Security and the Language of Intent:

What's the answer?

@tracypholmes

Tracy P Holmes

(she/her)

Developer Advocate
@tracypholmes

Background

@tracypholmes

What gave you the idea for this talk?

Like...what even is the language of intent?!

- Infrastructure is hard because we don’t capture intent easily
- TF tries to make it easier to express intent with infrastructure
- Language translation is broken - there isn’t a language that can express security easily

A little bit about me...

That was me! Kinda...

@tracypholmes

We shouldn't have to lockpick our security to feel secure!

Commonly Asked Questions

from practitioners...

Can Terraform make the most secure db on Azure/ AWS EVAR (aka make a really secure configuration)?

Yep! If you configure it the right way.

Ummm ok. So, what is the right way smarty pants?

Ok, look. It’s pretty well known that managing secrets in Terraform can be a pain in the you know what. Plaintext can get committed and pushed to your VCS by mistake. Or, stuff ends up in your .tfstate.

@tracypholmes

A few things to work on so your database is secure

- Make sure your Secrets are secret.
- Make sure your subnet and ports are locked down
- Make sure privileges are virtually nil
- Encrypted data is encrypted
- Access Control Lists (ACL) and other access are configured correctly

```
Scenario: No publicly open ports
    Given I have AWS Security Group defined
    When it contains ingress
    Then it must not have tcp protocol and port
1024-65535 for 0.0.0.0/0
```

Straight from the docs...

In Azure, Databases in SQL Database are protected by firewalls. By default, all connections to the server and database are rejected. You can use portal to Set Allow access to Azure services to OFF for the most secure configuration. Then, create a reserved IP (classic deployment) for the resource that needs to connect, such as an Azure VM or cloud service, and only allow that IP address access through the firewall. If you're using the Resource Manager deployment model, a dedicated public IP address is required for each resource.

Now back to languages...

If I asked you for the best way to express

Front End

- JavaScript
- HTML/CSS
- Typescript

Data Manipulation

- R
- Python
- MATLAB

Backend

- Java
- Go
- Ruby

But if I ask you about Security?

@tracypholmes

Policy As Code

What is Policy as Code?

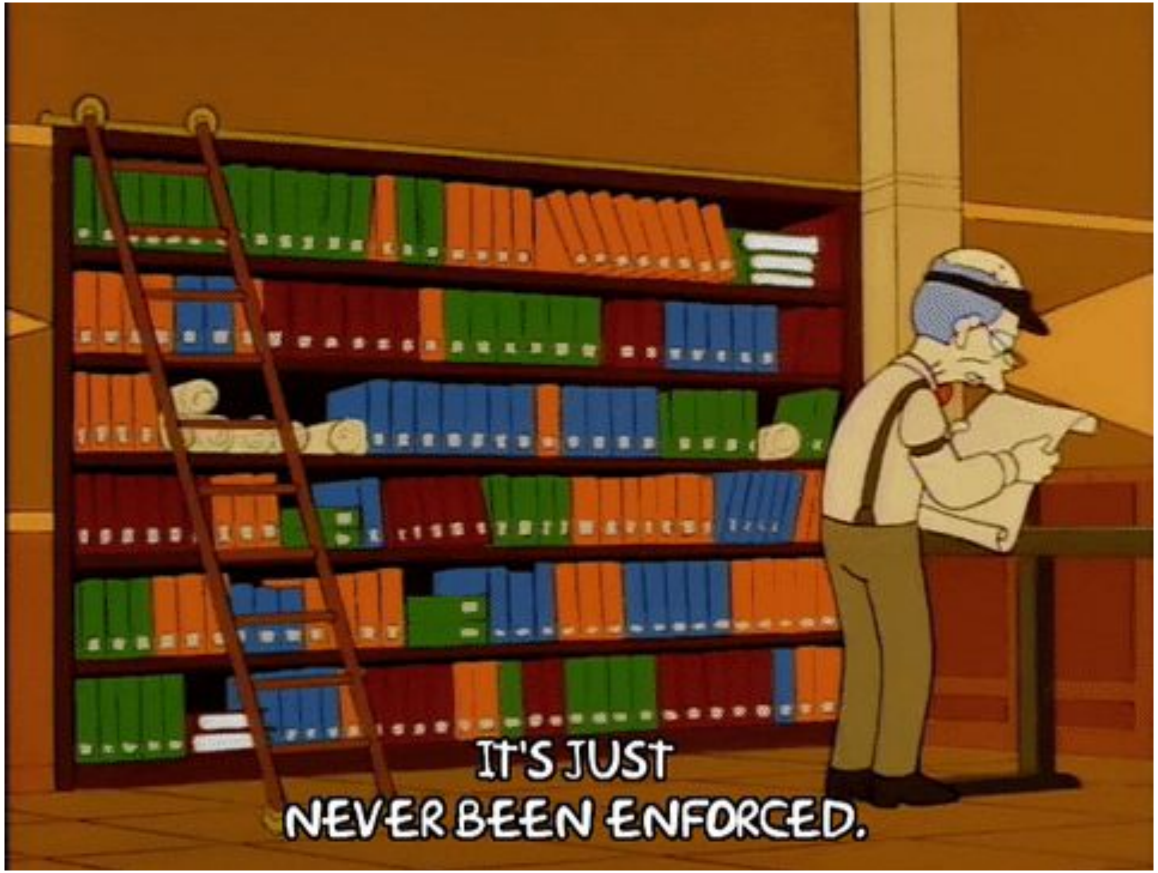
Policy as code is the idea of writing code in a high-level language to manage and automate policies. By representing policies as code in text files, proven software development best practices can be adopted such as version control, automated testing, and automated deployment.

Benefits of Policy as Code

1. Sandboxing
2. Codification
3. Version Control
4. Testing
5. Automation
6. Balance Developer Experience and Security

Enforcement

What to look for as an organization.



@tracypholmes

What is Sentinel?

Sentinel is a language and an embedded policy framework, which restricts Terraform actions to defined, allowed behaviors. It can be extended to use information from external sources.

```
import "time"

# Validate time is between 8 AM and 4 PM
valid_time = rule { time.now.hour >= 8 and time.now.hour < 16 }

# Validate day is M - Th
valid_day = rule {
    time.now.weekday_name in ["Monday", "Tuesday", "Wednesday", "Thursday"]
}

main = rule { valid_time and valid_day }
```

@tracypholmes

Why?!

Configuration Languages.

Configuration languages are formats such as JSON, YAML, XML, etc.

Configuration languages are good for static, declarative information. But they are not good for dynamic or logical rules.

Programming Languages.

The Sentinel language was designed with the goals of being non-programmer friendly while being programmer friendly.

Non-programmer friendly for non-programmers who may need the ability to enforce certain rules within a system. And, programmer friendly enough for constructs such as conditionals, loops, and functions for complex policies that a programmer may be writing.

@tracypholmes

Some more about Sentinel

Quirks, if you will.

Self proclaimed Issue #1.

If you don't have the right password links or a conformant password - Terraform will plan it, but it will not actually apply it.

@tracypholmes

Some more about Sentinel

Quirks, if you will.

Self proclaimed Issue #2.

Sentinel has a CLI to enable you to develop policies. BUT - Sentinel policy enforcement is only available in the Terraform Cloud or Terraform Enterprise.

@tracypholmes

```
database_firewall_rules = filter resources as _, v { v.type is "azurerm_postgresql_firewall_rule" }
database_servers = filter resources as _, v { v.type is "azurerm_postgresql_server" }

database_only_has_non_permissive_firewall_rules = rule {
	all database_firewall_rules as firewall_rule {
		firewall_rule.values.start_ip_address is not "0.0.0.0" and firewall_rule.values.end_ip_address is not
"255.255.255.255"
	}
}

database_has_conformant_password_length = rule {
	all database_servers as database_server {
		length(database_server.values.administrator_login_password) > 8 and
length(database_server.values.administrator_login_password) < 128
	}
}

main = rule {
	database_only_has_non_permissive_firewall_rules and
	database_has_conformant_password_length
}
```

Conclusion

@tracypholmes

Resources

1. Sentinel Docs - https://docs.hashicorp.com/sentinel
2. Vault Provider - https://learn.hashicorp.com/tutorials/terraform/secrets-vault
3. Compliance testing with Terraform and Azure - https://docs.microsoft.com/en-us/azure/developer/terraform/best-practices-compliance-testing
4. Terraform Compliance - https://terraform-compliance.com/pages/Examples/
5. Sentinel Policy Guide - https://www.hashicorp.com/resources/writing-and-testing-sentinel-policies-for-terraform

Thank You!

@tracypholmes