

Cloud Information Systems

Exercise 12

20th January 2025

1. Announcement: Guest Lecture This Week



- Guests: Check24
- Wednesday 22.01.2025, 12:30 in HS1
- Content of the Talk is **relevant for the exam**

2. Recap: Containers (Docker)

- lightweight form of virtualization
→ rapid creation, short lifetime, replication
 - built on top of operating system (Linux)
 - provides isolation between processes
 - allow packaging software artifacts, including their required libraries
- allows for consistent development environment



2. Recap: Containers (Kubernetes)

- (open-source) platform for managing containerized workloads and services
- provides framework to run distributed systems resiliently by handling tasks such as scaling, failover, and deployment patterns
- a pod is the smallest deployable unit in Kubernetes
 - can contain one or more containers

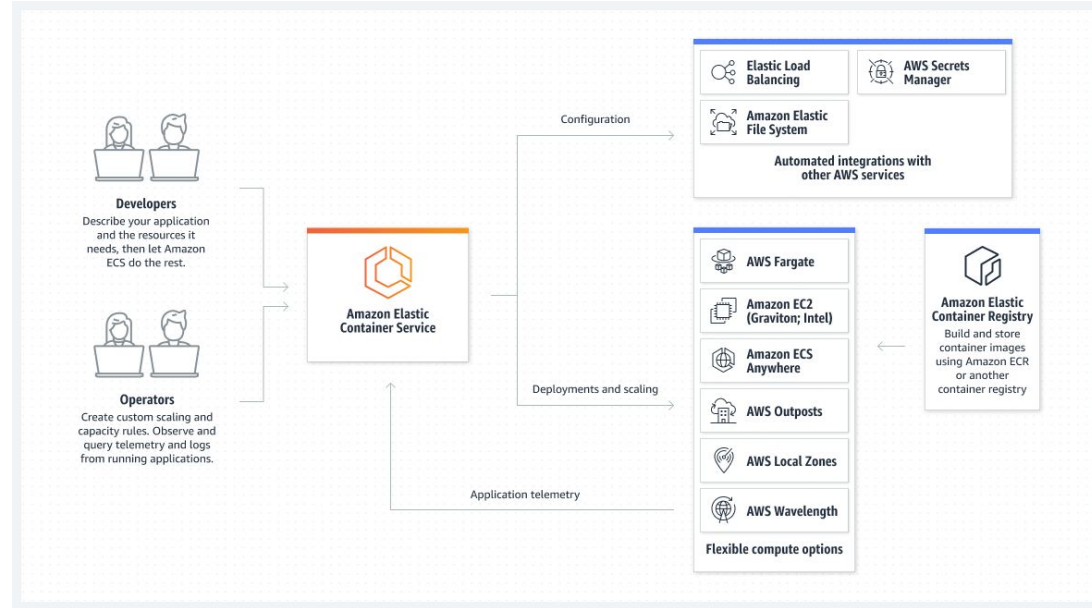


kubernetes

[Link](#)

2. Recap: Amazon Elastic Container Service (ECS)

- proprietary container management service that handles infrastructure tasks
- integrated with AWS ecosystem (and therefore, also tied to AWS)
- can run containers on EC2 or Fargate



[Guide](#)

[Documentation](#)

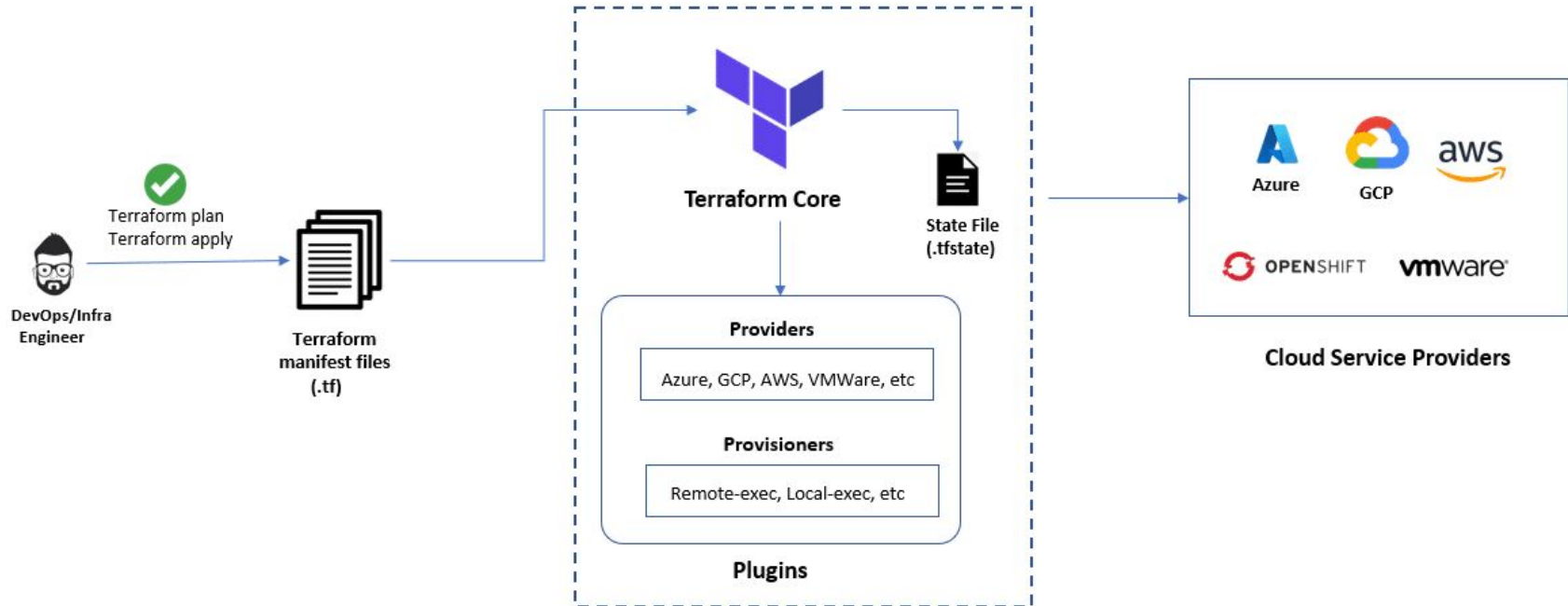
2. Recap: Terraform

- open-source infrastructure as code (IaC) tool by HashiCorp
- allows users to define and provision infrastructure resources across various cloud providers and services
- declare resources with Terraform language (HCL)

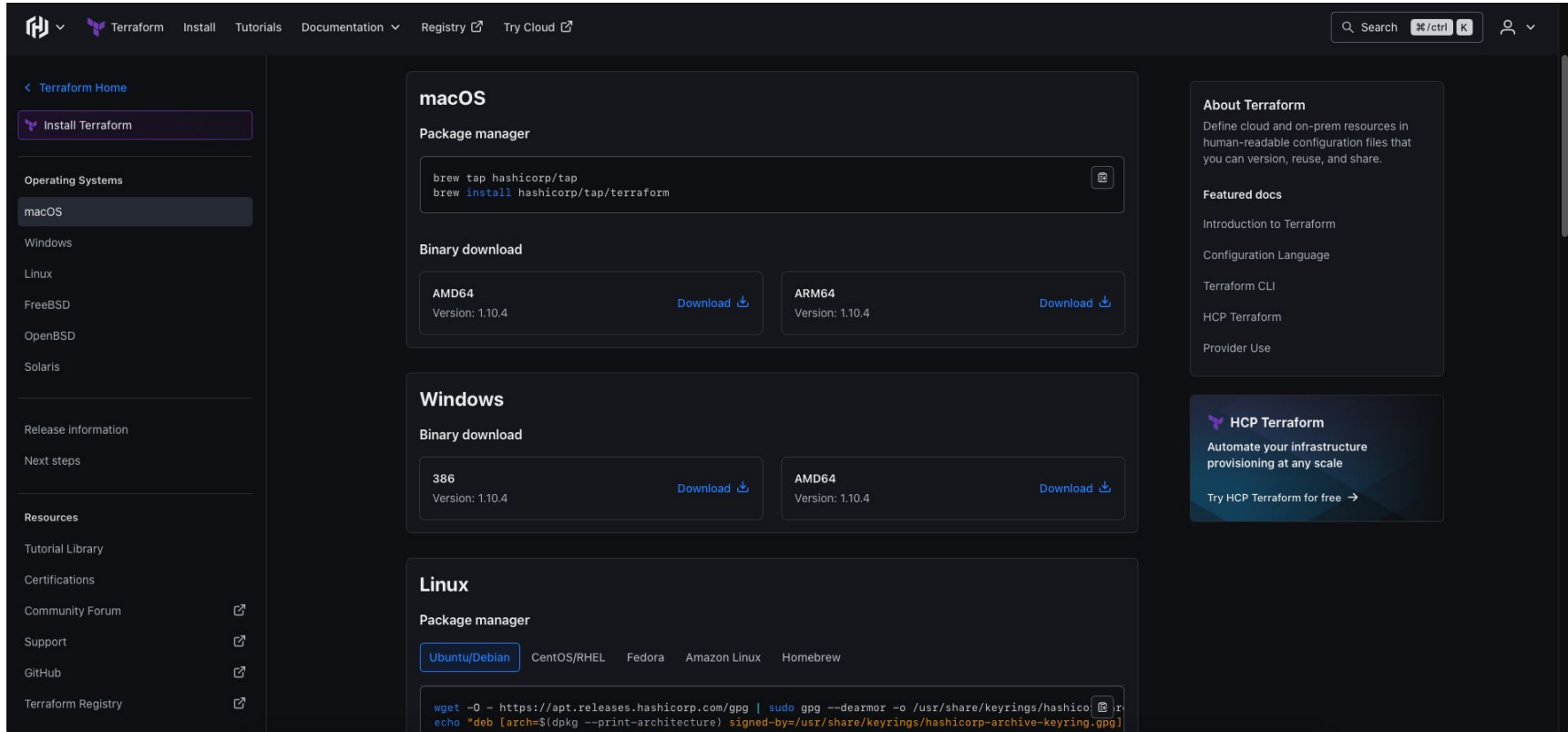


3. Live Demo

Terraform Architecture



1. Terraform Installation



The screenshot shows the Terraform installation guide website. The left sidebar contains navigation links: Terraform Home, Install Terraform (highlighted), Operating Systems (macOS, Windows, Linux, FreeBSD, OpenBSD, Solaris), Release Information, Next steps, and Resources (Tutorial Library, Certifications, Community Forum, Support, GitHub, Terraform Registry). The main content area is divided into three sections: macOS, Windows, and Linux. Each section provides instructions for installing Terraform using a package manager or downloading the binary. The macOS section shows the use of Homebrew to install Terraform. The Windows section shows the download of the Windows binary. The Linux section shows the use of apt to install Terraform. The right sidebar contains links to About Terraform, Featured docs (Introduction to Terraform, Configuration Language, Terraform CLI, HCP Terraform, Provider Use), and HCP Terraform (Automate your infrastructure provisioning at any scale, Try HCP Terraform for free).

macOS

Package manager

```
brew tap hashicorp/tap
brew install hashicorp/tap/terraform
```

Binary download

Architecture	Version	Download
AMD64	Version: 1.10.4	Download
ARM64	Version: 1.10.4	Download

Windows

Binary download

Architecture	Version	Download
386	Version: 1.10.4	Download
AMD64	Version: 1.10.4	Download

Linux

Package manager

[Ubuntu/Debian](#) CentOS/RHEL Fedora Amazon Linux Homebrew

```
wget -O - https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com/ $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update
sudo apt install terraform
```

2. Basic Configuration File

```
main.tf > ...  
1 terraform {  
2   required_providers {  
3     aws = {  
4       source = "hashicorp/aws"  
5       version = "~> 5.0"  
6     }  
7   }  
8  
9   required_version = ">= 0.12"  
10 }  
11  
12 provider "aws" {  
13   region = "us-east-1"  
14 }
```

Terraform Block: Specify Providers needed for the Infrastructure

Provider Block: Setup Provider (e.g., region, credentials, etc.)

3. Terraform Init

```
[tillsteinert@Tills-Mac-mini terraform % terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "> 5.0"...
- Installing hashicorp/aws v5.84.0...
- Installed hashicorp/aws v5.84.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
[tillsteinert@Tills-Mac-mini terraform % ls -la
.
..
.terraform
.terraform.lock.hcl
main.tf
```

Cache for downloaded Providers



Lock Provider Versions



- Important Commands
 - terraform init: Initialize Terraform in current working directory
 - terraform plan [--out=<file>]: Creates an execution plan
 - terraform apply: Apply generated plan to setup infrastructure
 - terraform destroy: Destroy all Object created by Terraform

4. Questions