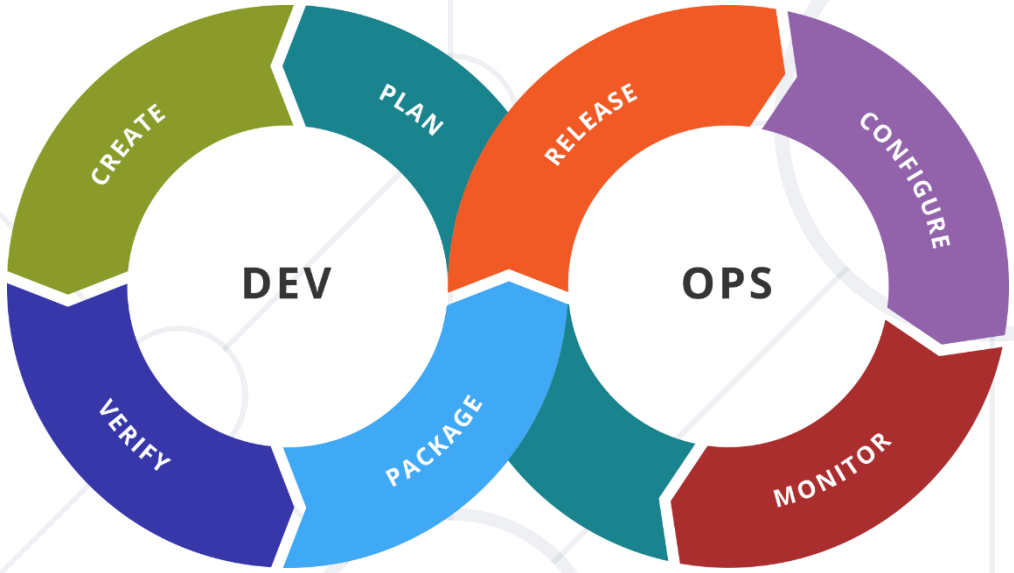
Terraform Fundamentals

Infrastructure as Code with Terraform



SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

You Have Questions?

Software University

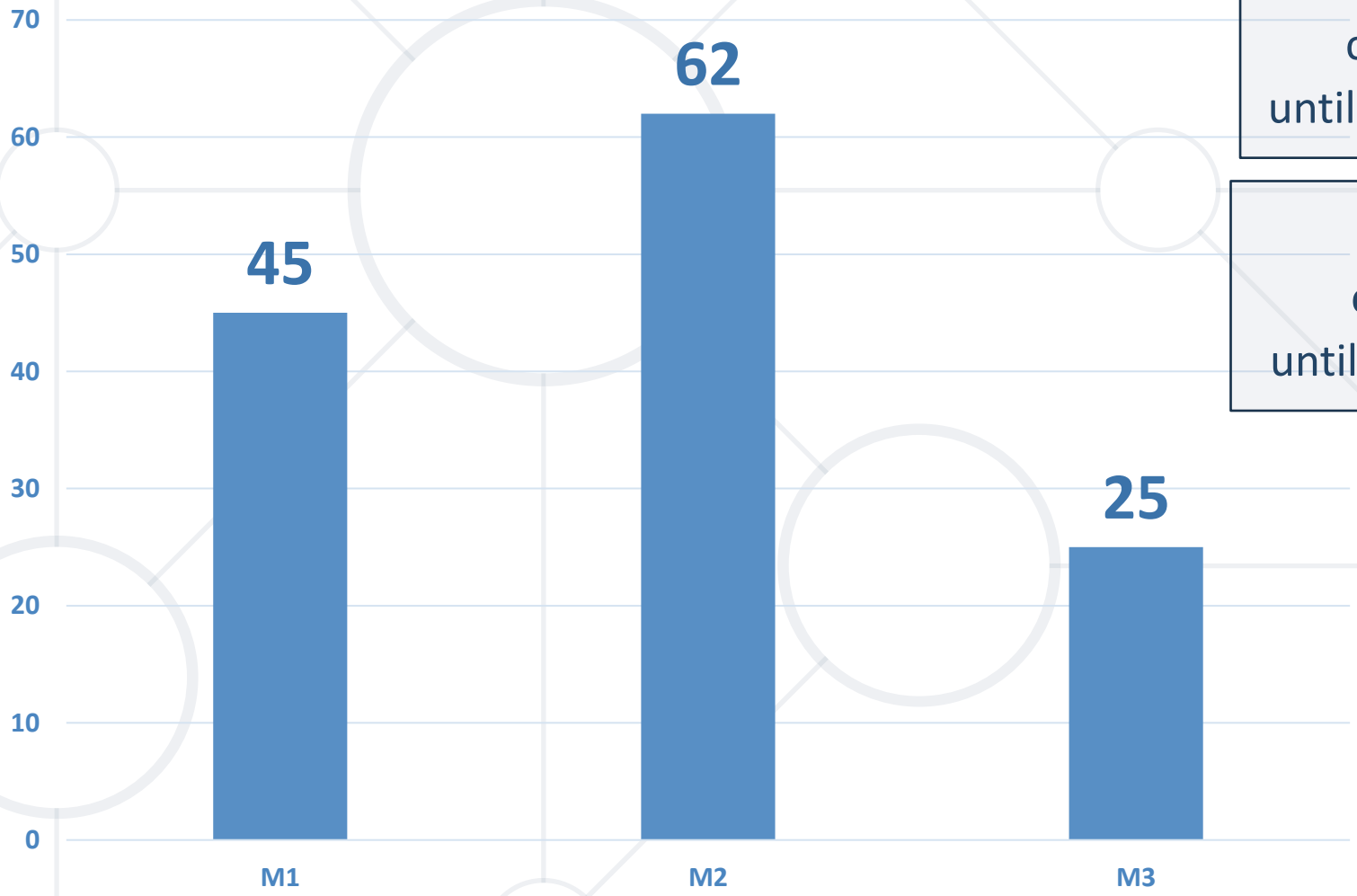
sli.do

#DevOps-CI

facebook.com/groups/

containerizationandinfrastructurejune2025

Homework Progress



Solutions for M3
can be submitted
until 23:59 on 01.07.2025

Solutions for M4
can be submitted
until 23:59 on 08.07.2025

Previous Module (M3)

Quick Overview

What We Covered

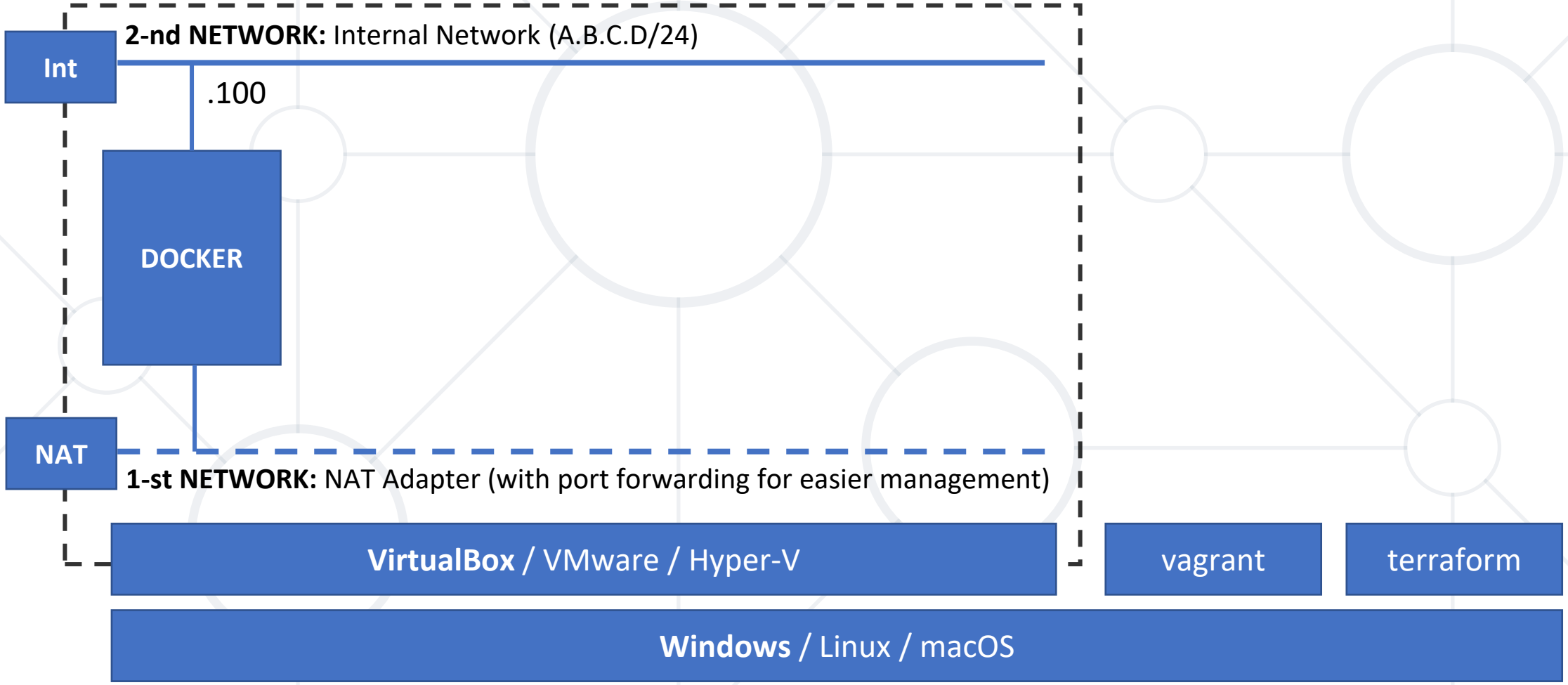
- Distributed Applications
  - Linking Methods
  - Docker Compose
- Docker Clusters
  - Components and Principles
  - Docker Swarm
- Podman

This Module (M4)

Topics and Lab Infrastructure

Table of Contents

Lab Infrastructure



Infrastructure as Code

Overview

Introduction

Infrastructure as code (IaC) is the process of **managing** and **provisioning** computer data centers through machine-readable **definition files**, rather than physical hardware configuration or interactive configuration tools

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

Terraform by HashiCorp

Overview

Introduction

- Terraform is a tool for
  - Building
  - Changing
  - Versioning

  } Infrastructure
- And it is doing it
  - Safely
  - Efficiently
- It can manage both **cloud providers** and **on-premise solutions**

Features

- **Infrastructure as Code**
  - Described using a high-level configuration syntax
- **Execution Plans**
  - Created during planning phase. It shows what would be done
- **Resource Graph**
  - Dependency is tracked and if possible, execution is parallelized
- **Change Automation**
  - Changes can be applied with minimal human interaction

Comparison with CM Tools

- Provides flexible **abstraction** of **resources** and **providers**
- It covers physical hardware, virtual machines, containers, etc.
- Configuration management tools expect that the target exists
- Terraform enables and cooperates with CM tools

Alternatives

- **Universal** (general purpose)
  - Pulumi
  - OpenTofu
- **Specialized**
  - AWS CloudFormation
  - Azure Resource Manager
  - Google Cloud Deployment Manager

HashiCorp Configuration Language (HCL)

Software University

```
# An AMI
variable "ami" {
  description = "the AMI to use"
}

/* A multi
   line comment. */
resource "aws_instance" "web" {
  ami               = "${var.ami}"
  count             = 2
  source_dest_check = false

  connection {
    user = "root"
  }
}
```

https://www.terraform.io/docs/configuration/syntax.html

Interpolation Syntax

Software University

- Interpolations are wrapped in `${}`, such as `${var.foo}`
- It allows you to reference variables, attributes of resources, call functions, etc.
- Simple math is possible - `${count.index + 1}`
- Conditionals are supported - CONDITION ? TRUEVAL : FALSEVAL
- Interpolation can be escaped with `$${foo}`
- More information here:

https://www.terraform.io/docs/configuration/interpolation.html

Beware of changes between versions

Building Blocks (1)

- **Configuration files**
  - Must end with **.tf** (or **.tf.json**)
  - Are loaded in alphabetical order
  - Content is appended **not merged**
- **Override files**
  - Name should be **override** or end with **_override**
  - Loaded after the non-override files in alphabetical order
  - Content is **merged**

* Terraform files are declarative
* Order of variables, resources, etc. doesn't matter

Building Blocks (2)

- **Resources**
  - Play **central part** in our infrastructure

```
resource "aws_instance" "web" {
  ami           = "ami-408c7f28"
  instance_type = "t1.micro"
}
```

  - Combination of **type** and **name** must be **unique**
  - Have also **meta-parameters**, **timeouts**, **dependencies**
  - There are also **connection blocks** and **provisioners**

Building Blocks (3)

- **Data Sources**
  - Used to fetch or calculate external information
  - Can be used to drive the infrastructure creation process
- **Providers**
  - Responsible for the lifecycle of the resources
  - Multiple providers are allowed
  - External components (incl. 3rd party) with separate lifecycle

Building Blocks (4)

- **Variables**
  - Input variables serve as parameters for modules
  - When used in root module
    - Can be set from CLI
    - Or with environment variables
- **Outputs**
  - Define values that will be highlighted to the end user
  - Provide a way to easily extract and query resources information

Building Blocks (5)

- **Local Values**
  - Assign name to an expression that can be used multiple times
- **Modules**
  - Used for modularization and encapsulation of resources
- **Terraform**
  - Used to configure Terraform itself

Installation

- All **major operating systems** are supported
- Just go to https://www.terraform.io/downloads.html
- Older versions are also available
- When upgrading, check the Upgrade Guide for possible issues
- For extensions (modules) check here:

  https://registry.terraform.io/
- Additionally, install at least **syntax highlighting** plugin
- VS Code is a good option with lots of extensions

Practice: See It in Action

Live Demonstration in Class

Terraform and Docker

Explore Basic Concepts

Introduction

- Dedicated **Docker** provider
- Used to interact with Docker **containers** and **images**
- Uses Docker API, it can work with **Docker** and **Docker Swarm**
- Docker Resources
  - **docker_container**, **docker_image**, **docker_network**, **docker_volume**
- Swarm Resources
  - **docker_config**, **docker_secret**, **docker_service**

Practice: See It in Action

Live Demonstration in Class

Terraform and the Cloud

From 0 to 100 in 1 Hour ;)

Introduction

- Dedicated **Amazon Web Services** provider
- Provides support for many resources for AWS
- Should be configured with the proper credentials
- The following methods are supported, in this order:
  - Static credentials
  - Environment variables
  - Shared credentials file
  - EC2 Role

Practice: See It in Action

Live Demonstration in Class

Summary

- Terraform is
  - Tool for infrastructure provisioning
  - Support many platforms
  - Declarative approach
  - Solutions can be modularized
  - Extensible with 3rd party modules
- Terraform is **not**
  - Competitor of Ansible, Chef, Puppet, or Salt
  - Instead, they can be used in combination

Resources

Software University

- Terraform site
  https://www.terraform.io
- Terraform Module Registry
  https://registry.terraform.io/
- Terraform documentation
  https://www.terraform.io/docs/index.html
- Visual Studio Code
  https://code.visualstudio.com/
- VIM Terraform syntax highlighting
  https://github.com/hashivim/vim-terraform

SoftUni Diamond Partners

Software University

Coca-Cola HBC Bulgaria

Postbank

top EMPLOYER България Bulgaria 2025

FOR A BETTER WORLD OF WORK

SUPER HOSTING .BG

createX

DRAFT KINGS

THE CROWN IS YOURS

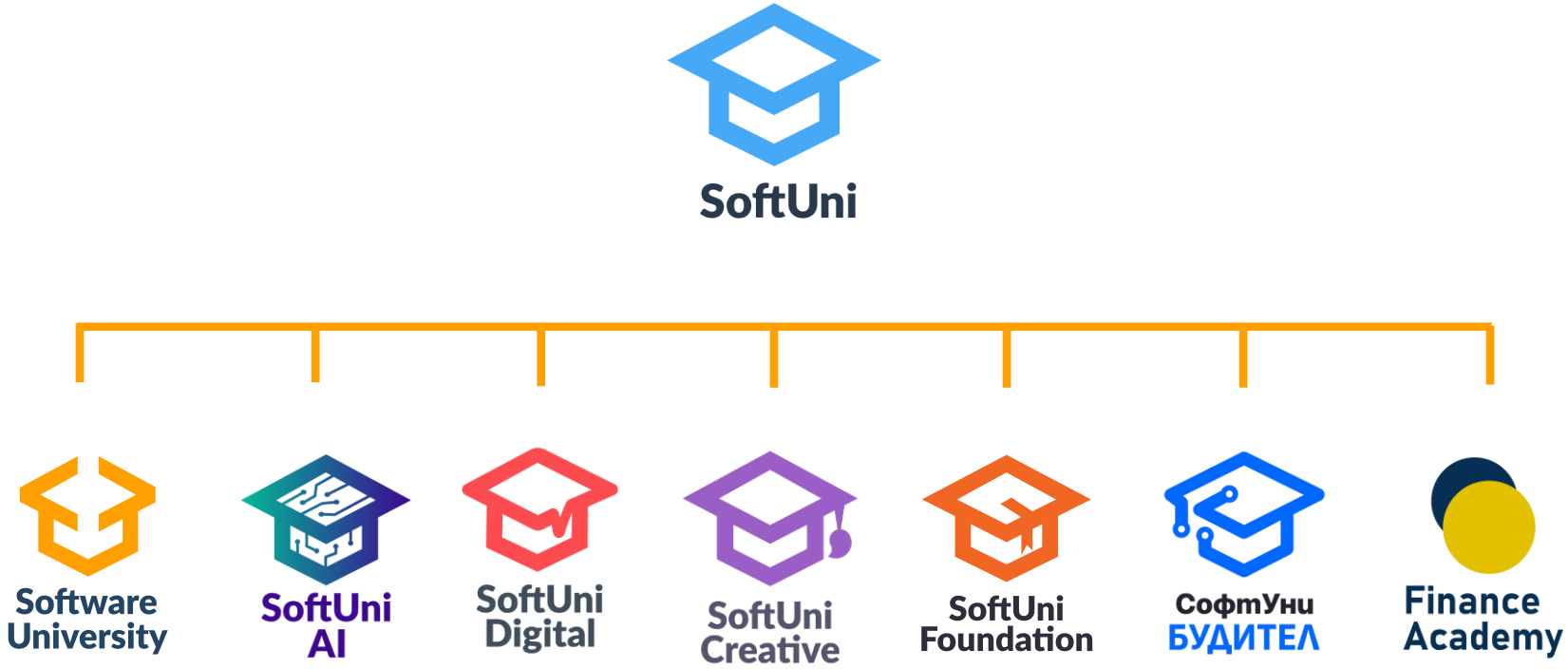
INDEAVR

Serving the high achievers

encorp.io

VIVACOM

Questions?



Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Pr ofession and Job for Software Developers
  - softuni.bg, softuni.org
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://softuni.org
- © Software University – https://softuni.bg