W&CC: Infrastructure as Code

I heard you love templates, so I put templates inside your templates

Floris Westerman

Traditional Infrastructure

Manual management

Ad-hoc configuration

Error-prone

Hard to scale

Goals for IaC

Reproducible, ephemeral, immutable environments

Clarity

Scalability

Minimal manual work

Infrastructure part of development cycle

Typical features

Declarative code

Version control

Continuous Integration & Deployment

Tools

Program (NPM, pip, NuGet)

Docker (Dockerfile, Docker Compose)

Kubernetes (Kustomize, Helmfile; Rancher, Typhoon)

Software (Ansible, Chef, Puppet)

Virtual Machines (Vagrant)

Operating Systems (k3OS, Flatcar, CoreOS)

Infrastructure (Terraform, CloudFormation)

Development Environments

Some package manager

Docker containers + docker-compose

Vagrant? Puppet? Ansible?

Simple & Quick

```
docker build .
docker-compose up -d
```

Terraform

Modular and versatile

"All or nothing" approach

Live demo

```
terraform plan
terraform apply
```

Continuous Deployment

Incremental and rolling updates

Short time-to-release

Vulnerability checking and other goodies

Tools: GitHub Actions, GitLab, Jenkins, CircleCI

Secrets Management

All configuration is in the repository… what about secrets?

On-demand credentials

Cloud “vault” providers: Azure Key Vault, Google Cloud KMS

Self-hosted: HashiCorp Vault

Simple: Mozilla SOPS

State & Persistance

(Virtually) no application is stateless

Conflicts with ephemerality

Often disregarded

Bigger Picture

IaC often goes hand in hand with:

- Agile workflow
- Network Function Virtualization (ref. Software Defined Networking course)
- Microservices
- DevOps best practices

Merlin OpenStack Cloud

OpenStack

Open-source modular cloud platform

Alternative to big cloud providers

Components

Nova: Virtual Machines

Neutron: Networking

Cinder: Block Storage

Keystone: Identity Management

Glance: Images

Merlin Cloud

OpenStack platform for HPC

Available for W&CC

Credentials distributed soon™

https://merlin.hpc.rug.nl/

Terraform

Basics

Domain-Specific Language: HCL

Resources

Variables

Outputs

Data sources

Providers

Connects Terraform to some API

CRUD operations through resources

Open-source and modular

Available through Registry

Modules

Reusable sets of resources

Every project is a module

Also available through Registry

Backends

Terraform has state

Stored in “backends”

Default: local file

Alternatives: GitLab, Terraform Cloud

Infrastructure as Code

Infrastructure as Live Coding