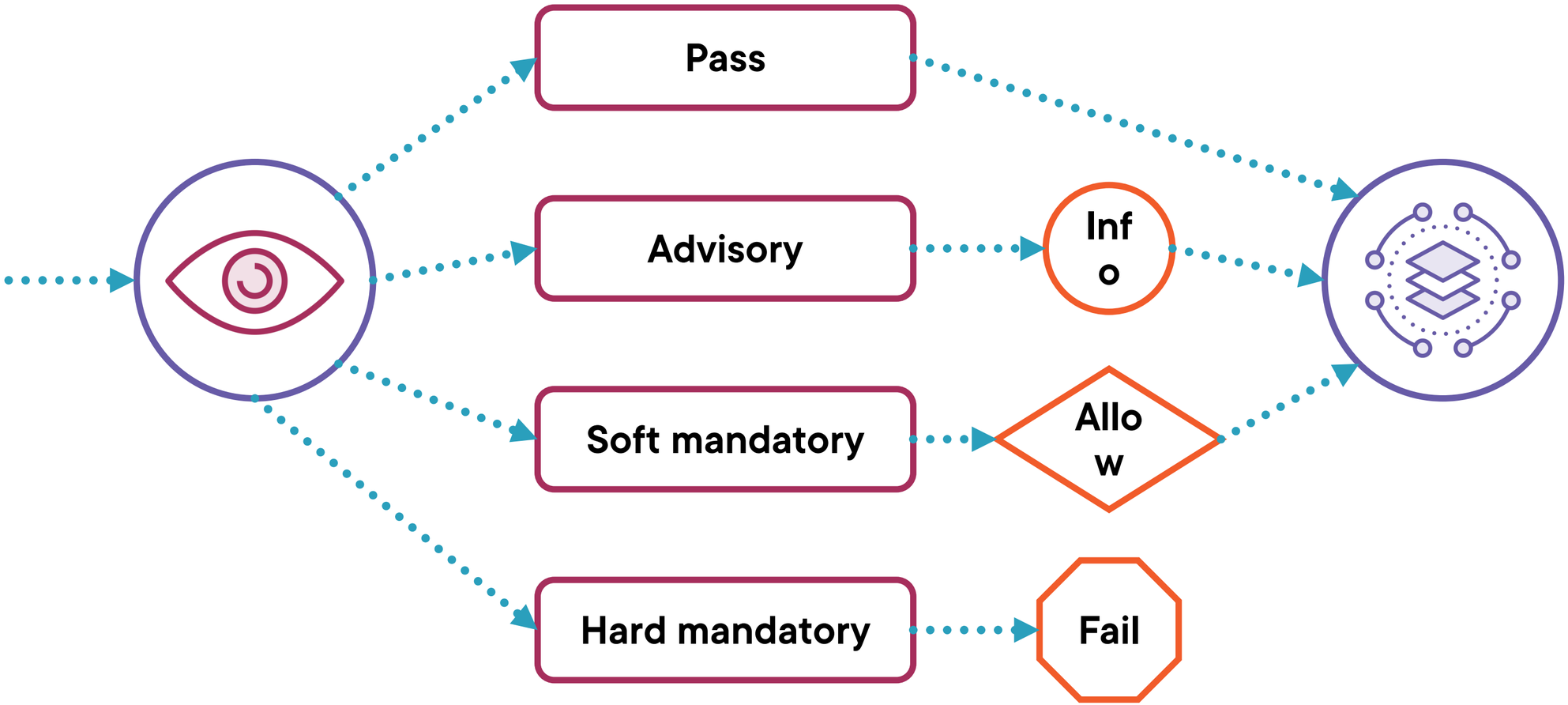
**Enforcement levels**

# Sentinel on Terraform Cloud

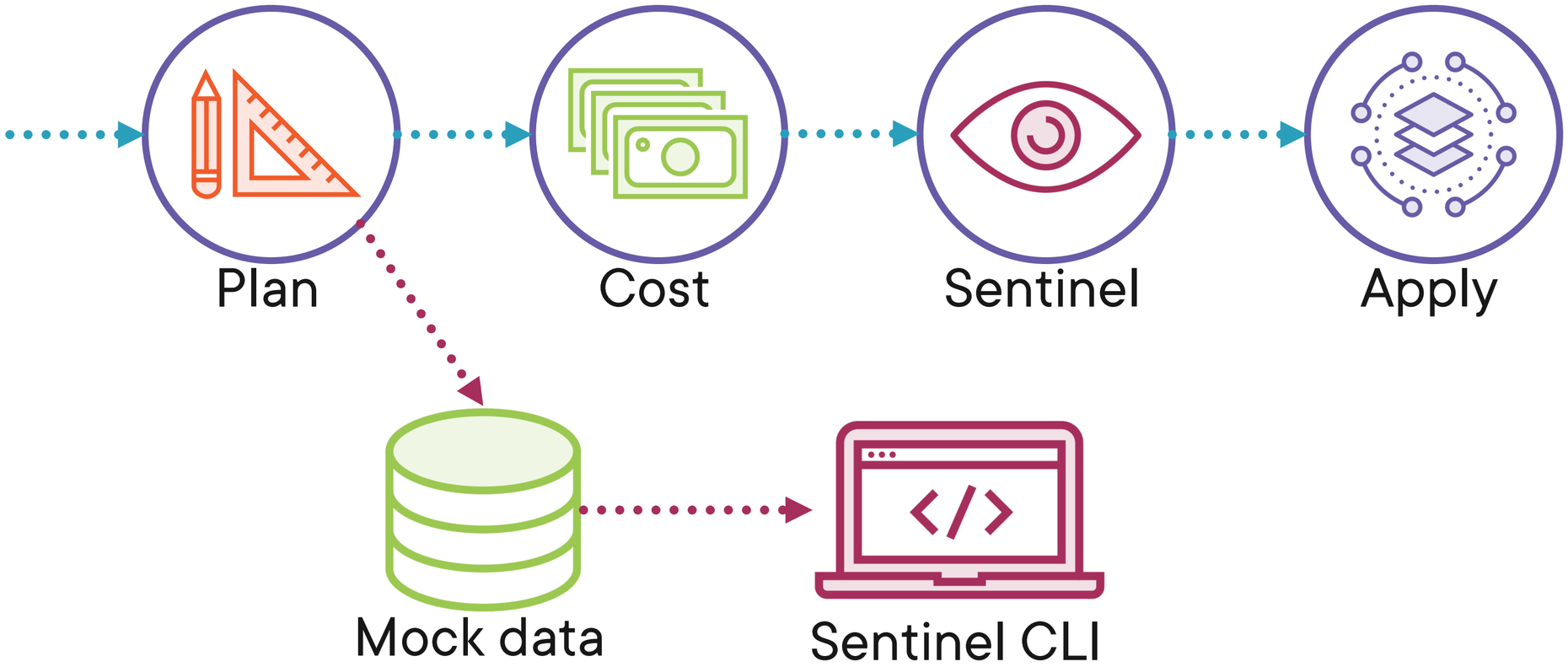
# Policy Evaluation



# Enforcement Actions



# Testing Sentinel



**instance-size.sentinel**

```
# Import tfplan

import "tfplan/v2" as tfplan


# Get all EC2 instances

ec2_instances = filter tfplan.resource_changes as _, rc {

  rc.type is "aws_instance" and rc.mode is "managed" }


# Test the instance size

main = rule { all ec2_instances as _, instance {

  instance.change.after.instance_type in ["t2.micro","t2.small"] }

}
```

## sentinel.hcl

```
policy "instance-type" {
  enforcement_level = "hard-mandatory"
}

policy "instance-tags" {
  source = "https://raw.githubusercontent.com/.../instance-tags.sentinel"
  enforcement_level = "advisory"
}
```

# Using Modules

## Leverage existing function libraries

**sentinel.hcl**

```
module "tfplan-functions" {

  source = "https://raw.githubusercontent/.../
  tfplan-functions.sentinel"

}
```

**instance-tags.sentinel**

```
import "tfplan-functions" as plan


ec2_instances =
  plan.find_resources("aws_instance")
```

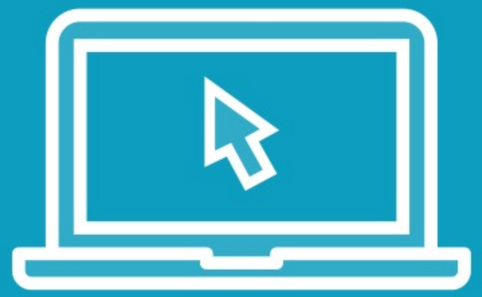
All workspaces

- Project and billable tags
- No SSH from everywhere

Development workspaces

- Smaller instance types
- No remote-exec or local-exec provisioners

# Demo



**Prepare Sentinel policies and policy sets**

**Apply and test global policy set**

**Apply and test development policy set**

# Summary

**Sentinel is policy as code**

**Managed in version control**

**Global or selective application**

**Evaluated during plan process**

**Enforcement level**

# Up Next:
# Operating Terraform Cloud for Teams