Prerequisites
WiFi: MSFTGUEST Event Code: msevent820sn

Materials: **http://bit.ly/2V2d1NQ**

Make sure you have:

- Azure subscription - free with a credit card https://azure.microsoft.com/free/
  Ask me for a temporary one if you don't have any
- Azure CLI: https://docs.microsoft.com/en-us/cli/azure/install-azure-cli
  Run `azure login` to input credentials
- Pulumi: https://www.pulumi.com/docs/get-started/install/
- .NET Core SDK 3.1: https://dotnet.microsoft.com/download

For 2nd set of labs, if you plan to do Docker or Kubernetes tutorials:

- Docker CE: https://docs.docker.com/install/
- Kubectl CLI: https://kubernetes.io/docs/tasks/tools/install-kubectl/

WiFi:
SSID: TODO
Password: TODO

Hands-On Infrastructure as Code Workshop with Pulumi, Azure, and C#

Mikhail Shilkov <mikhail@pulumi.com> @MikhailShilkov

Copenhagen
18 February 2020

Workshop Outline

Part 1 — Infrastructure as Code Concepts (60m)

- overview
- one end-to-end lab

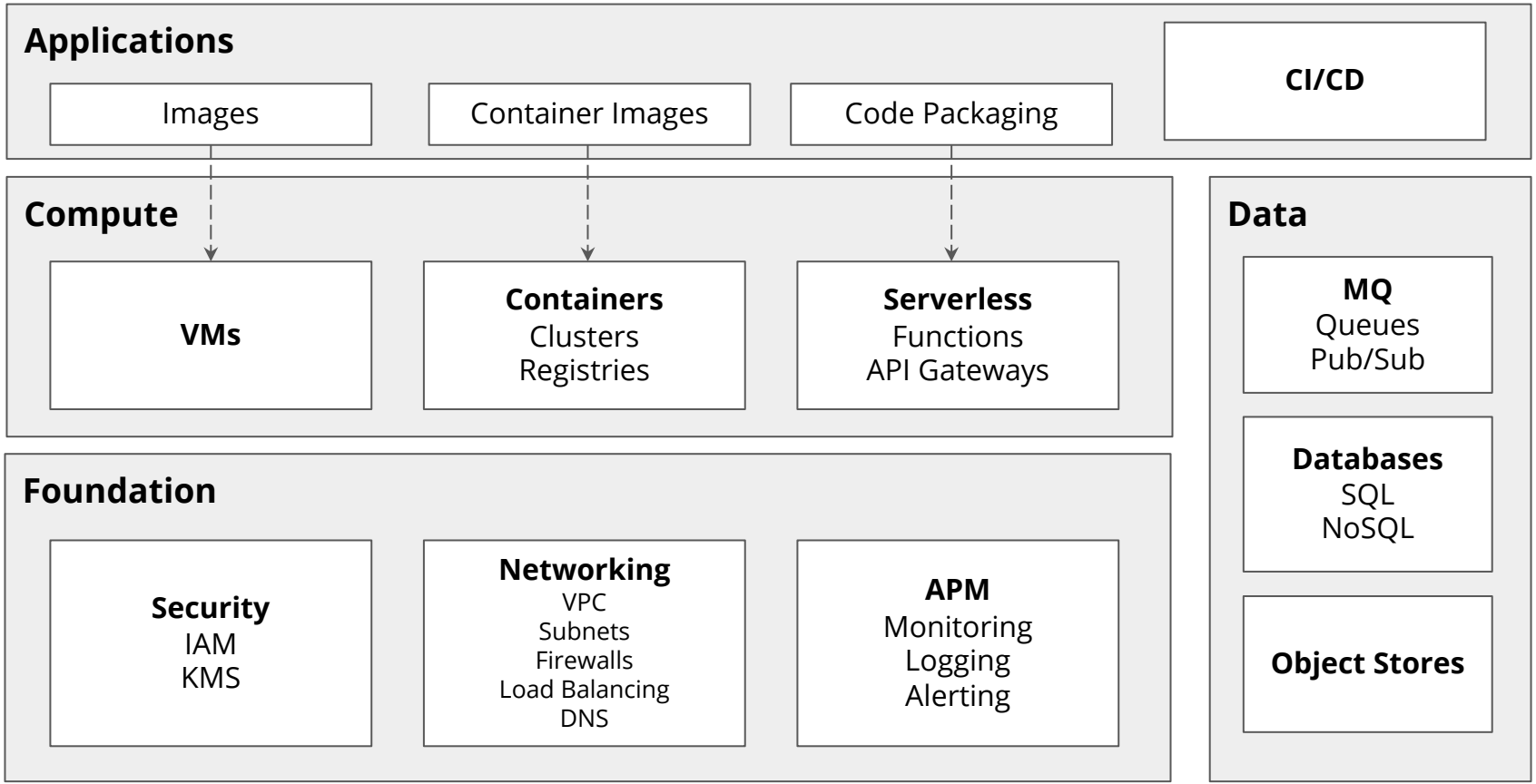
Part 2 — Modern Cloud Architectures (120m)

- four labs:
  - Serverless
  - VMs
  - containers (on Container Instances)
  - containers (on Kubernetes)

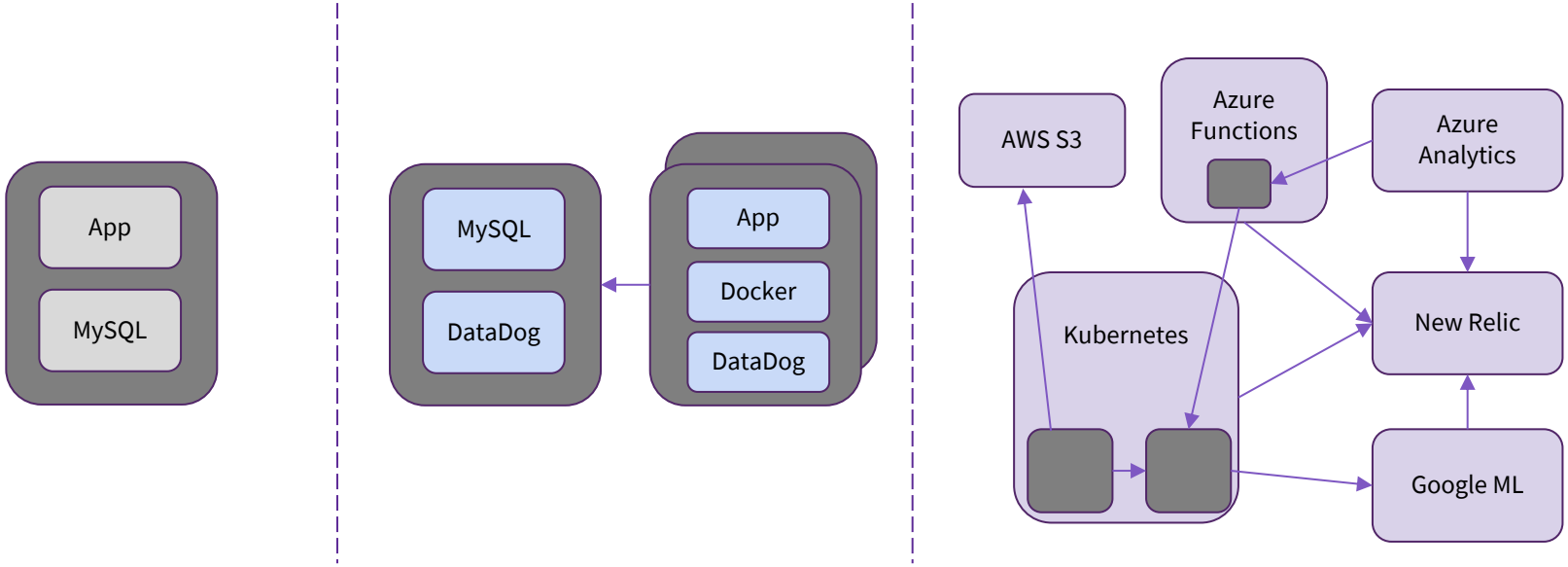
Infrastructure as Code Concepts

20m Overview
40m Hands-On Lab 1
110m Hands-On Labs 2-5
10m Recap

Infrastructure Landscape



Cloud Transition



EARLY CLOUD
Mostly Static
N-Tier Apps
VMs
Private Cloud

CURRENT CLOUD
Partly Dynamic
Less Monolithic
VMs and Containers*
Hybrid - Public/Private

FUTURE CLOUD
Fully Dynamic
Hyper-Connected Services
Containers and Serverless
Mostly Public Cloud

*Experimentation

Concepts

So, your application needs infrastructure resources (VM, database, cluster, queue, etc). How do you create and manage them?

- **manual**: point and click to create/modify resources in the web portal.
- **ad-hoc automation**: CLI commands or scripts to create/modify resources.
- **infrastructure as code**:
  - **provisioning**: declaratively create/modify resources.
  - **configuration**: change state of an existing resource post-provisioning.

Philosophical difference between immutable and mutable infrastructure (cattle vs pets).

- VMs are usually pets.
- containers and serverless are usually cattle.

Infrastructure as Code (IaC)

Declare infrastructure as “code,” using:

- markup languages (YAML/JSON)
- domain-specific languages (DSLs)
- general purpose (“real”) languages (C#/TypeScript/Python/...)

Benefits:

- automatable
- repeatable
- review and version like code (often in Git)

Infrastructure as Code Landscape

Cloud provider tools

- AWS CloudFormation and CDK
- Azure Resource Manager (ARM) Templates
- Google Deployment Manager

Cloud independent tools

- Kubernetes YAML
- Helm YAML/templates
- HashiCorp Terraform
- **Pulumi (what we will be using today)**

Infrastructure as Code (JSON)

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "InstanceType" : "t2.micro",
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "ImageId" : { "ami-0080e4c5bc078760e" },
      }
    },
    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP over port 80",
        "SecurityGroupIngress" : [{
          "IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" : "0.0.0.0/0"
        }]
      }
    }
  }
}
```

Infrastructure as Code (DSL)

```
provider aws {
  region = "eu-central-1"
}

resource "aws_security_group" "web_sg" {
  description = "Enable HTTP over port 80"
  ingress {
    protocol    = "tcp"
    from_port   = 80
    to_port     = 80
    cidr_blocks = [ "0.0.0.0/0" ]
  }
}

resource "aws_instance" "web" {
  ami           = "ami-0080e4c5bc078760e"
  instance_type = "t2.micro"
  security_groups = [ "$(aws_security_group.web_sg.id)" ]
}
```

Infrastructure as Code (Language)

```
import * as aws from "@pulumi/aws";

const group = new aws.ec2.SecurityGroup("web-sg", {
    description: "Enable HTTP over port 80",
    ingress: [
        { protocol: "tcp", fromPort: 80, toPort: 80, cidrBlocks: ["0.0.0.0/0"] },
    ],
});

const server = new aws.ec2.Instance("web", {
    instanceType: "t2.micro",
    securityGroups: [ group.id ],
    ami: "ami-0080e4c5bc078760e",
});
```

Infrastructure as Code (Language 💡)

```
import * as aws from "@pulumi/aws";

const group = new aws.ec2.SecurityGroup("web-sg", {
    description: "Enable HTTP over port 80",
    ingress: [
        { protocol: "tcp", fromPort: 80, toPort: 80, cidrBlocks: ["0.0.0.0/0"] },
    ],
});

for (let az in aws.getAvailabilityZones().names) {
    const server = new aws.ec2.Instance(`web-${az}`, {
        instanceType: "t2.micro",
        securityGroups: [ group.id ],
        ami: "ami-0080e4c5bc078760e",
        availabilityZone: az,
    });
}
```

Using Real Languages

Full power of real languages:

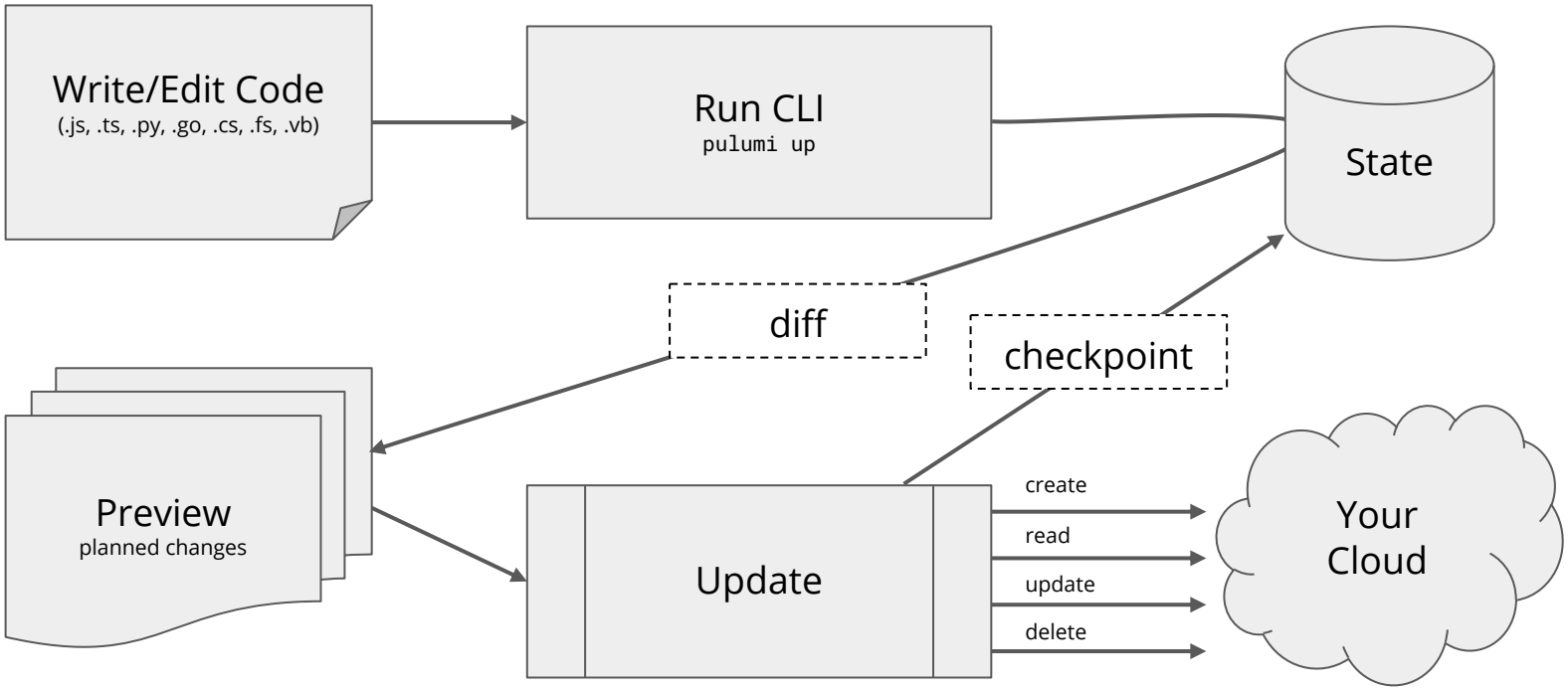
- control flow: loops, conditionals.
- abstraction and reuse: functions, classes, packages.
- share and reuse, don’t copy and paste.
- ****still a desired-state configuration engine** ⇐ *important***

Leverage existing tools, communities, and best practices:

- authoring: IDEs, linters, test frameworks, etc.
- online communities, training, books, knowledge bases.

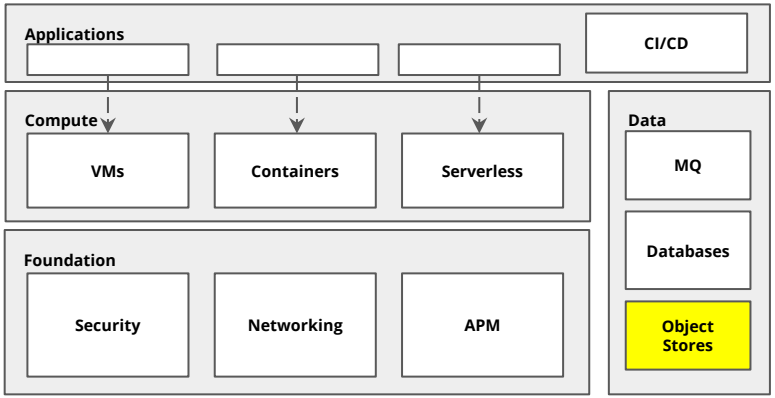
Easier for developers and infrastructure engineers to collaborate.

Pulumi Workflow



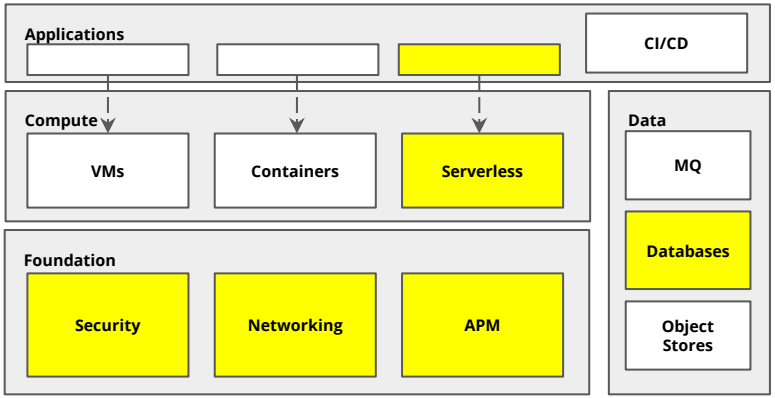
Lab 1 — Modern Infrastructure as Code

- Create a New Project
- Configure AWS
- Provision Infrastructure
- Update Infrastructure
- Make Your Stack Configurable
- Create a Second Stack
- Destroy Your Infrastructure



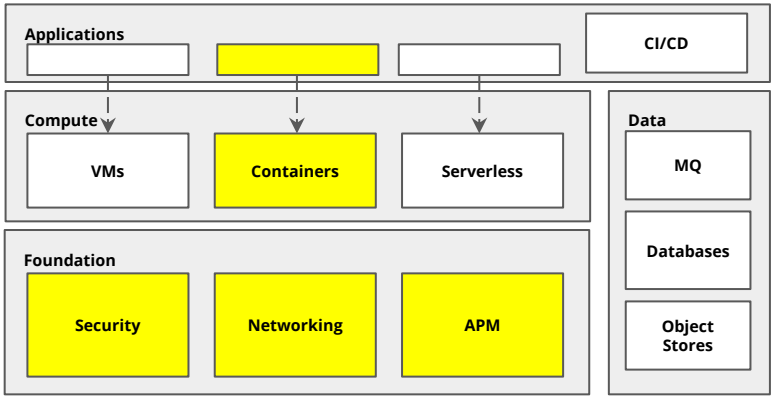
Lab 2 — Serverless with Azure Functions

1. Create a Serverless Azure Functions-based HTTP endpoint
2. Provision all required resources, e.g. Storage and Plan
3. Deploy code and infrastructure together



Lab 3 — Deploying Containers to Azure Container Instances

- Create an Container Group
- Deploy an “Nginx” Service
- Build and Publish a Custom, Private Container Image
- Do a Rolling Deployment

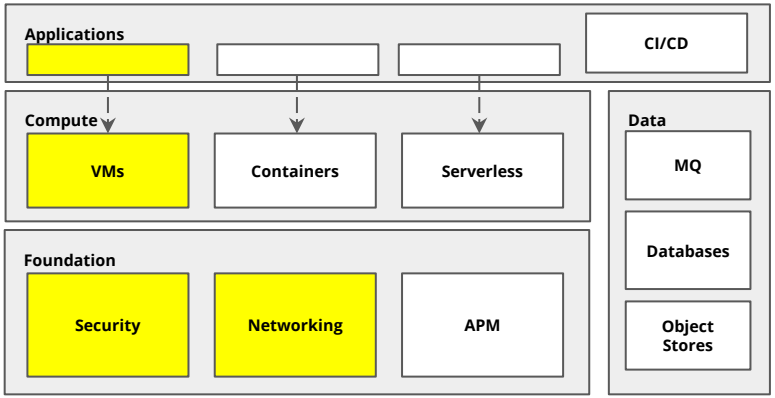


Special prerequisites:

- Docker

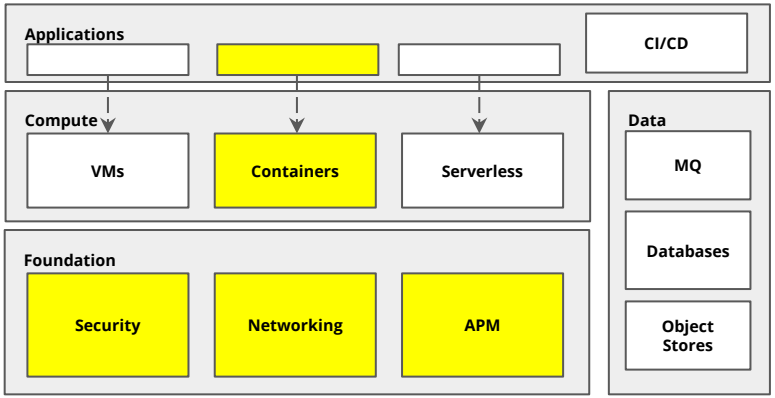
Lab 4 — Provisioning Azure Virtual Machines

- Create a VM and Access It
- Assign and Retrieve a Public IP address



Lab 5 — Deploying Containers to a Kubernetes Cluster

- Connect to a Kubernetes Cluster
- Create a Kubernetes Namespace
- Create a Kubernetes Deployment
- Create a Load-Balanced Kubernetes Service
- Do a Rolling Deployment
- Scale Out the Service



Special prerequisites:

- kubectl
- KUBECONFIG

Lab X — Do Your Own Thing

- Think of a project you have at work or as a hobby that uses cloud
- Try to replicate that kind of cloud resources with Pulumi
- Ideas: App Service, SQL Database, Cosmos DB, KeyVault, …
- Look at https://github.com/pulumi/examples for sample code

https://github.com/pulumi/infrastructure-as-code-workshop
or http://bit.ly/2V2d1NQ
then choose Azure / C#

Hands-On Labs Part 1

20m Overview
40m Hands-On Labs

- Creating a New Project
- Configuring Azure
- Provisioning Infrastructure
- Updating Your Infrastructure
- Making Your Stack Configurable
- Creating a Second Stack
- Destroying Your Infrastructure

https://github.com/pulumi/infrastructure-as-code-workshop

Hands-On Labs Part 2

110m Hands-On Labs

10m Recap

- Provisioning Virtual Machines
- Deploying Containers to Container Instances
- Deploying Containers to a Kubernetes Cluster
- Using Azure Functions for Serverless Applications

Recap

20m Introduction
150m Hands-On Labs
10m Recap

Finishing Labs

All available on GitHub **https://github.com/pulumi/infrastructure-as-code-workshop**

- Infrastructure as Code Lab
- Modern Application Architecture Labs
  - Provisioning Virtual Machines
  - Deploying Containers to Azure Container Instances
  - Deploying Containers to a Kubernetes Cluster
  - Using Azure Functions for Serverless Applications

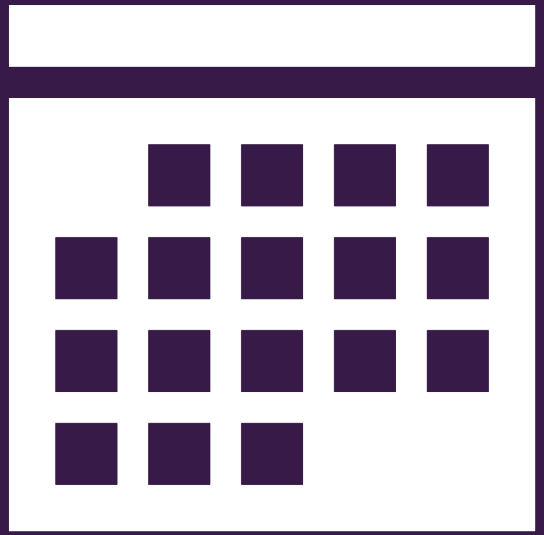
Possible Next Steps

Complete additional labs (all open source).

- Additional providers:
  - AWS, Azure, GCP, DigitalOcean
  - Kubernetes
  - vSphere, OpenStack, F5 BigIP
  - Datadog, NewRelic, GitHub, GitLab
  - more....
- Use secrets management.
- Multi-project infrastructure architectures.
- Import some existing infrastructure.
- Convert some HCL to Pulumi! **https://github.com/pulumi/tf2pulumi**
- Policy as Code.
- Continuous delivery (e.g., triggered by Git commit).

Upcoming Events

Meetups, Talks and Workshops with Pulumi



18 Feb **Copenhagen, DK**

18 Feb **Kaunas, LT**

19 Feb **Malmo, SE**

20 Feb **Amsterdam, NL**

21 Feb **Madrid, ES**

25 Feb **Prague, CZ**

26 Feb **Frankfurt, DE**

30 Mar **Amsterdam, NL**

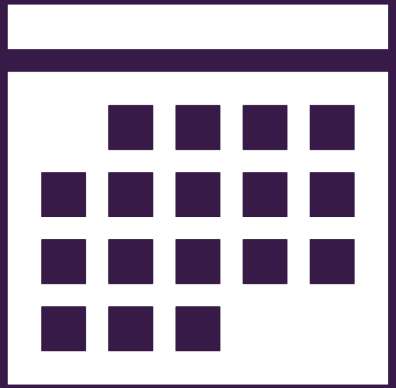
http://bit.ly/2OQ00Tt

pulumi

@PulumiCorp

Pulumi IaC Workshops

Coming to a city near you



Amsterdam – 30 March

The Pulumi team will provide hands-on instruction in a 4-hour workshop showing you how to setup your infrastructure for Kubernetes and other useful configurations.

Includes lunch, workshop, labs and happy hour.

You don't have to attend KubeCon to join this workshop

http://bit.ly/2HgD9wd

pulumi
@PulumiCorp

Q&A

Thank you!

Documentation: **https://pulumi.com/docs**
Examples: **https://github.com/pulumi/examples**
Community Slack: **https://slack.pulumi.com**
Workshop: **https://github.com/pulumi/infrastructure-as-code-workshop**

Mikhail Shilkov <mikhail@pulumi.com> @MikhailShilkov