



Consul Integrations

August 2022

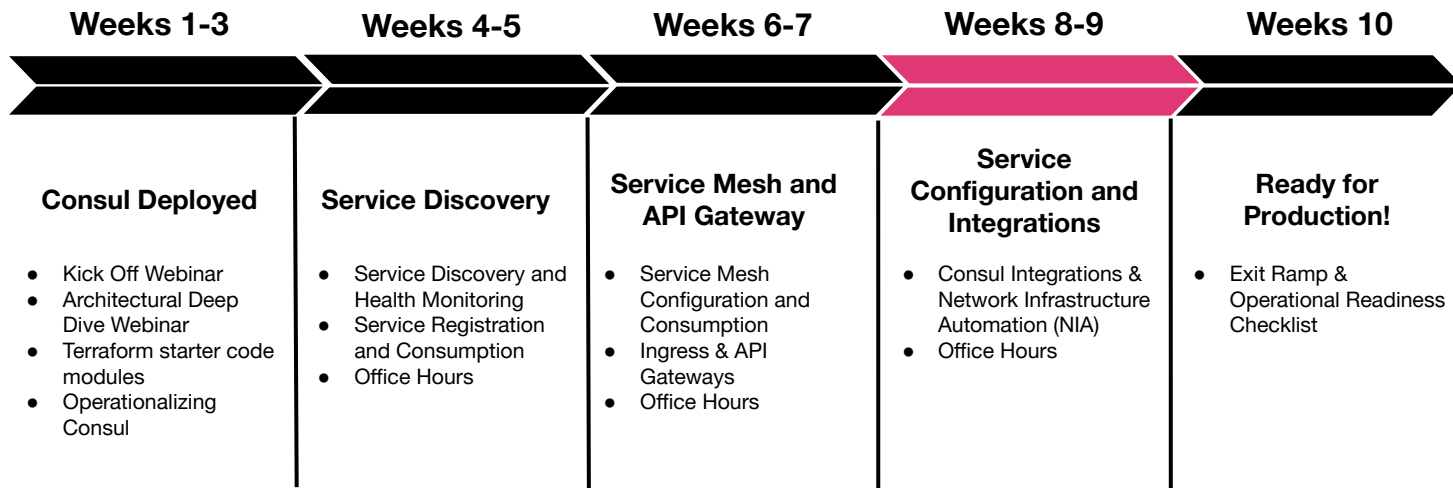
Copyright © 2021 HashiCorp



Agenda

1. What is NIA
2. What is CTS
3. NIA & CTS Together

Consul Enterprise Path to Production



01

Network Infrastructure Automation (NIA)



Current Challenges

What enterprises are currently facing



Slow Manual Processes

Ticketing systems are not allowing networks to move as fast as developers want



Increased Costs

Organizations want to find a way to optimize and increase efficiency with their existing and new networking infrastructure



Increased Risk

Higher risk in network outage from misconfigurations across multiple networking infrastructure devices

What is NIA?



- Network Infrastructure Automation (NIA) is the overarching concept, and Consul-Terraform-Sync (CTS) is the primary technology used within NIA
- NIA is the idea that changes in our deployment should automatically trigger changes to our network infrastructure devices to reflect the deployment and drive traffic to them
- In order to automate the NIA workflow, a tool has to watch for changes and trigger a handler. The primary tool for watching for changes is CTS.



NIA Example

```
resource "panos_dag_tags" "example" {  
  for_each = var.services  
  # vsys name to associate IP address and TAG  
  vsys = var.vsys_name  
  register {  
    # Service or node address to associate the TAG  
    ip = each.value.address == "" ? each.value.node_address : each.value.address  
    # TAGs based on service name  
    tags = [each.value.name]  
  }  
}
```

02

Consul Terraform Sync

Consul Terraform Sync

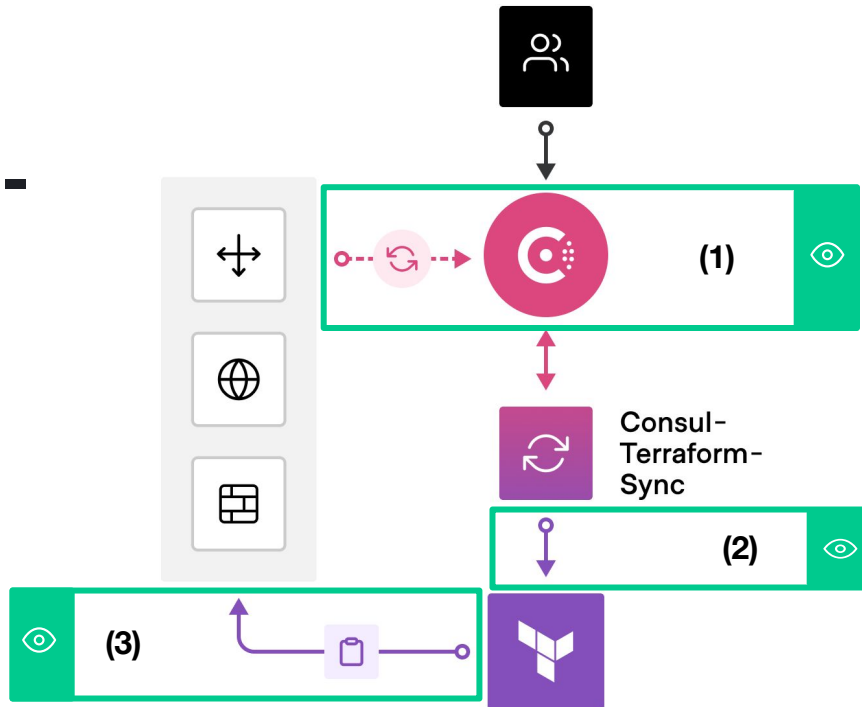


- Consul-Terraform-Sync (CTS) is a utility that watches for changes in Consul, and triggers actions in Terraform in near real time
- Terraform is used as the underlying automation tool
- CTS leverages the rich ecosystem of providers in Terraform, an modules in the Private Module Registry, and well as governance via Sentinel policies



Consul-Terraform-Sync (CTS)

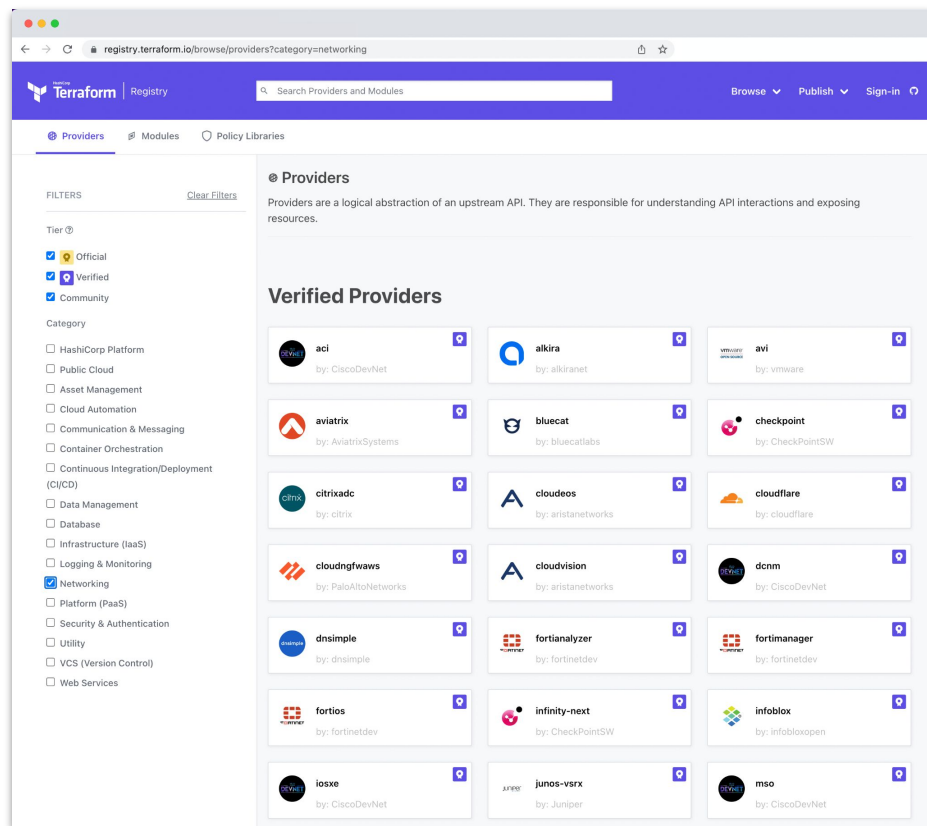
CTS generates new Terraform manifest and initiates a Terraform run to apply changes.



Leverage the Terraform Ecosystem



- Modules in the Terraform Private Module Registry (PMR) can be leveraged for automation with CTS
- Sentinel in Terraform Cloud/Enterprise will provide the governance necessary to make sure that the automation is safe
- There are many providers to us in automation tasks



CTS Installation Best Practices

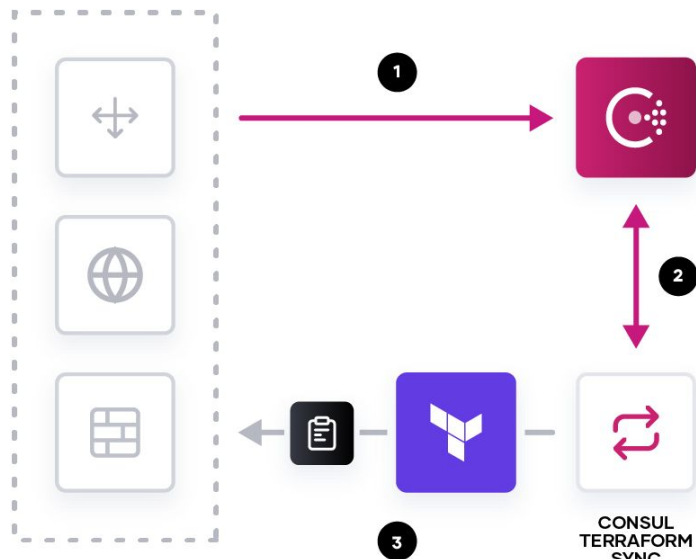


- The server running CTS should be a dedicated instance
- Consul Agent **should be** installed onto the CTS server, if not installed CTS **must** be able to reach a Consul Agent
- CTS should run under a **dedicated user** account
- The working and configuration directories for CTS require full write permissions
- When CTS is deployed in conjunction with Consul admin partitions, each admin partition requires a CTS instance

Declarative, Service & Workflow Driven Network Automation

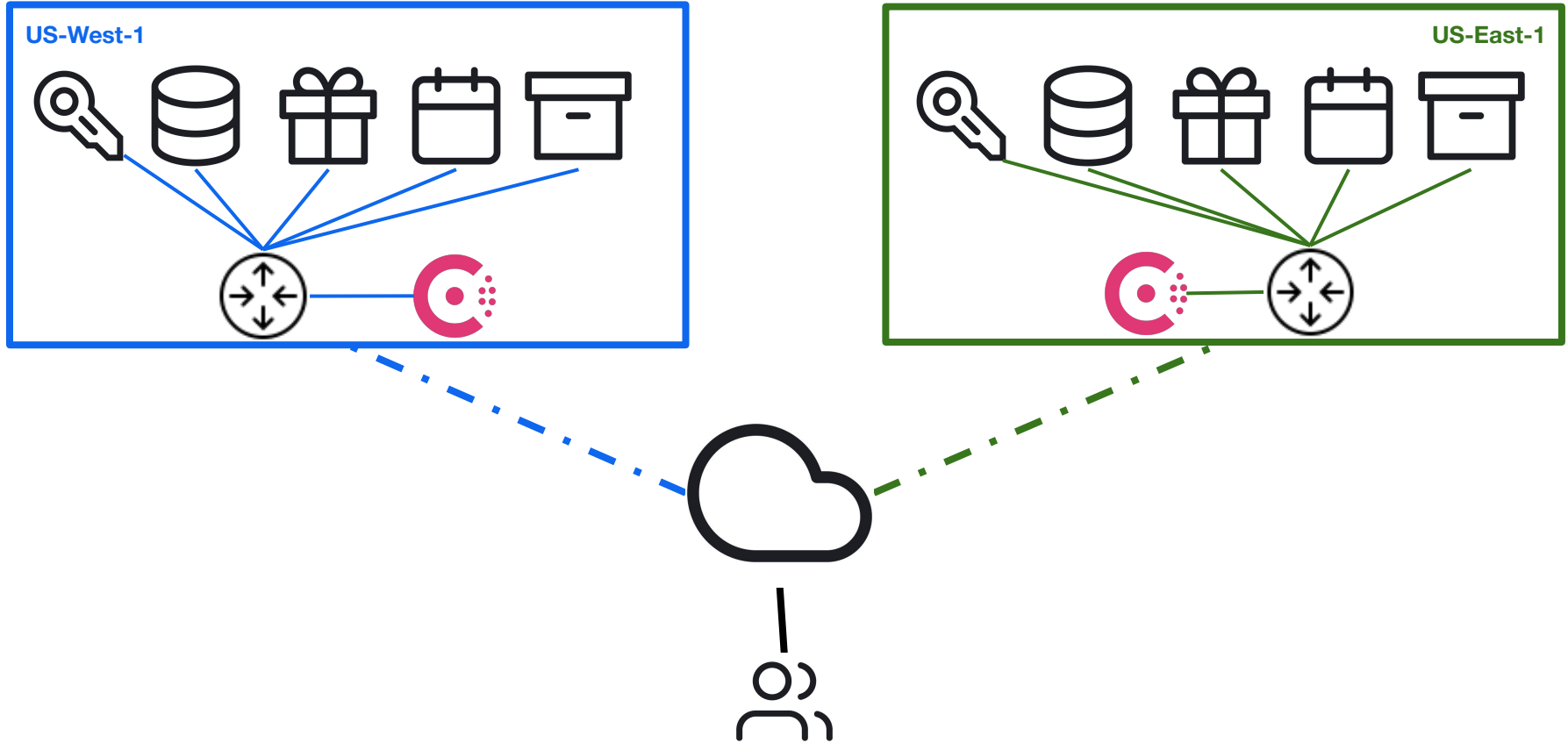


- Automate manual tasks across multiple network devices
- Lightweight installation
- Robust ecosystem

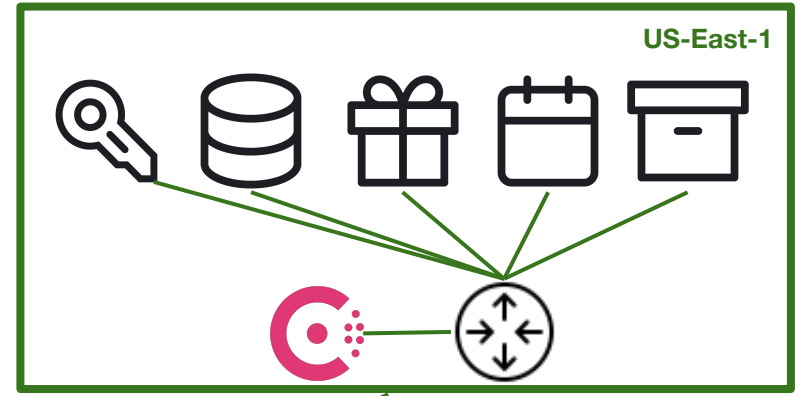
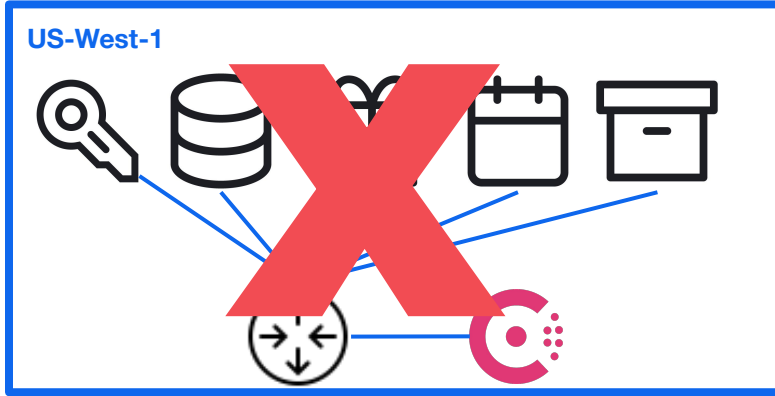


- 1) Consul catalog updated with new service registrations/changes
- 2) Consul-Terraform-Sync pulls service info from Consul catalog
- 3) Consul-Terraform-Sync generates new Terraform manifest and initiates a Terraform run to apply changes

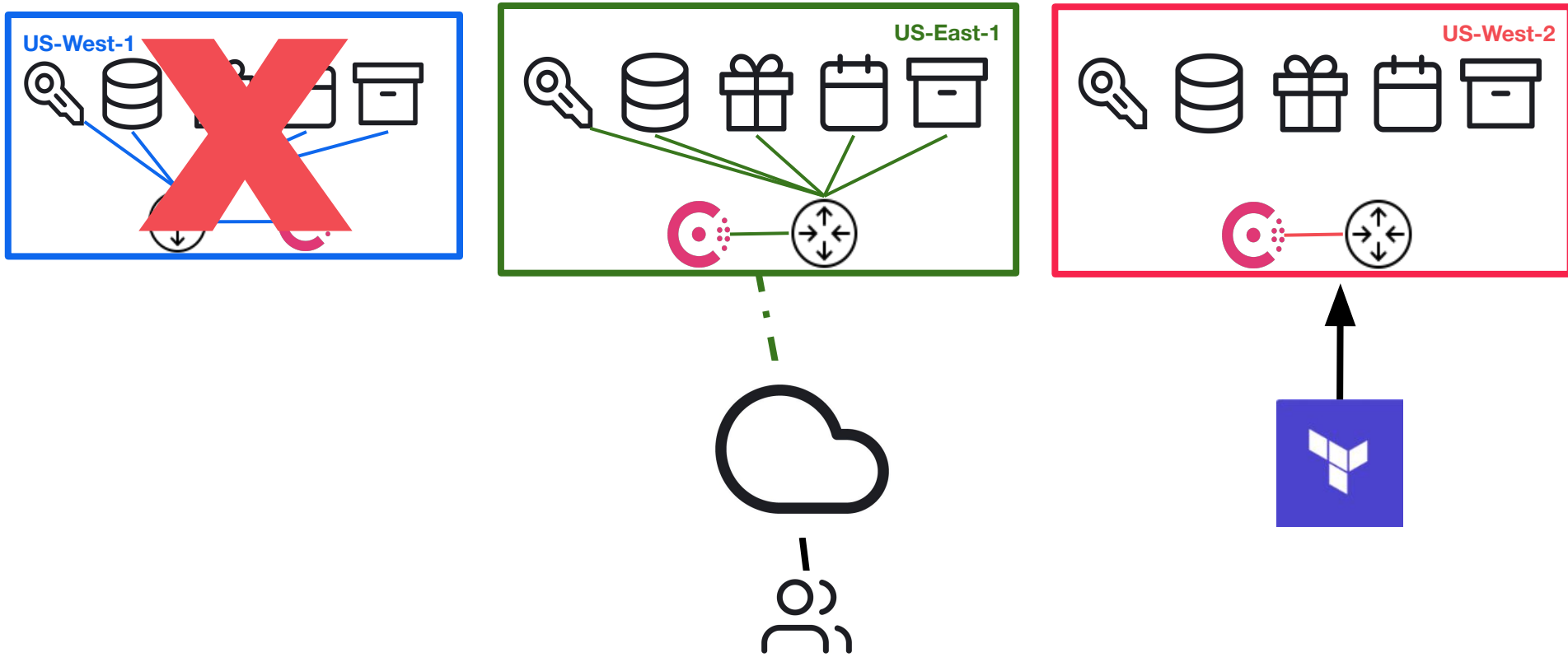
A practical example: “The Dreaded X”



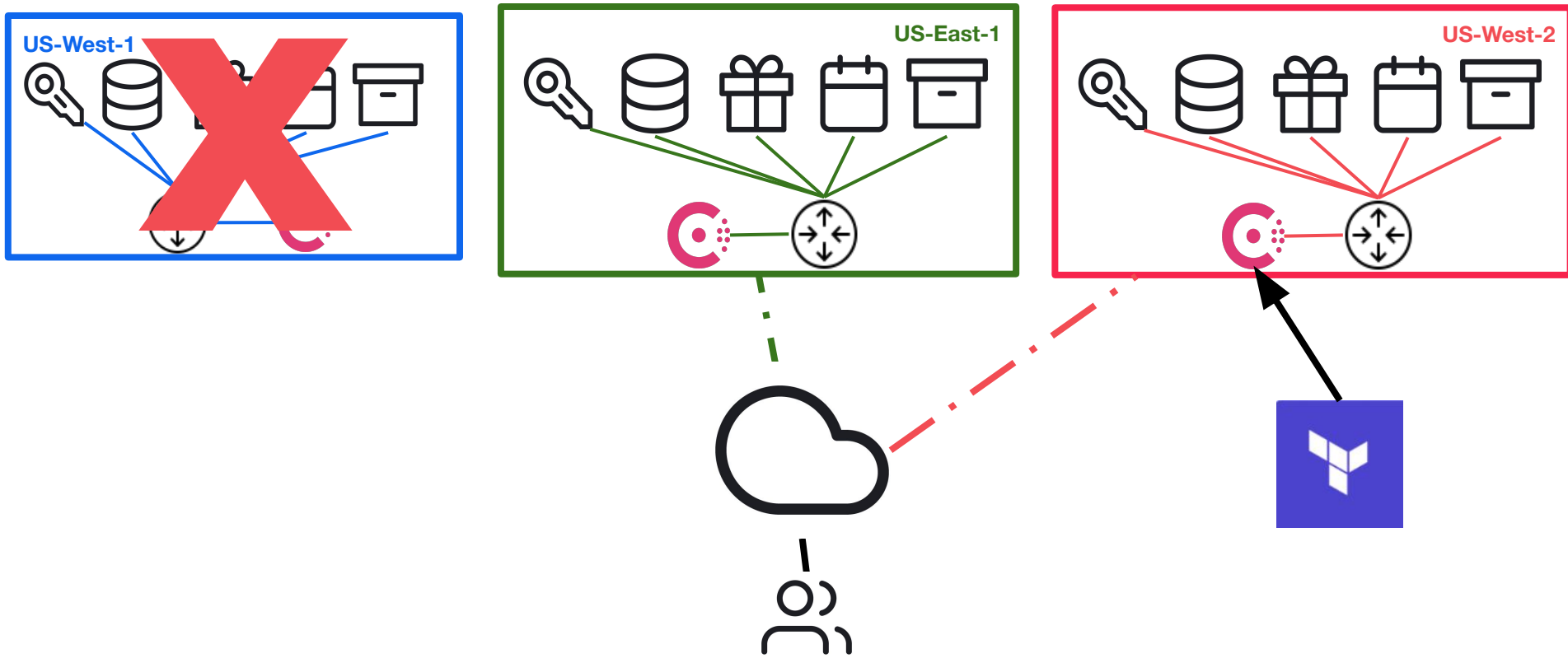
A practical example: “The Dreaded X”



A practical example: “The Dreaded X”



A practical example: “The Dreaded X”



03

Combining NIA & CTS

Some Things Go Well Together



Just like a Peanut Butter and Jelly Sandwich, NIA and CTS go well together and provide a full automation solution

CTS Configuration Example

CODE EDITOR

```
log_level    = "INFO"
working_dir  = "sync-tasks"
port        = 8558
id          = "cts-01"

syslog {
  enabled = true
  facility = "local2"
  name = "CTS"
}

buffer_period {
  enabled = true
  min     = "5s"
  max     = "20s"
}

tls {
  enabled = true
  cert    = "/path/to/cert.pem"
  key     = "/path/to/key.pem"
  verify_incoming = true
  ca_cert = "/path/to/ca.pem"
}

driver "terraform" {
  log          = false
  persist_log  = false
  backend "consul" {
    gzip = true
  }
}

task {
  name          = "cts-example"
  description   = "Example task"
  module        = "findkim/print/cts"
  version       = "0.1.0"
  condition "services" {
    names = ["web", "api"]
  }
}
```



CTS License Block

If no license block is specified, then CTS will try to auto retrieve a license from Consul.

```
license {  
  path = "path/to/license.lic"  
  auto_retrieval {  
    enabled = true  
  }  
}
```

CODE EDITOR



CTS: Terraform-cloud Network Driver

- Network Drivers are a set of tools that propagate the network infrastructure changes that CTS is pushing
- Only one network driver can be used per CTS deployment.
- The two primary network drivers used by CTS are the terraform and terraform-cloud drivers

```
driver "terraform-cloud" {
  hostname      = "https://app.terraform.io"
  organization  = "my-org"
  token        = "<TEAM_TOKEN>"

  workspaces {
    tags          = ["source:cts"]
    tags_allowlist = []
    tags_denylist  = []
  }

  required_providers {
    myprovider = {
      source = "namespace/myprovider"
      version = "1.3.0"
    }
  }
}
```



CTS: Global Configs

- “**log_level**” sets the logging level for the CTS process
- “**working_dir**” specifies working directory for CTS artifacts
- “**port**” sets the port number for API requests
- “**id**” is the CTS instance
- The “**syslog**” configuration block enables the CTS process to log to syslog

CODE EDITOR

```
log_level    = "INFO"
working_dir  = "sync-tasks"
port        = 8558
id          = "cts-server01"

syslog {
  enabled = true
  facility = "local2"
  name = "CTS"
}
```



CTS: Global Configs

- The “**buffer_period**” block sets a timer to prevent a flapping change from instrumenting continuous changes
- The “**tls**” configuration block configures tls on the CTA API

```
buffer_period {  
  enabled = true  
  min     = "5s"  
  max     = "20s"  
}  
  
tls {  
  enabled = true  
  cert    = "/path/to/cert.pem"  
  key     = "/path/to/key.pem"  
  verify_incoming = true  
  ca_cert = "/path/to/ca.pem"  
}
```




CTS: Consul Configuration

- The Consul Agent can be run either on the host that is running CTS or on a separate host
- “**transport**” sets configuration for connecting to Consul

```
consul {  
  address = "consul.example.com"  
  auth {}  
  tls {}  
  token = null  
  
  transport {}  
  service_registration {  
    service_name = "cts"  
    address = "172.22.0.2"  
    default_check {  
      address = "http://172.22.0.2:8558"  
    }  
  }  
}
```

CTS ACL Requirements



Policy	Resources
service:read	Any services monitored by tasks
node:read	Any nodes hosting services monitored by tasks
keys:read	Any Consul KV pairs monitored by tasks
namespace:read	Any namespaces for resources monitored by tasks. This is required for enterprise consul.
service:write	The CTS service when service registration is enabled
keys:write	Consul-terraform-sync - only required when using Consul as the Terraform backend



CTS: Task Block

The task block defines the automation task and the conditions that need to be met in order for the automation to be processed

```
task {  
  name          = "cts-example"  
  description    = "Example task"  
  module        = "findkim/print/cts"  
  version       = "0.1.0"  
  condition "services" {  
  
    names = ["web", "api"]  
  
  }  
}
```

CODE EDITOR

```
task {  
  name      = "catalog_service_condition_task"  
  module    = "path/to/catalog-services-module"  
  providers = ["my-provider"]  
  
  condition "catalog-services" {  
    datacenter      = "dc1"  
    regex           = "web.*"  
    use_as_module_input = false  
  }  
  module_input "services" {  
    names = ["web-api"]  
    datacenter = "dc2"  
  }  
}
```



Condition Statements: Catalog Services

The catalog-services condition statement triggers an automation on initial registration/de-registration

```
task {  
  name      = "consul_kv_condition_task"  
  description = "execute on changes to Consul KV entry"  
  module     = "path/to/consul-kv-module"  
  providers  = ["my-provider"  
  condition "consul-kv" {  
    path          = "my-key"  
    recurse       = true  
    datacenter    = "dc1"  
    namespace     = "default"  
    use_as_module_input = true  
  }  
}  
}
```



Condition Statements: Consul KV

The consul KV condition statement is a trigger that executes on a change to a KV value

```
CODE EDITOR

task {
  name          = "scheduled_task"
  description    = "execute every Monday using
service information from web and db"
  module        = "path/to/module"

  condition "schedule" {
    cron = "* * * * Mon"
  }

  module_input "services" {
    names = ["web" "db"]
  }
}
```



Condition Statements: Scheduled Conditions

A scheduled condition is similar to a cron job

CTS Configuration Elements



Global Configurations

- log_level
- syslog
- buffer_period
- tls
- consul
- transport

License

Network Driver

- driver
- backend
- required_providers

Task Block

- task
- condition(s)
- module_input

```
CODE
EDITOR

log_level  = "INFO"
working_dir = "sync-tasks"
port       = 8558
id         = "cts-01"

syslog {
  enabled = true
  facility = "local2"
  name    = "CTS"
}

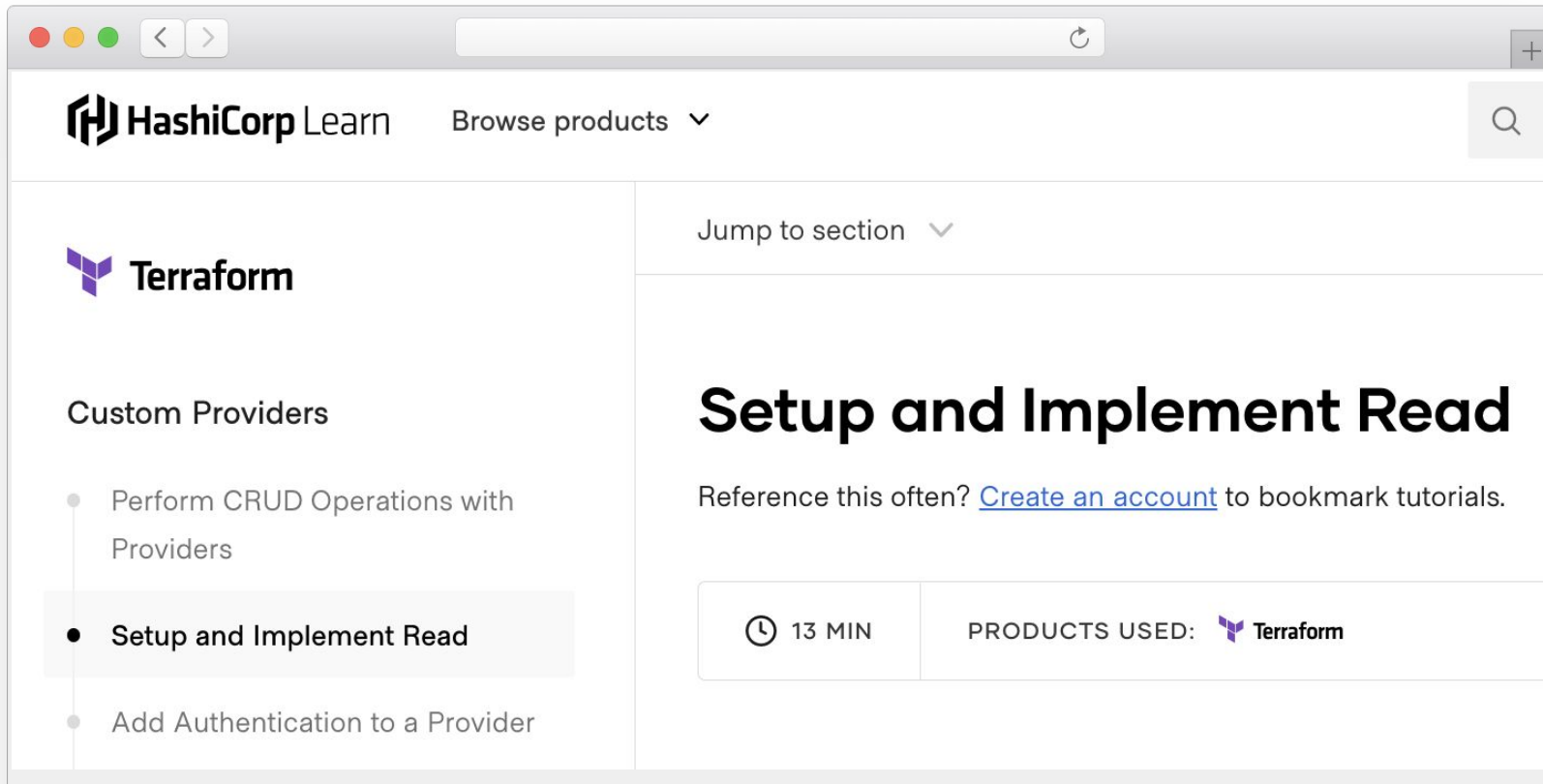
buffer_period {
  enabled = true
  min     = "5s"
  max     = "20s"
}

tls {
  enabled = true
  cert    = "/path/to/cert.pem"
  key     = "/path/to/key.pem"
  verify_incoming = true
  ca_cert = "/path/to/ca.pem"
}

driver "terraform" {
  log          = false
  persist_log  = false
  backend "consul" {
    gzip = true
  }
}

task {
  name          = "cts-example"
  description    = "Example task"
  module        = "findkim/print/cts"
  version       = "0.1.0"
  condition "services" {
    names = ["web", "api"]
  }
}
```

Custom Terraform Providers



The screenshot shows a web browser window displaying the HashiCorp Learn website. The page is titled "Custom Providers" under the "Terraform" section. A list of topics is shown on the left, with "Setup and Implement Read" highlighted. The main content area displays the title "Setup and Implement Read" and a link to "Create an account". Below this, a box indicates a duration of 13 minutes and lists the products used as Terraform.

HashiCorp Learn Browse products ▾

Terraform

Custom Providers

- Perform CRUD Operations with Providers
- **Setup and Implement Read**
- Add Authentication to a Provider

Jump to section ▾

Setup and Implement Read

Reference this often? [Create an account](#) to bookmark tutorials.

🕒 13 MIN PRODUCTS USED: Terraform

Sentinel in TFE/TFC



github.com/hashicorp/terraform-sentinel-policies

Search or jump to... Pull requests Issues Marketplace Explore

hashicorp / terraform-sentinel-policies Public

Watch 69 Fork 76 Star 47

<> Code Issues 1 Pull requests 2 Actions Projects Security Insights

main 1 branch 2 tags

Code

hcrhall Merge pull request #6 from Rabattkarte/main 8de905d on Jul 18 22 commits

aws	make allowed_regions a configurable param	last month
azure	add links to aws, azure, and registry functions docs	7 months ago
cloud-agnostic	adding tests for prevent-tfe-provider-workspace-deletion	5 months ago
common-functions	Give pshamus credit	7 months ago
gcp	Specify module gcp-functions for testing 🙌	last month
vmware	remove raw data	7 months ago
.gitignore	remove raw data	7 months ago
LICENSE	Initial commit	7 months ago
README.md	add map_key files and check-ec2-environment-tag.sentinel	7 months ago

README.md

Example Third Generation Sentinel Policies for Terraform

This repository contains Sentinel policies for use with Terraform Cloud and Terraform Enterprise.

About

Example Sentinel Policies for use with Terraform Cloud and Terraform Enterprise

- Readme
- MPL-2.0 license
- Code of conduct
- 47 stars
- 69 watching
- 76 forks

Releases 2

v1.0.1 Latest on Feb 1

+ 1 release

Packages

No packages published

Sentinel in TFE/TFC



HashiCorp Sentinel Playground Learn Sentinel Sentinel Docs Give Feedback

policy.sentinel Parameters Globals

1 import "tfplan/v2" as tfplan
2
3 // Sentinel filter expression used to filter out all aws_s3_bucket resources
4 // that will change once the Terraform plan has been applied.
5 aws_s3_buckets = filter tfplan.resource_changes as _, resource_changes {
6 resource_changes.type is "aws_s3_bucket" and
7 resource_changes.mode is "managed" and
8 (resource_changes.change.actions contains "create" or
9 resource_changes.change.actions is ["update"])
10 }
11
12 // Sentinel rule used to evaluate the configuration of all filtered aws_s3_bucket resources.
13 // The rule ensures when all changes have been applied, the S3 bucket configuration will
14 // have the "private" ACL configured.
15 aws_s3_bucket_acl_is_private = rule {
16 all aws_s3_buckets as _, aws_s3_bucket {
17 aws_s3_bucket.change.after.acl is "private"
18 }
19 }
20
21 main = rule {
22 aws_s3_bucket_acl_is_private
23 }
24

Share

mock-tfplan-v2.sentinel x

1 resource_changes = {
2 "aws_s3_bucket.b": {
3 "address": "aws_s3_bucket.b",
4 "change": {
5 "actions": [
6 "create",
7],
8 "after": {
9 "acl": "private",
10 },
11 },
12 "mode": "managed",
13 "type": "aws_s3_bucket",
14 },
15 }
16

Add Mock

Output

Download Run

Press "Run" to get policy output.

04

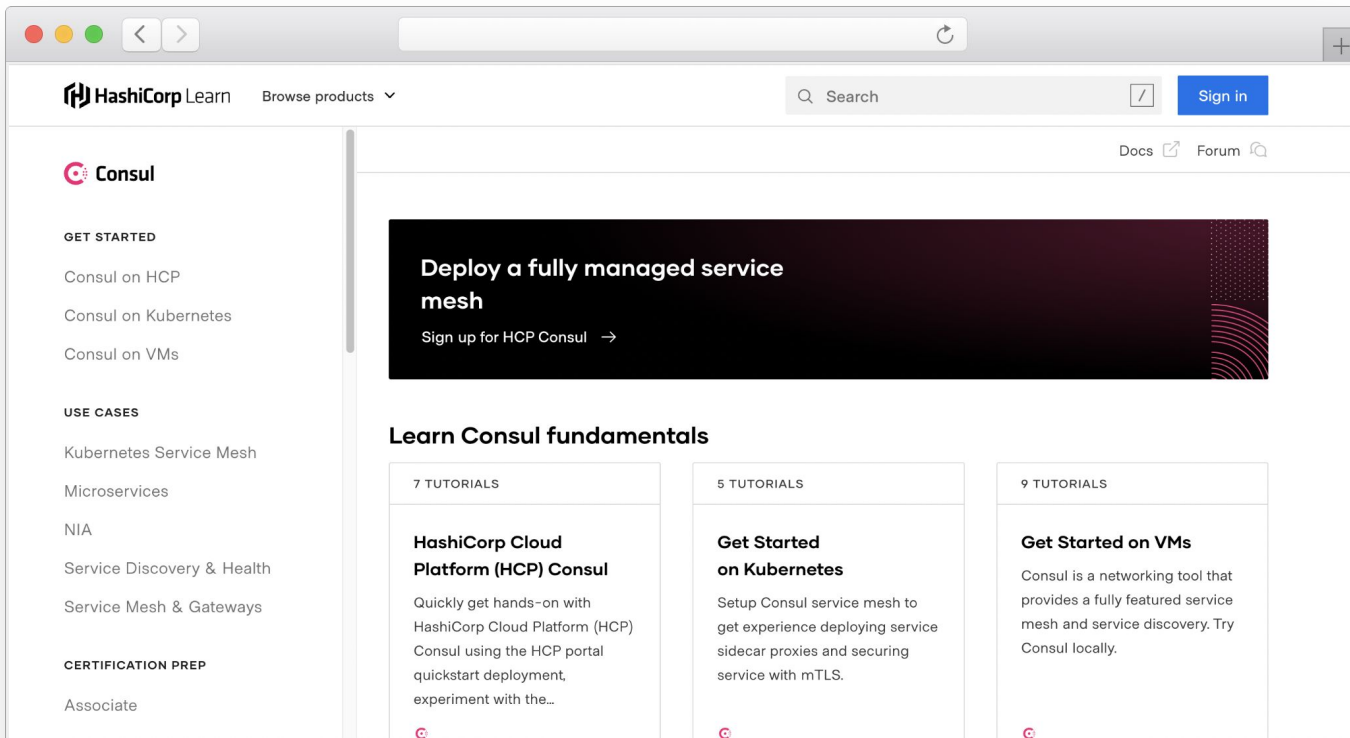
Next Steps

Learn

<https://learn.hashicorp.com/consul>



Step-by-step guides to accelerate deployment of Consul



The screenshot shows the HashiCorp Learn website for Consul. The page has a dark theme. At the top, there's a navigation bar with the HashiCorp logo, 'HashiCorp Learn', a 'Browse products' dropdown, a search bar, and a 'Sign in' button. Below the navigation bar, there's a sidebar on the left with the Consul logo and a list of links under 'GET STARTED' (Consul on HCP, Consul on Kubernetes, Consul on VMs) and 'USE CASES' (Kubernetes Service Mesh, Microservices, NIA, Service Discovery & Health, Service Mesh & Gateways). Under 'CERTIFICATION PREP', there's a link to 'Associate'. The main content area features a large dark banner with the text 'Deploy a fully managed service mesh' and a link 'Sign up for HCP Consul'. Below this, there's a section titled 'Learn Consul fundamentals' with three columns of tutorials. The first column is titled 'HashiCorp Cloud Platform (HCP) Consul' and has 7 tutorials. The second column is titled 'Get Started on Kubernetes' and has 5 tutorials. The third column is titled 'Get Started on VMs' and has 9 tutorials.

HashiCorp Learn Browse products

Search Sign in

Docs Forum

Consul

GET STARTED

- Consul on HCP
- Consul on Kubernetes
- Consul on VMs

USE CASES

- Kubernetes Service Mesh
- Microservices
- NIA
- Service Discovery & Health
- Service Mesh & Gateways

CERTIFICATION PREP

- Associate

Deploy a fully managed service mesh

Sign up for HCP Consul →

Learn Consul fundamentals

7 TUTORIALS

HashiCorp Cloud Platform (HCP) Consul

Quickly get hands-on with HashiCorp Cloud Platform (HCP) Consul using the HCP portal quickstart deployment, experiment with the...

5 TUTORIALS

Get Started on Kubernetes

Setup Consul service mesh to get experience deploying service sidecar proxies and securing service with mTLS.

9 TUTORIALS

Get Started on VMs

Consul is a networking tool that provides a fully featured service mesh and service discovery. Try Consul locally.



Resources

- [Learn Guide: Network Automation with CTS](#)
- [Consul Terraform Sync](#)
- [CTS Network Drivers](#)
- [CTS Tasks](#)
- [Configuration Options for CTS](#)
- [CTS Compatibility](#)
- [Secure CTS for Production](#)
- [A10 Network Terraform Modules & Provider\(s\)](#)
- [F5 Terraform Modules & Provider\(s\)](#)
- [Palo Alto Networks Terraform Modules & Provider\(s\)](#)
- [VMWare Terraform Modules & Provider\(s\)](#)

Need Additional Help?



Customer Success

Contact our Customer Success Management team with any questions. We will help coordinate the right resources for you to get your questions answered.

customer.success@hashicorp.com

Technical Support

Something not working quite right? Engage with HashiCorp Technical Support by opening a ticket for your issue at support.hashicorp.com.

Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers

discuss.hashicorp.com

Upcoming Onboarding Webinars



Webinar

*Program Closing &
Readiness Checklist*

Topics include: A readiness checklist and some great resources to continue on your Consul journey



Q & A



Thank You

customer.success@hashicorp.com

www.hashicorp.com