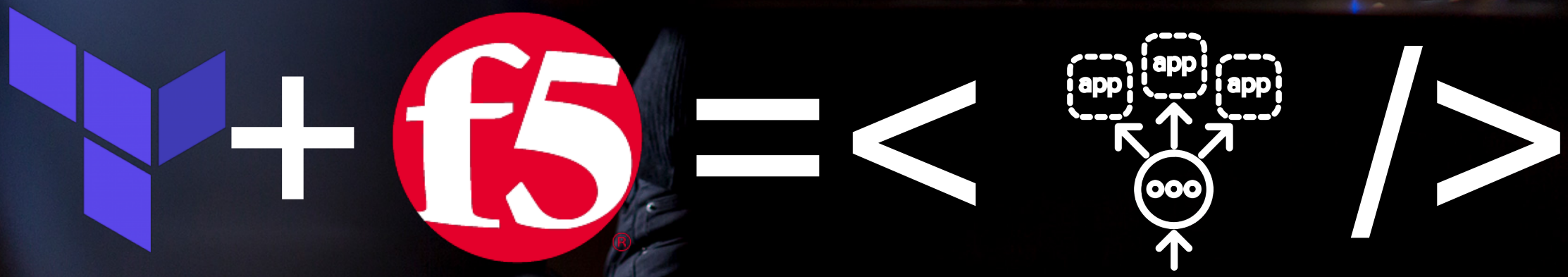
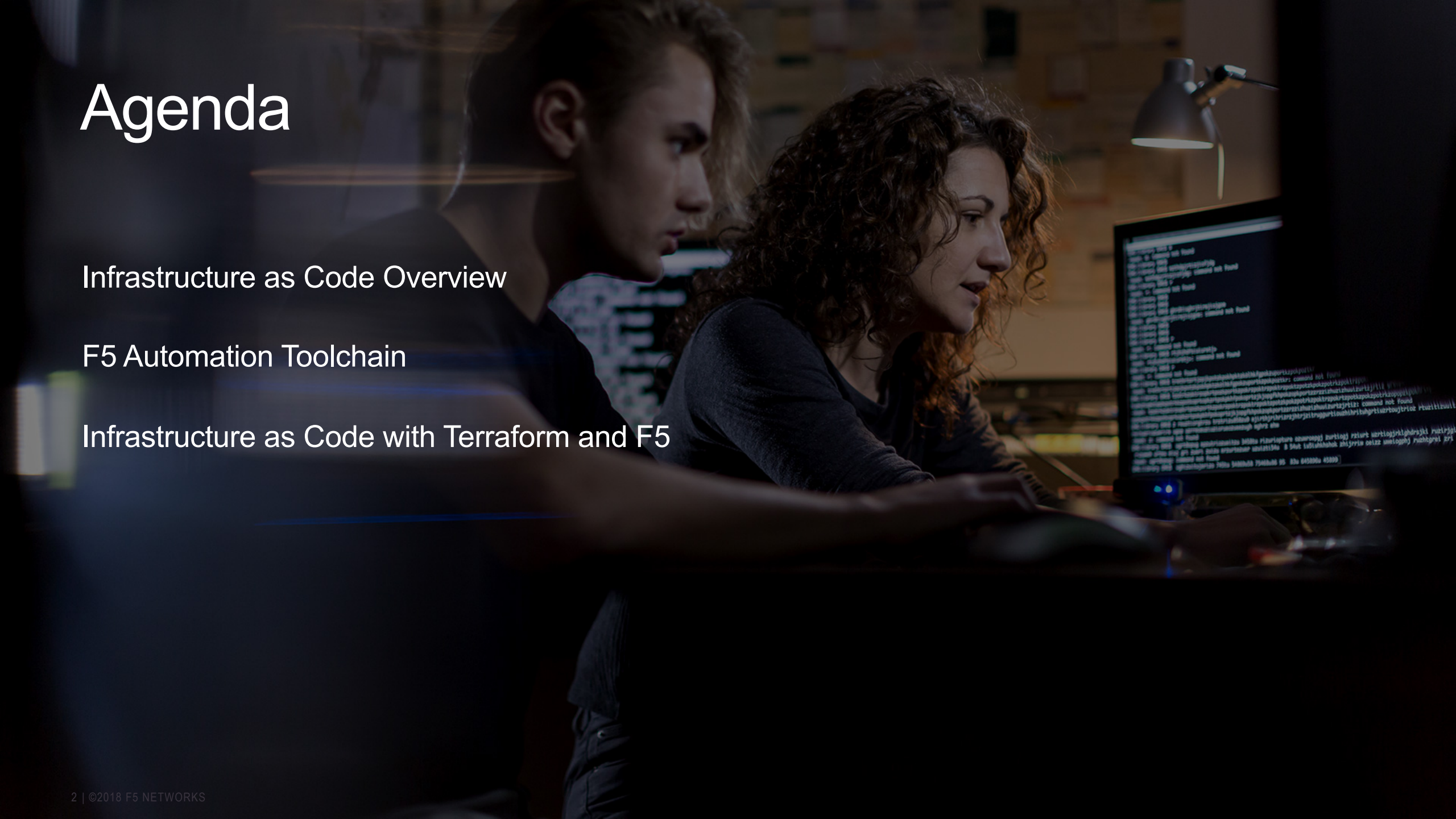
Infrastructure as Code Webinar



Agenda



Infrastructure as Code Overview

F5 Automation Toolchain

Infrastructure as Code with Terraform and F5

Infrastructure as Code Overview

Key Market Trends

Execution at the speed of software

- Agility, DevOps, NFV, SDN, new services platforms
- Fast changing business needs

Changing customer behavior and new expectations

- Everything on demand
- New services with a press of a button – or auto deployed

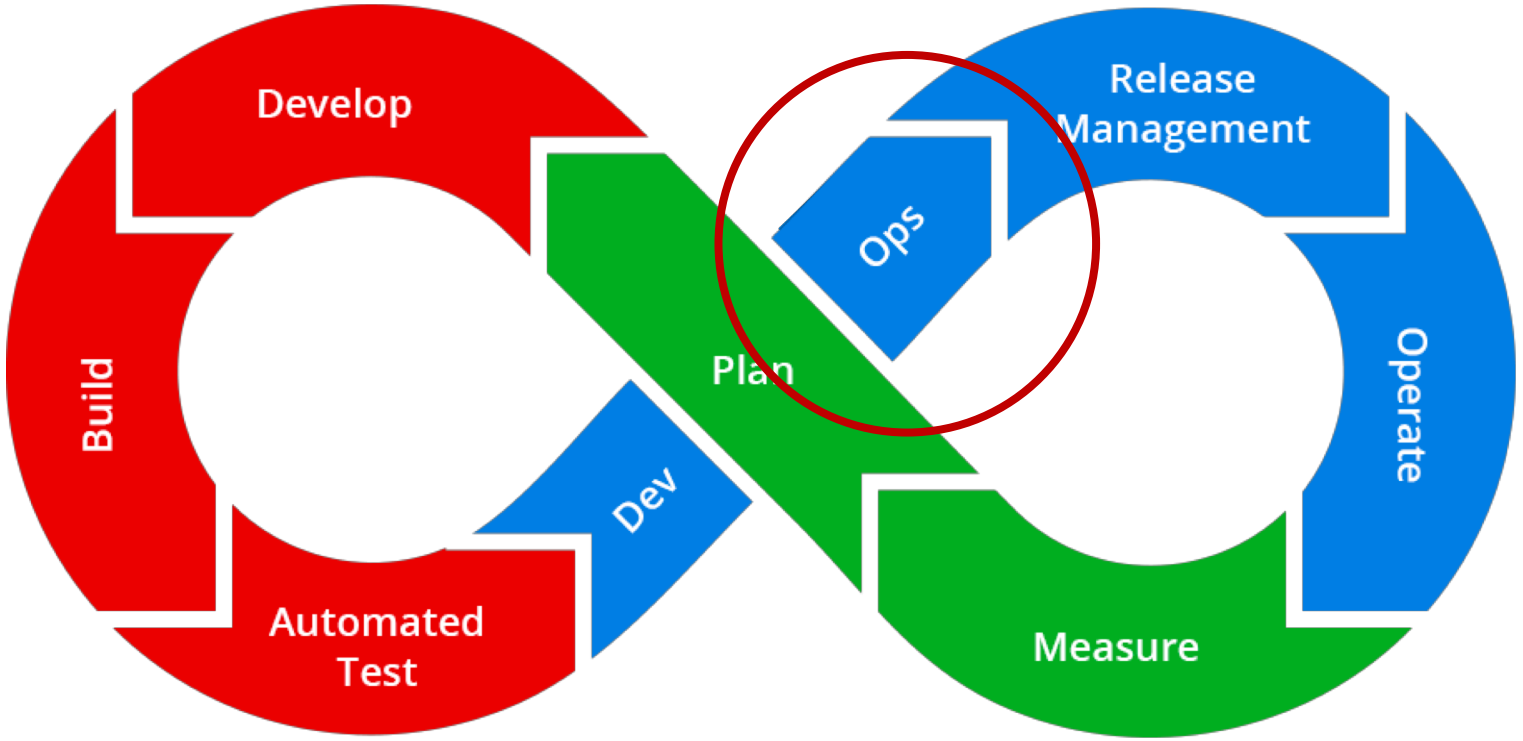
Rapidly changing business models

- Cloud services, microservices, virtualization, programmable networks
- New ecosystems and value chains

All of this requires successful, flexible automation.

DevOps Continuous Integration Cycle

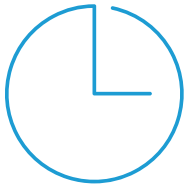
What's Missing?



Why Infrastructure as Code

Cost Reduction

- Reduce cost of deployment and maintenance
- Focus engineers on other higher-level tasks
- Reduce potential outages



Speed Increase

- Deploy apps and needed infrastructure on-demand
- Mimic Cloud Provider capabilities

Risk Reduction

- Reduce human errors
- Build inherent security
- Can be integrated into same CI/CD system developers use today

https://en.wikipedia.org/wiki/Infrastructure_as_code

Two Main Approaches

IMPERATIVE	VS	DECLARATIVE
Tell the system **HOW** to do something – every step of the way		Tell the system **WHAT** you want, and let it figure out **HOW** to do it

Declarative API – you define the desired end-state – it fills in the details on "how to get there"

Declarative Benefits

DECLARATIVE

Tell the system **WHAT** you want, and let it figure out **HOW** to do it

Benefits

- Minimizes need for TMOS domain expertise
- Speeds dev time
- Requires few API calls
- Minimizes deployment errors
- Eases integration int A&O systems
- Repeatable and enforces policy compliance

Declarative API – replace numerous API calls with one declarative API

Methods

Push	vs	Pull
Configuration is pushed to the infrastructure		Infrastructure has to pull configurations from master node

There is no “right” method, the choice is typically based on the tool you choose.

Tools

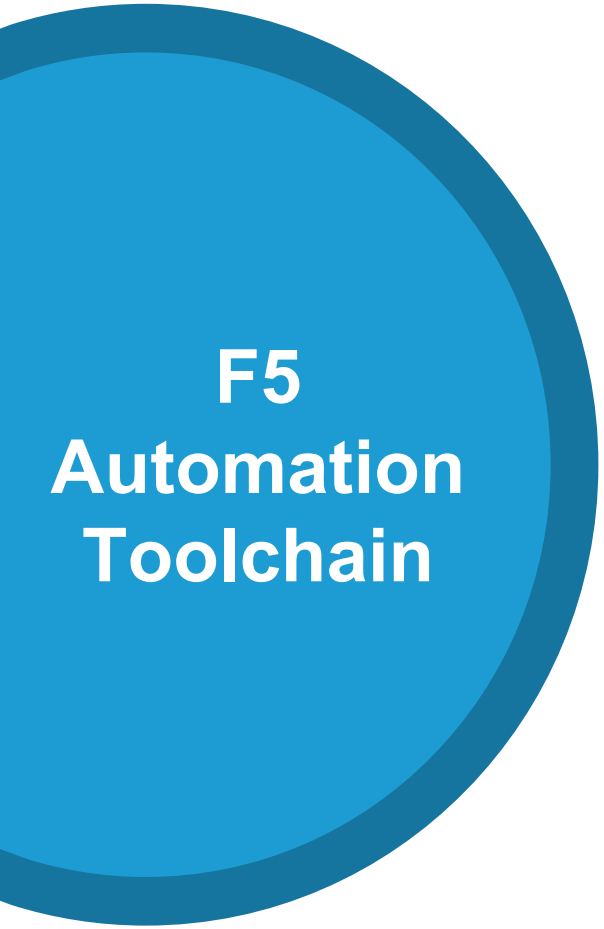
Ansible	VS	Terraform
Great configuration management tool with a powerful template engine		Orchestration tool with a powerful state management

I would argue you can and should use both together!
Check out the Linux Academy blog for more info.

https://linuxacademy.com/blog/devops/ansible-vs-terraform-fight/

F5 Automation Toolchain

F5 Automation Toolchain

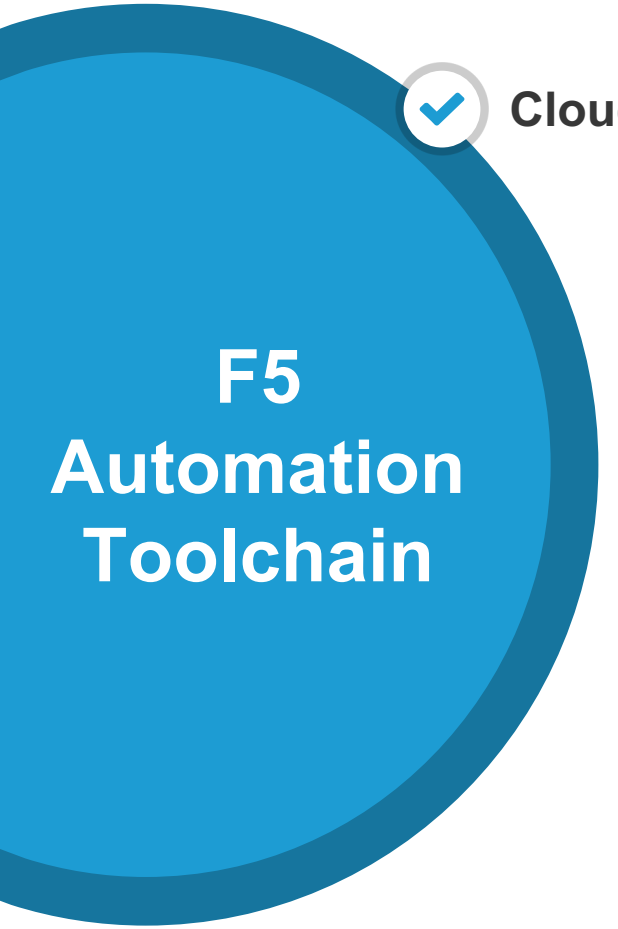


The Toolchain

A set of free tools for automating deployment and configuration of F5 devices and services through declarative APIs.

Use for automation and integration of F5 solutions into automation and orchestration systems.

F5 Automation Toolchain



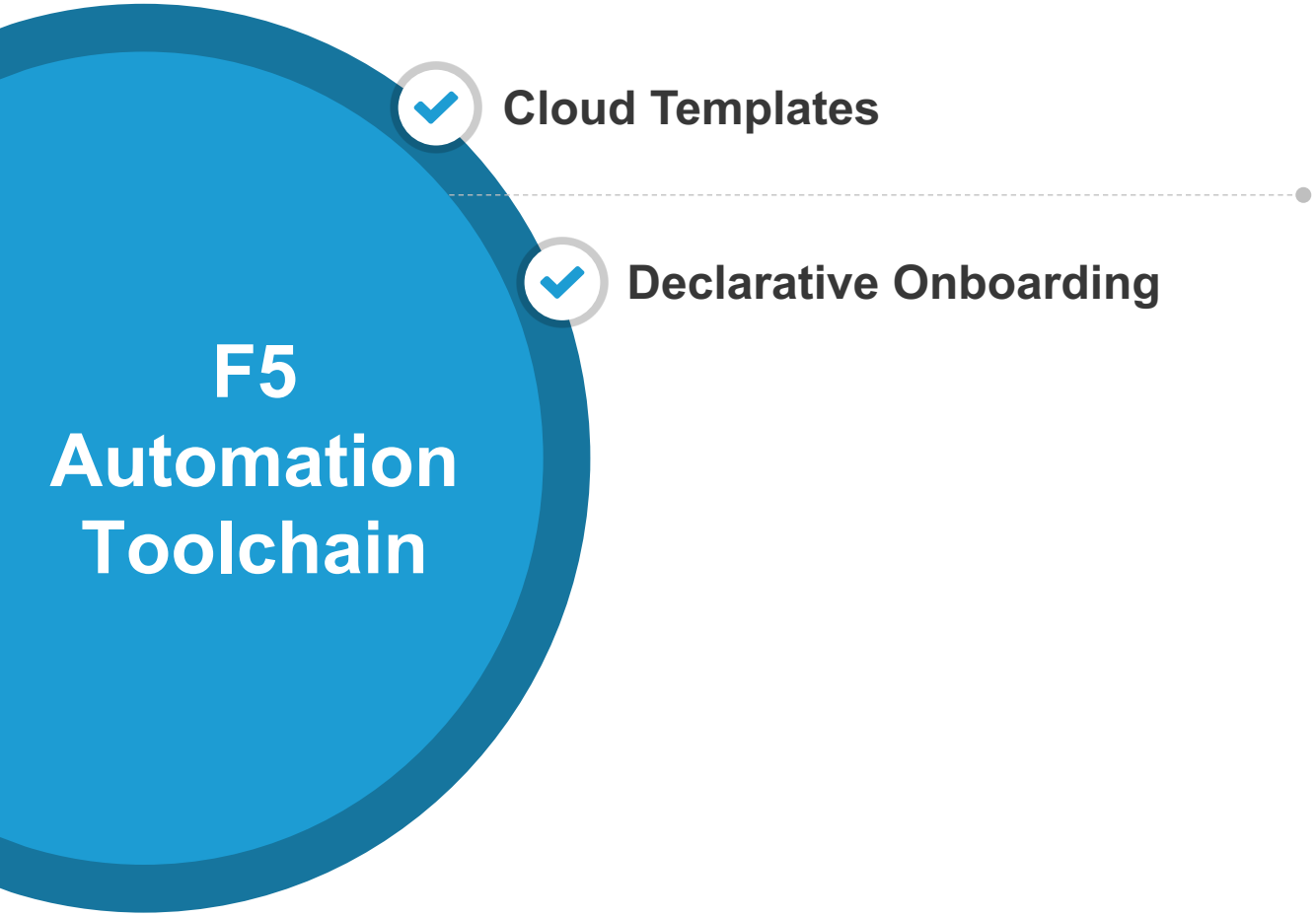
Cloud Templates

F5 Cloud Templates

Easily deploy F5 fully configured, operational F5 BIG-IP VEs into public and private clouds.

Leverage each cloud's native services to support a wide variety of BIG-IP VE use cases.

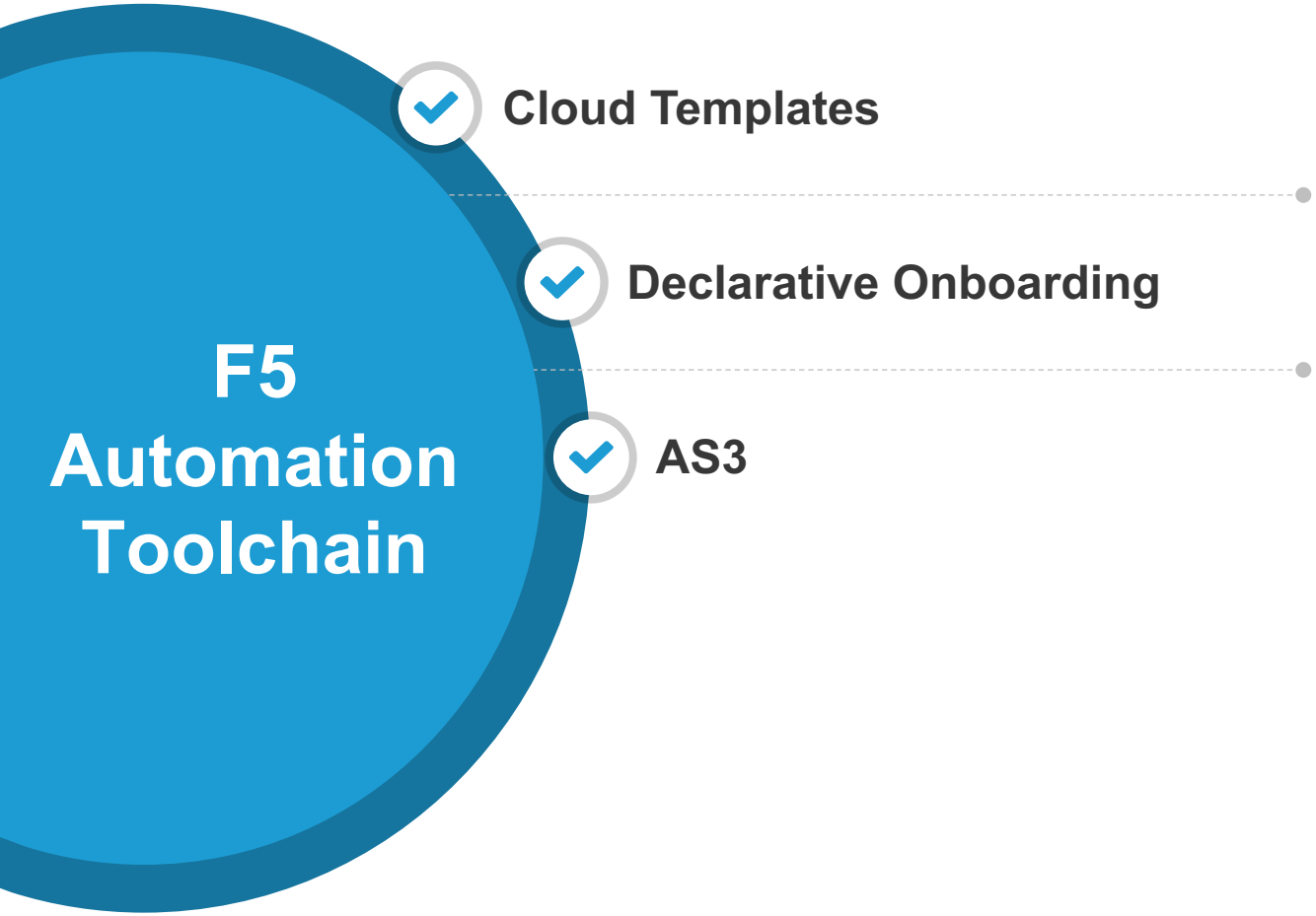
F5 Automation Toolchain



F5 Declarative Onboarding

Provisions initial configuration of virtual BIG-IP devices through a declarative API.

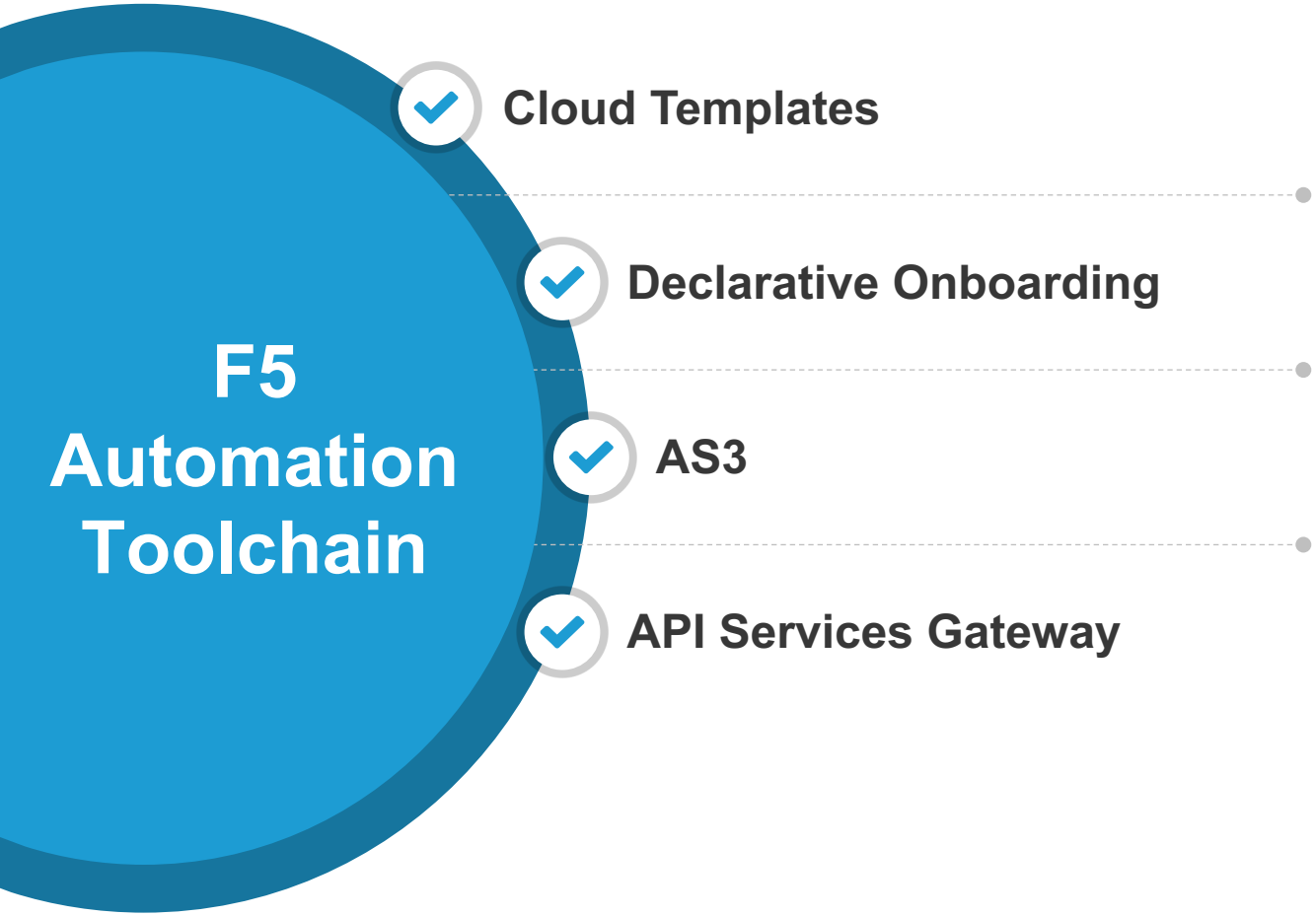
F5 Automation Toolchain



F5 Application Services 3 Extension

Configure F5 modules using declarative APIs. Runs on TMOS, as a container, or in BIG-IQ 6.1+

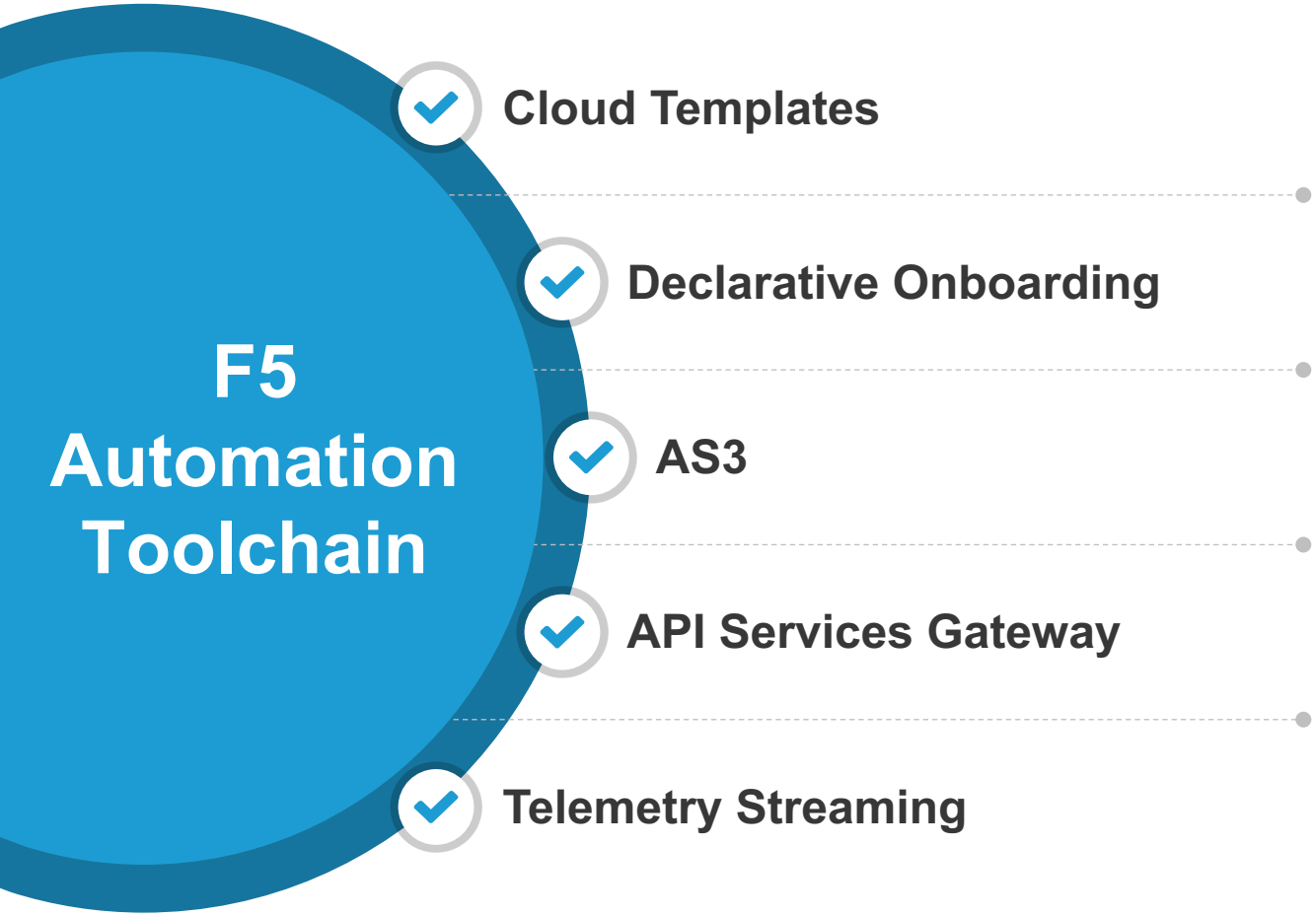
F5 Automation Toolchain



F5 API Services Gateway

Runs F5 & Customer BIG-IP API Extensions in a container or VM platform.

F5 Automation Toolchain



F5 Telemetry Streaming

Stream BIG-IP telemetry for analytics and automation.

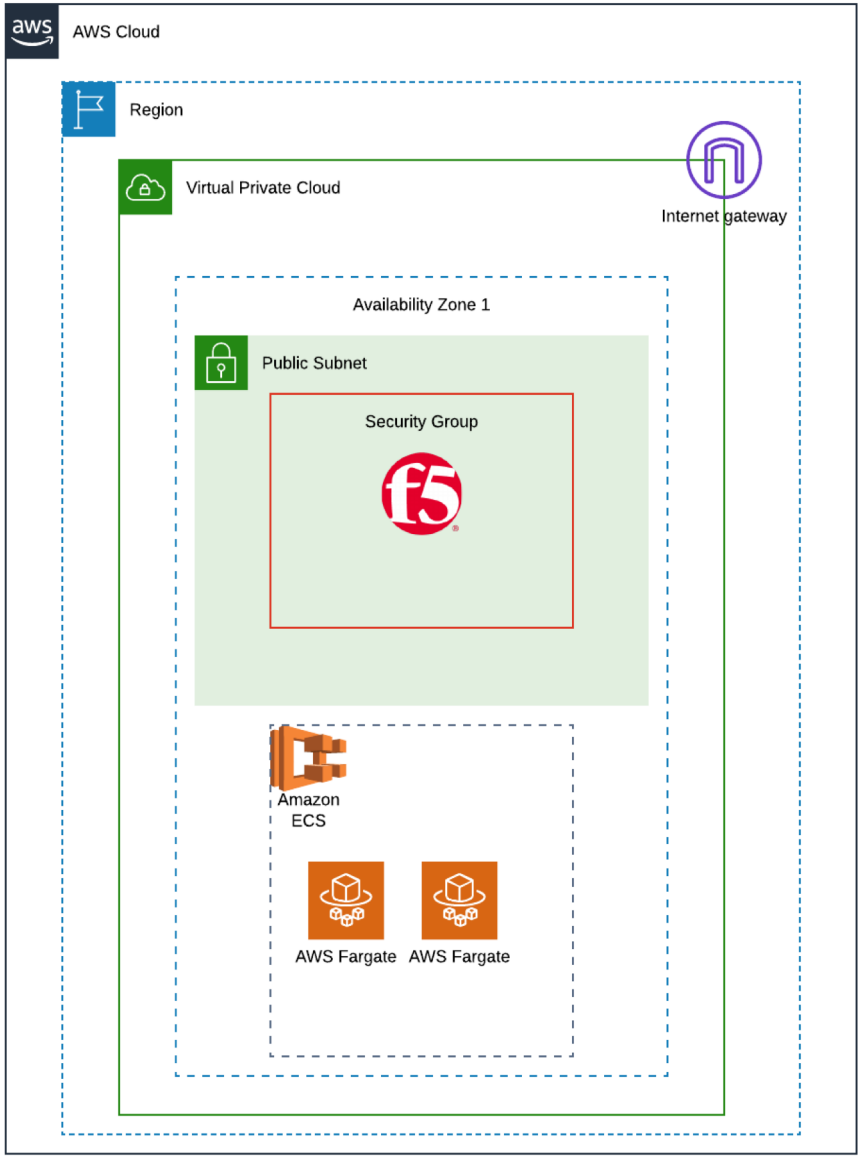
Telemetry Streaming is an BIG-IP API extension that will send client/server stats from the BIG-IP in Kafka format.

IaC with Terraform and F5

Design

In this example we'll deploy:

- Single AZ deployment in us-east-2
  - Security group to only allow management from your public IP
  - Security group to allow http and https traffic from anywhere
- Single BIG-IP instance
  - Single NIC
  - 25 Mbps
  - Marketplace PAYG licensing
- ECS Cluster
  - 2 Fargate instances running demo app in docker container



Terraform Basics

OVERVIEW

Ability to define desired infrastructure end state and manage the planning and deployment of said end state repeatably in multiple environments

Key Benefits:

- Easily manage infrastructure using code repositories
- Ability to see what changes will be made using plan method
- Powerful state management
- Ability to “taint” certain components in your infrastructure without re-deploying everything

Maintained by Hashicorp

Declarative approach

Push method

Written in Go

Terraform Basics

COMMANDS AND MODULES

This example uses Terraform modules to help break up the individual components for ease of deployment and reuse

- BIG-IP
- Compute (ECS)
- IAM
- Networking

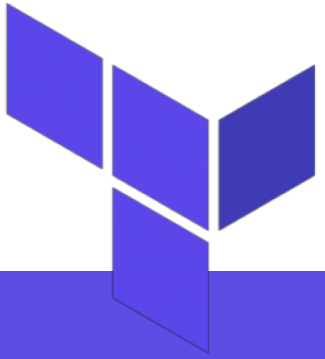
We define output files so we can pass variables between the various module

For a great tutorial check out Linux Academy's Managing Applications and Infrastructure with Terraform Course (7 Day Trial)

```
terraform init

terraform plan

terraform apply

terraform taint

terraform destroy
```

Terraform Basics

VARIABLE INTERPOLATION



Interpolation allows you to embed variables within strings using Terraform or JSON syntax

- Wrapped in ${}
- Example:
  - resource "aws_vpc" "f5_demo_vpc" {
    cidr_block = "${var.vpc_cidr}”
    enable_dns_hostnames = true
    enable_dns_support = true
    }

Allows you to reference variables, attributes of resources, call functions and more

variables.tf

```
variable "vpc_cidr" {}
variable "f5_count" {}
variable "public_cidrs" {
    type = "list”
}
```

terraform.tfvars

```
vpc_cidr = "10.0.0.0/16”
f5_count = 1
public_cidrs = [
"10.0.1.0/24","10.0.2.0/24”
]
```

Requirements

System Requirements:

Terraform 0.11.13 or higher

BIG-IP 13.1 or higher

Install and configure the AWS CLI

- https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html

Tips:

If you do not want to use the AWS CLI then you'll need to define your access and secret key information in the AWS Provider

The AWS CLI is used to find the private IP address of the ECS instances since this is not returned in the AWS API payload upon creation

Demo

Q&A

Resources

Session 1 Recording – BIG-IP Automation with Ansible
https://www.youtube.com/watch?v=AodZltew9wk

Session 2 Code and Slides – https://github.com/codygreen/Automation_Webinar

Session 3 Registration Link (also in follow-up email) –
https://f5networks.zoom.us/webinar/register/9415537400250/WN_dVwU_fVTTQeCGH6QwrR3cA