

Evaluatie van Pulumi als IaC-tool binnen de AWS-omgeving van Brussels Airport Company



**HO
GENT**

Context

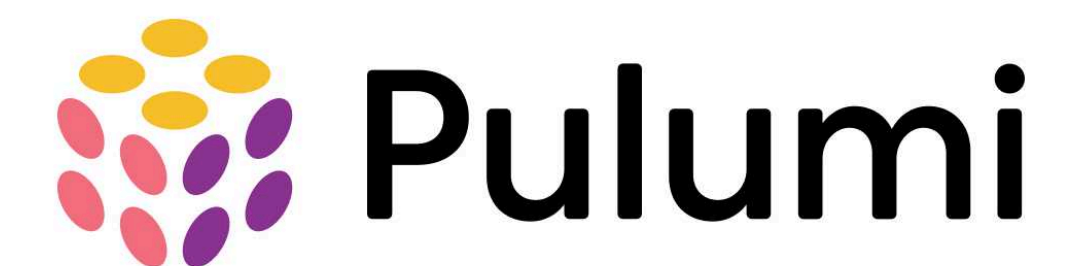
Hoe kwam het onderwerp tot stand?

Stage als Cloud Engineer bij Brussels Airport

Cloud Foundation team, verantwoordelijk voor het opzetten en beheer van de cloud

Momenteel Terraform als IaC, maar open om nieuwe tools te bekijken

Is Pulumi interessant voor ons?



Onderzoeksvraag

Kan **Pulumi Terraform vervangen als IaC-tool binnen
Brussels Airport company voor het beheer van AWS?**

Deelvragen

Hoe is de huidige **AWS-architectuur** van Brussels
Airport Company opgebouwd?

Deelvragen

Welke problemen worden er ervaren bij het gebruik van Terraform?

Deelvragen

Hoe werkt Pulumi in **vergelijking** met Terraform, op vlak van snelheid, debugging en herbruikbaarheid van code?

Methodologie

Verloop van het onderzoek



Fase 1

Analyse van de huidige
AWS omgeving

Fase 2

Literatuurstudie

Fase 3

PoC uitwerken

Analyse

Studie van de huidige AWS omgeving

Multi account: Alle resources worden onderverdeeld in accounts

AWS Control Tower: Dienst om geautomatiseerd een veilige landing zone op te zetten

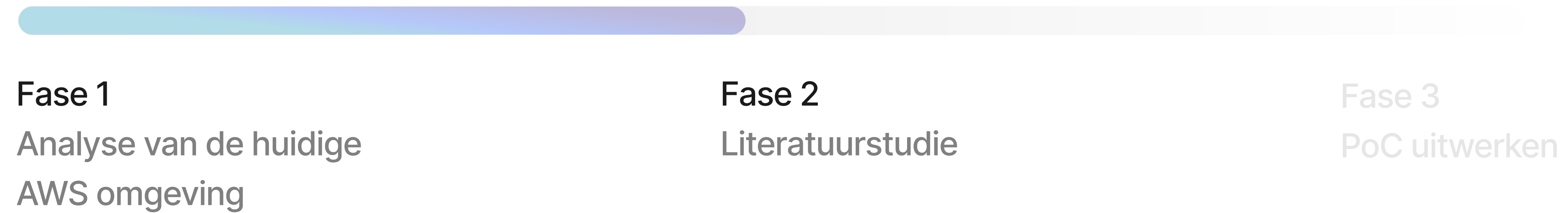
OUs: De accounts zijn onderverdeeld in OU's om ze logisch te bundelen

CI/CD: Veilig uitrollen van code naar productie via pipelines



Methodologie

Verloop van het onderzoek



Pulumi

1

Code, gebruik talen zoals:
Python, TS, GO, Java en
C#

```
index.html
34 const events = [
35   'dragenter',
36   'dragleave', // to allow drop
37   'dragover',
38   'drop'
39 ];
40 events.forEach(e => {
41   fileDropZone.addEventListener(e, (ev) => {
42     ev.preventDefault();
43     if (ev.type === 'dragenter') {
44       fileDropZone.classList.add('solid-b
45     }
46     if (ev.type === 'dragleave') {
47       fileDropZone.classList.remove('
48     }
49     if (ev.type === 'drop') {
50       fileDropZone.classList.remove('
51     }
52   });
53 });
```

2

Flexibiliteit, betere
herbruikbaarheid van code
en complexere logica



Pulumi

3

Geavanceerde
testmogelijkheden



1

Code, gebruik talen zoals:
Python, TS, GO en C#

```
Index.html
34 const events = [
35   'dragenter',
36   'dragleave',
37   'dragover', // to allow drop
38   'drop'
39 ];
40 events.forEach(e => {
41   fileDropZone.addEventListener(e, (ev) => {
42     ev.preventDefault();
43     if (ev.type === 'dragenter') {
44       fileDropZone.classList.add('solid-t
45     }
46     if (ev.type === 'dragleave') {
47       fileDropZone.classList.remove('
48     }
49     if (ev.type === 'drop') {
50       fileDropZone.classList.remove('
51     }
52   });
53 });
```

2

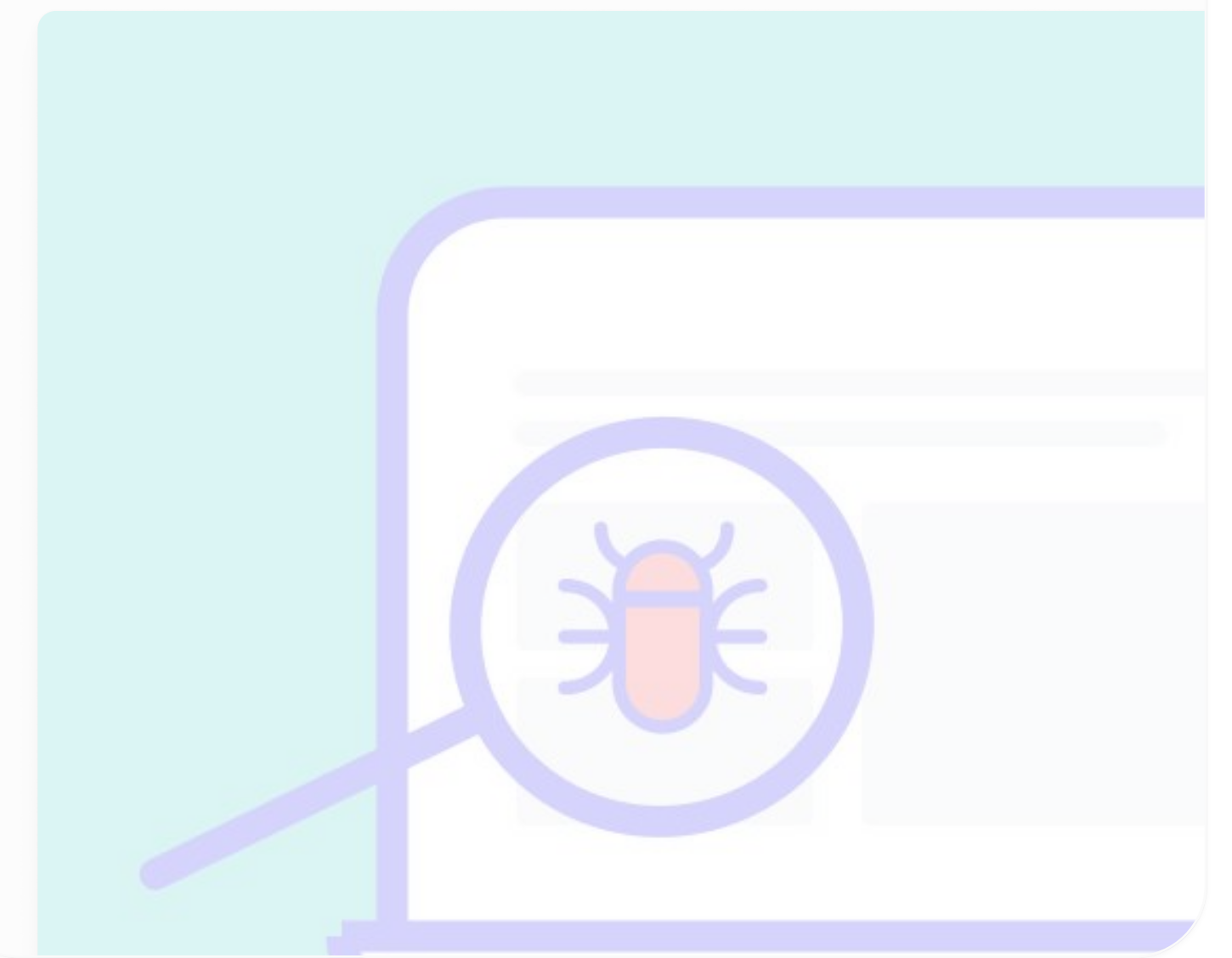
Flexibiliteit, betere
herbruikbaarheid van code
en complexere logica



Pulumi

3

Geavanceerde
testmogelijkheden



1

Code, gebruik talen zoals:
Python, TS, GO en C#

```
Index.html
34 const events = [
35   'dragenter',
36   'dragleave',
37   'dragover', // to allow drop
38   'drop'
39 ];
40 events.forEach(e => {
41   fileDropZone.addEventListener(e, (ev) => {
42     ev.preventDefault();
43     if (ev.type === 'dragenter') {
44       fileDropZone.classList.add('solid-t
45     }
46     if (ev.type === 'dragleave') {
47       fileDropZone.classList.remove('
48     }
49     if (ev.type === 'drop') {
50       fileDropZone.classList.remove('
51     }
52   });
53 });
```

2

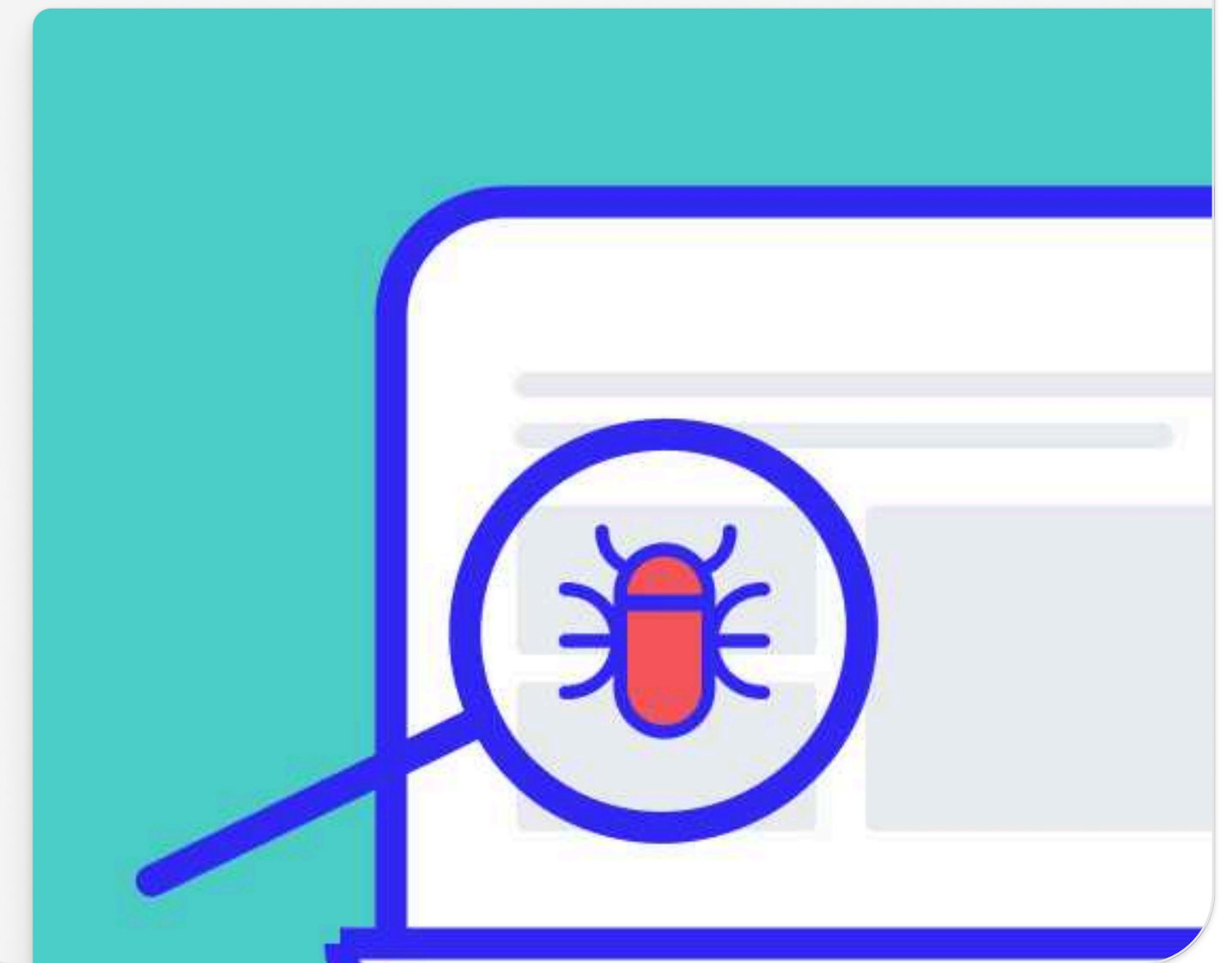
Flexibiliteit, betere
herbruikbaarheid van code
en complexere logica



Pulumi

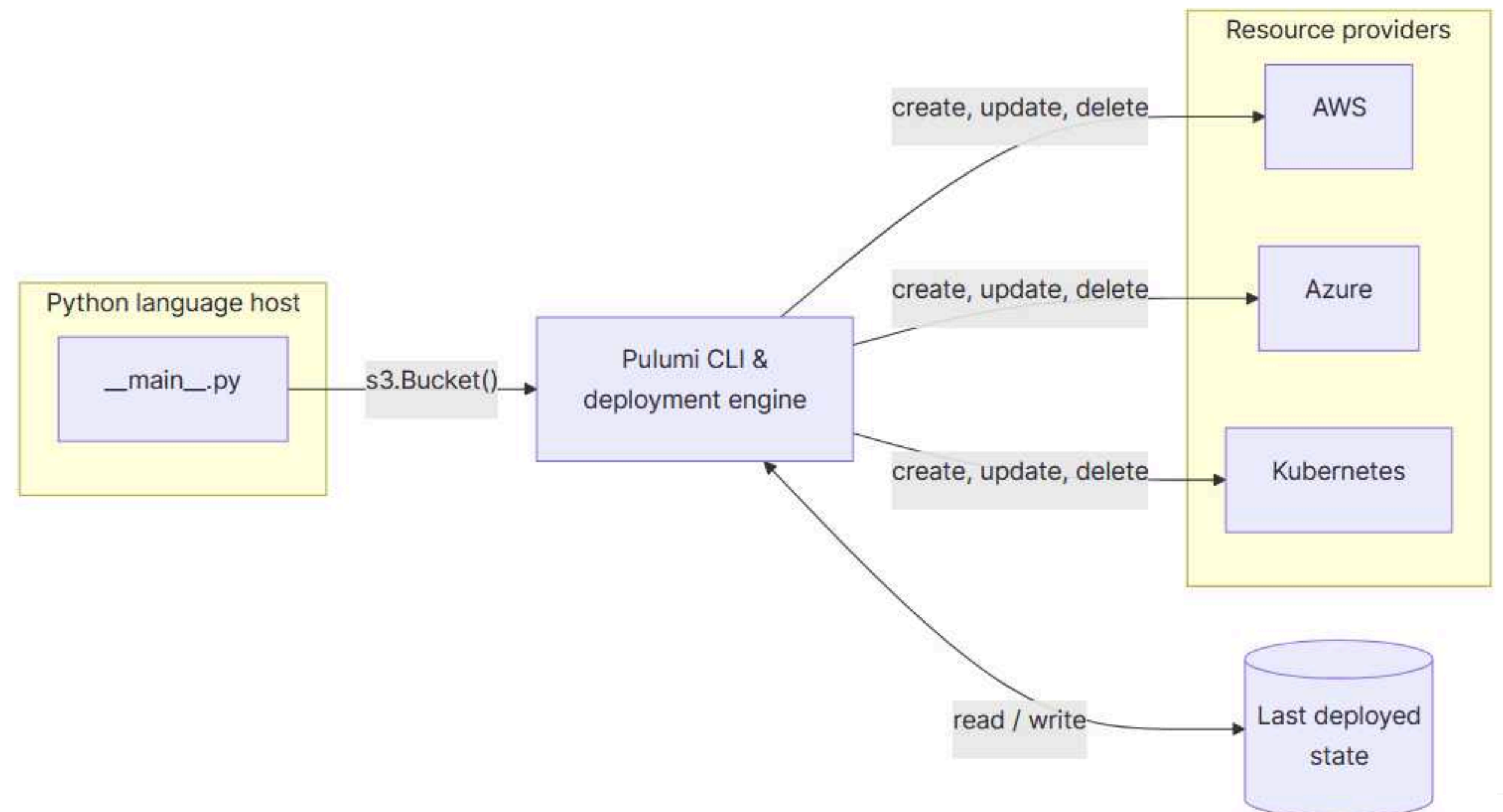
3

Geavanceerde
testmogelijkheden



Hoe werkt Pulumi?

1. Code schrijven
2. De code gaat door de language host: detecteert en registreert resources
3. Deployment engine: vergelijkt gewenste state met huidige state
4. Providers: communiceren met de cloud



Code voorbeelden

```
main.py

import pulumi
import pulumi_aws as aws

example = aws.s3.Bucket("example",
    bucket="my-tf-test-bucket",
    tags={
        "Name": "My bucket",
        "Environment": "Dev",
    })
```

```
main.py

import pulumi
import pulumi_aws as aws

size = 't2.micro'
ami = aws.ec2.get_ami(most_recent="true",
    owners=["137112412989"],
    filters=[{"name": "name", "values": ["amzn2-ami-hvm-*"]})

group = aws.ec2.SecurityGroup('webserver-secgrp',
    description='Enable HTTP access',
    ingress=[
        { 'protocol': 'tcp', 'from_port': 80, 'to_port': 80, 'cidr_blocks': ['0.0.0.0/0'] }
    ])

server = aws.ec2.Instance('webserver-www',
    instance_type=size,
    vpc_security_group_ids=[group.id], # reference security group from above
    ami=ami.id)

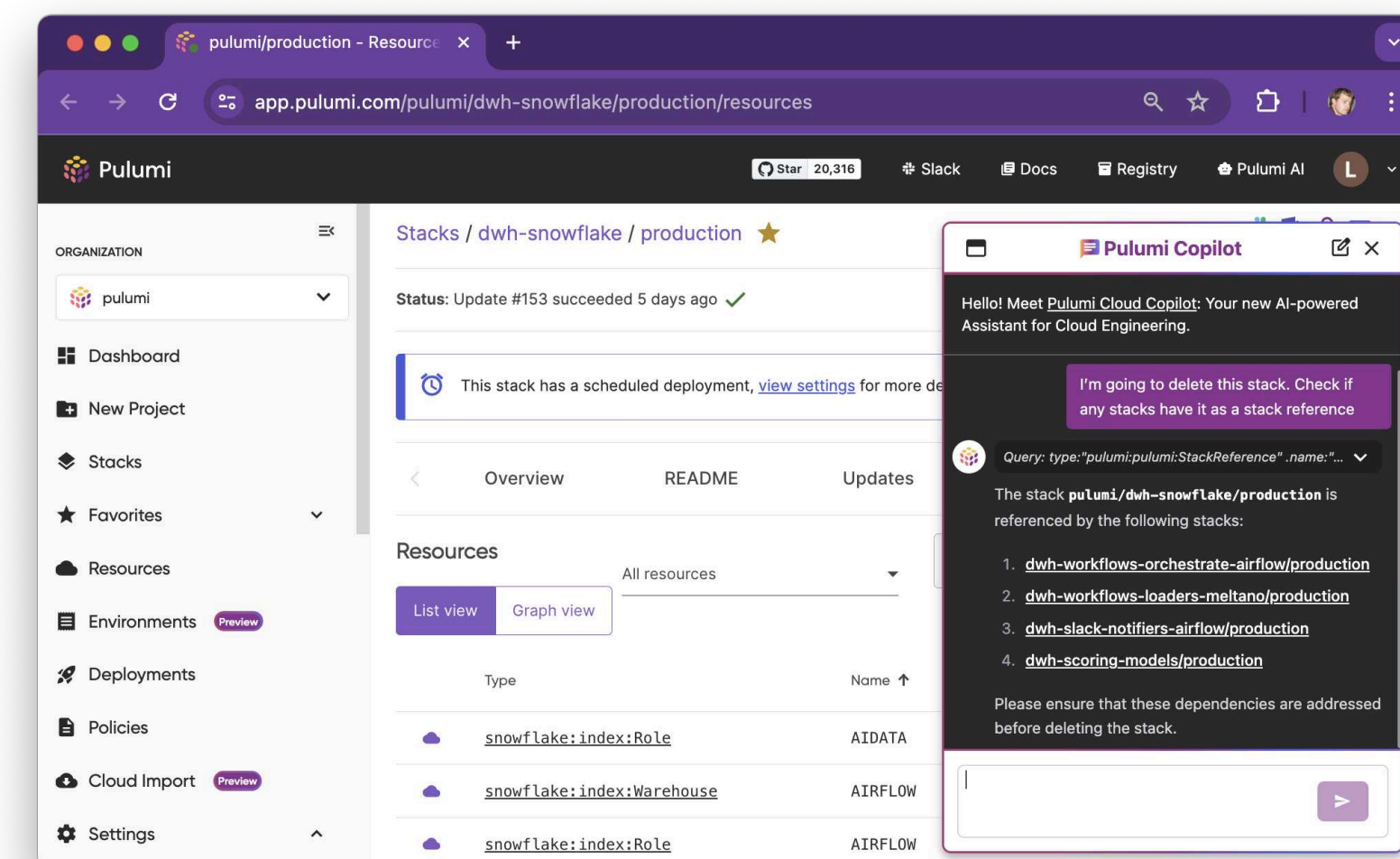
pulumi.export('publicIp', server.public_ip)
pulumi.export('publicHostName', server.public_dns)
```

Managed backend beheer

Pulumi cloud

- ✓ Managed state met versiegeschiedenis
- ✓ Eenvoudig secretsbeheer
- ✓ Automatische drift detectie
- ✓ AI assistent, NEO
- ✓ Cloud resource inventory inclusief niet-Pulumi resources

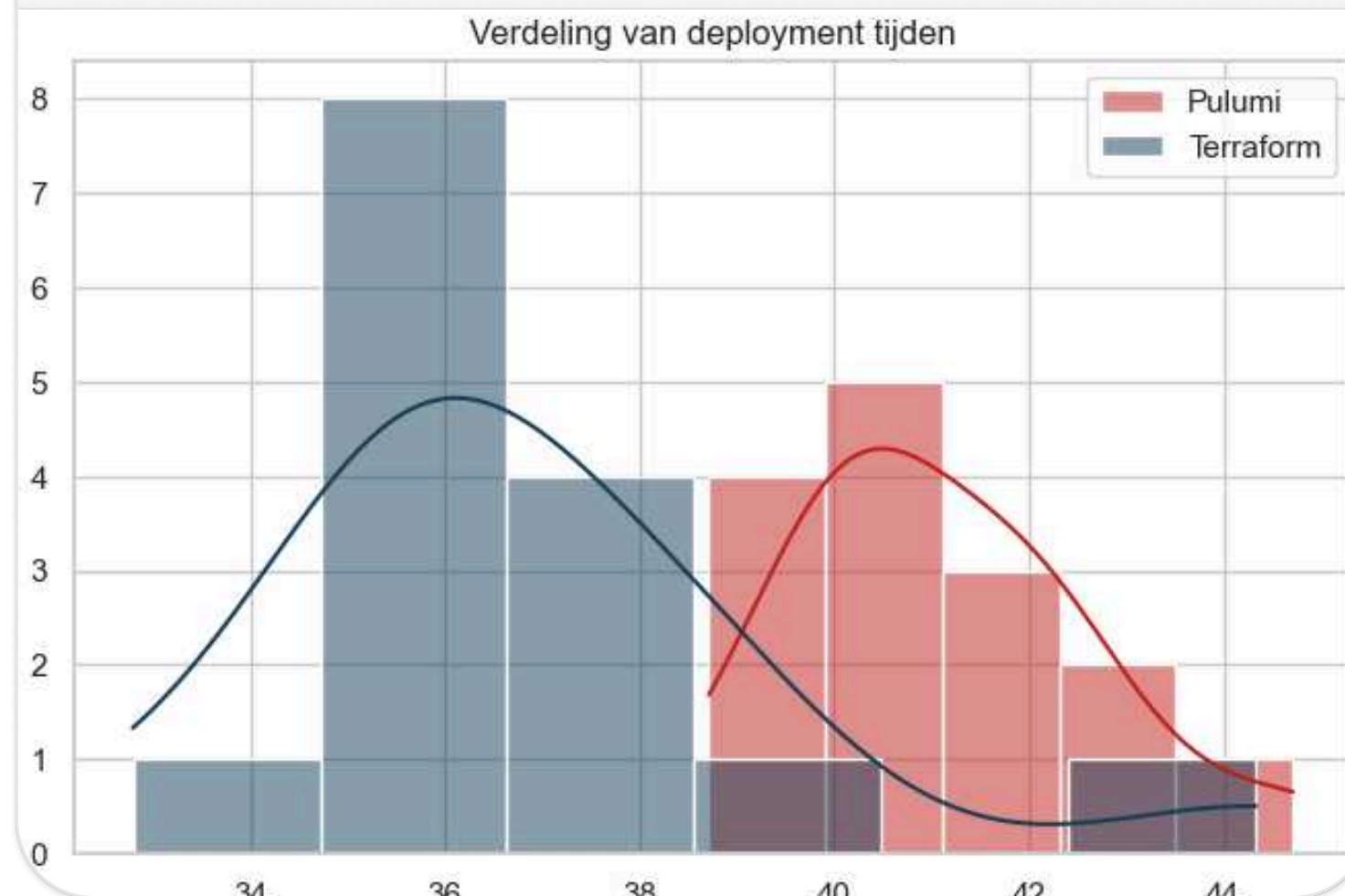
- ✗ Duur, zeker voor enterprise
- ✗ Minder controle
- ✗ Vendor lock-in



Terraform

1

Prestaties: Terraform is significant sneller dan Pulumi



2

Grote community, de tool bestaat langer en wordt vaker gebruikt



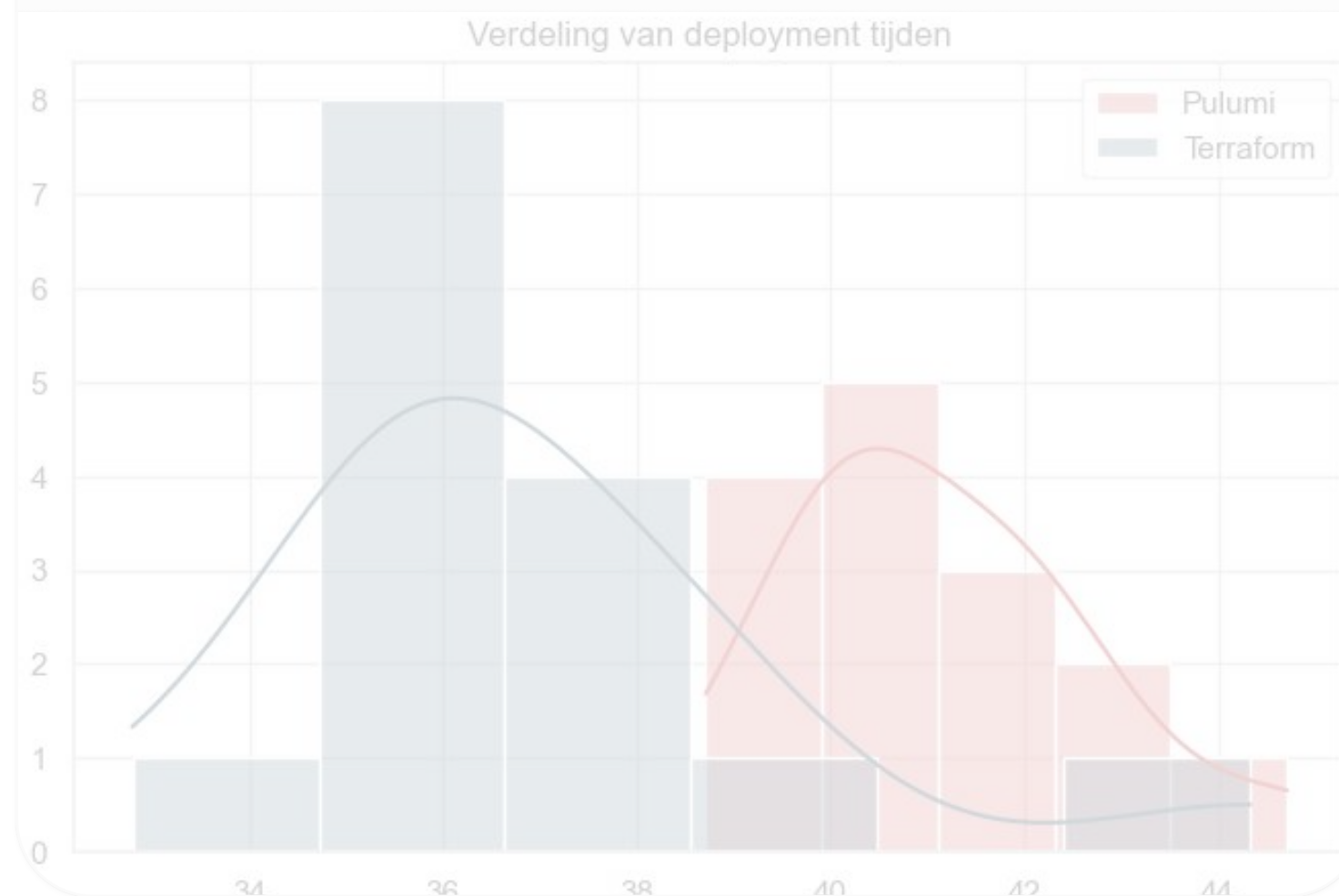
3

Leesbaarheid en learning curve, eenvoudiger te begrijpen en te leren dan Pulumi



1

Prestaties: Terraform is significant sneller dan Pulumi



2

Grote community, de tool bestaat langer en wordt vaker gebruikt



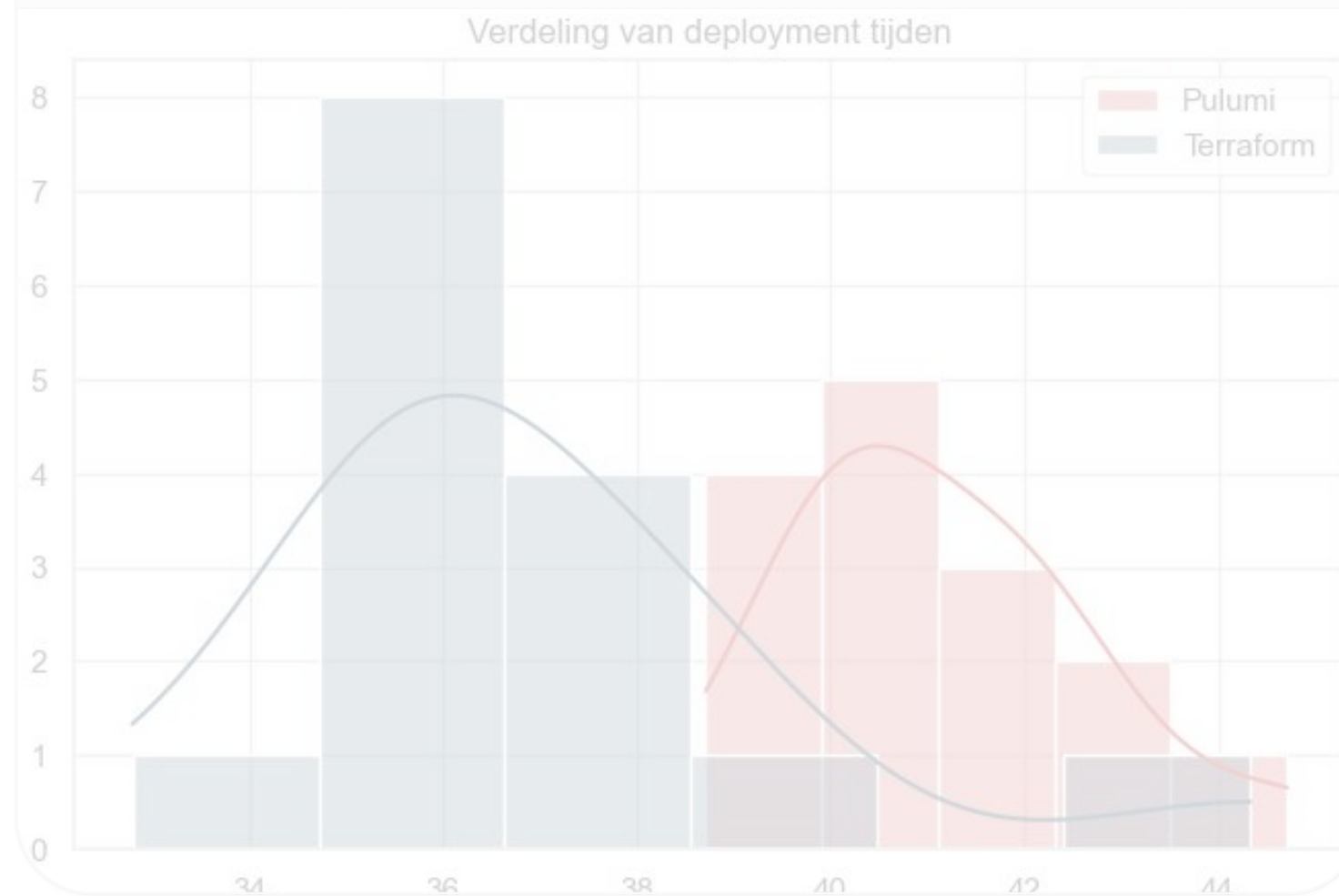
3

Leesbaarheid en learning curve, eenvoudiger te begrijpen en te leren dan Pulumi



1

Prestaties: Terraform is significant sneller dan Pulumi



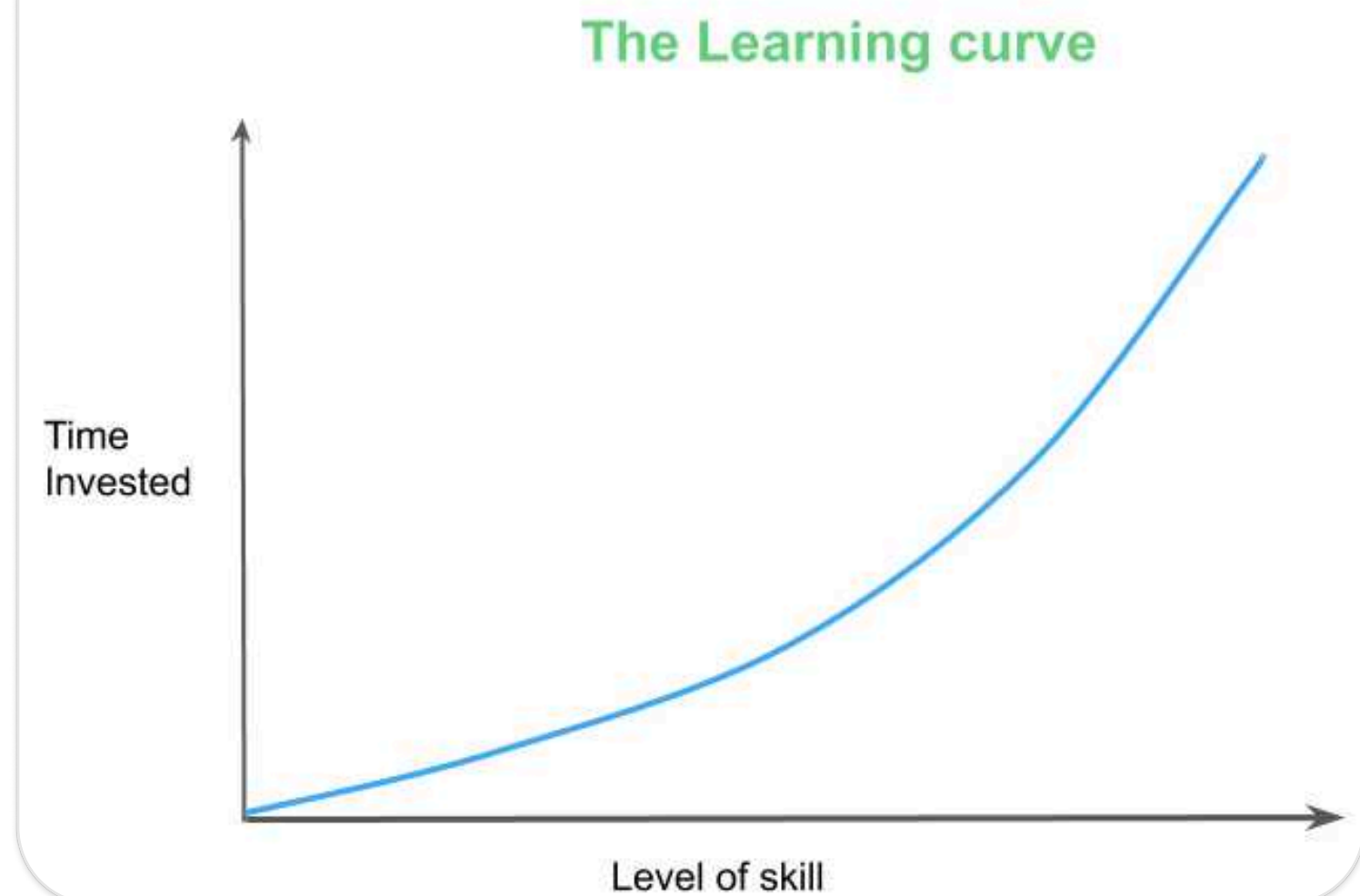
2

Grote community, de tool bestaat langer en wordt vaker gebruikt



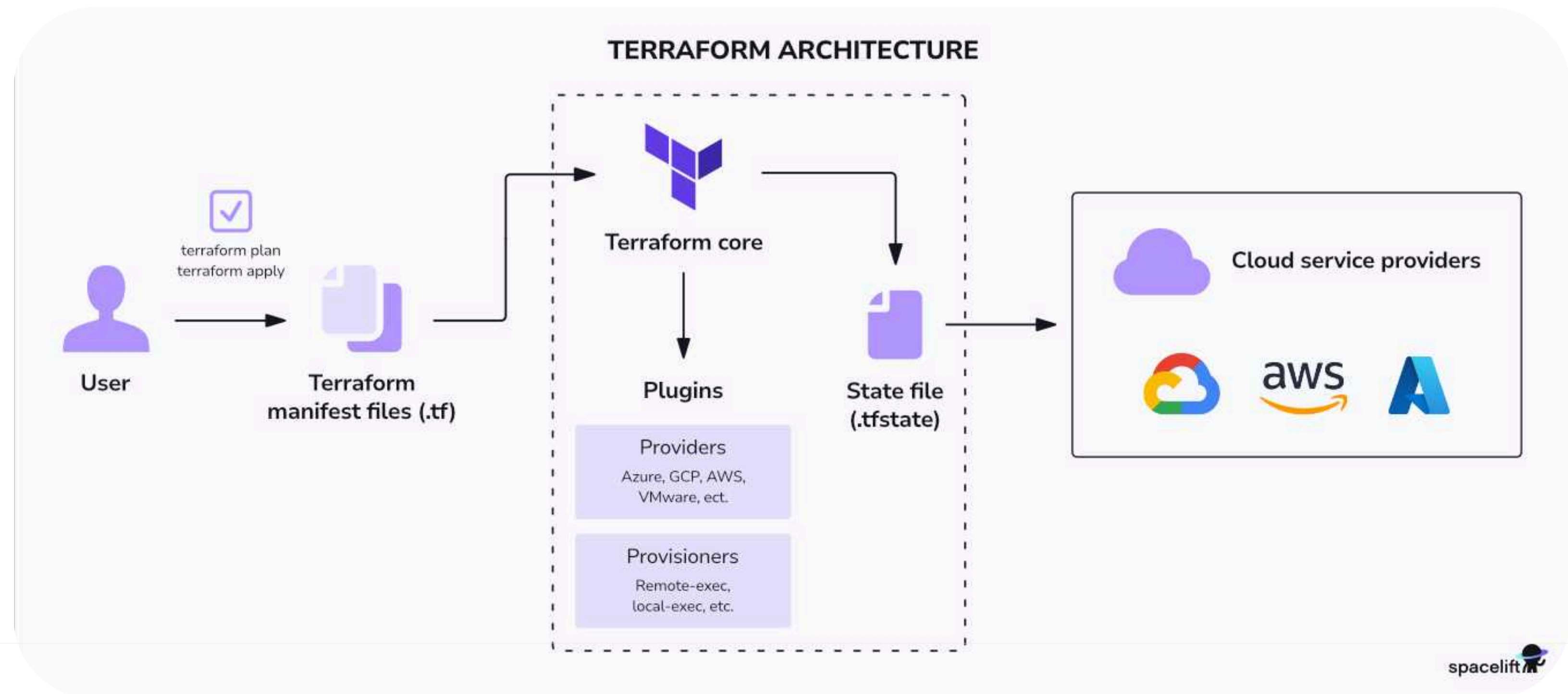
3

Leesbaarheid en learning curve, eenvoudiger te begrijpen en te leren dan Pulumi



Hoe werkt Terraform?

1. Code schrijven in .tf files
2. Terraform core vergelijkt state met desired state
3. Aanmaken, wijzigen en verwijderen van resources a.d.h.v. providers
4. Statefile aangepast



Terraform

Code voorbeelden

```
main.py

resource "aws_s3_bucket" "example" {
  bucket = "my-tf-test-bucket"

  tags = {
    Name          = "My bucket"
    Environment = "Dev"
  }
}
```

```
main.py

# Security Group
resource "aws_security_group" "ec2_sg" {
  name          = "ec2-security-group"
  description   = "Allow SSH access"

  ingress {
    description = "SSH"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["YOUR_PUBLIC_IP/32"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

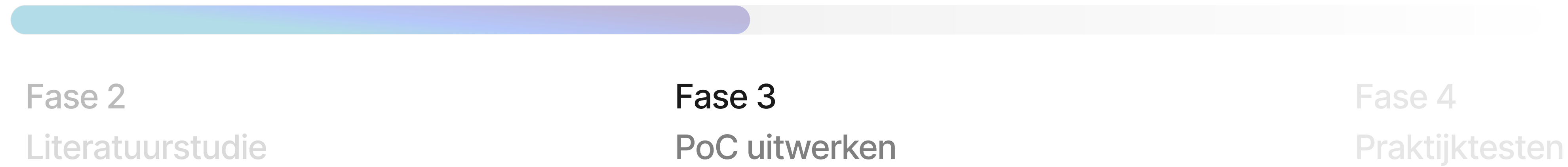
# EC2 Instance
resource "aws_instance" "web_server" {
  ami                = data.aws_ami.amazon_linux.id
  instance_type      = "t3.micro"
  vpc_security_group_ids = [aws_security_group.ec2_sg.id]

  tags = {
    Name = "terraform-ec2"
  }
}

output "instance_public_ip" {
  value = aws_instance.web_server.public_ip
}
```

Methodologie

Verloop van het onderzoek



Proof of Concept

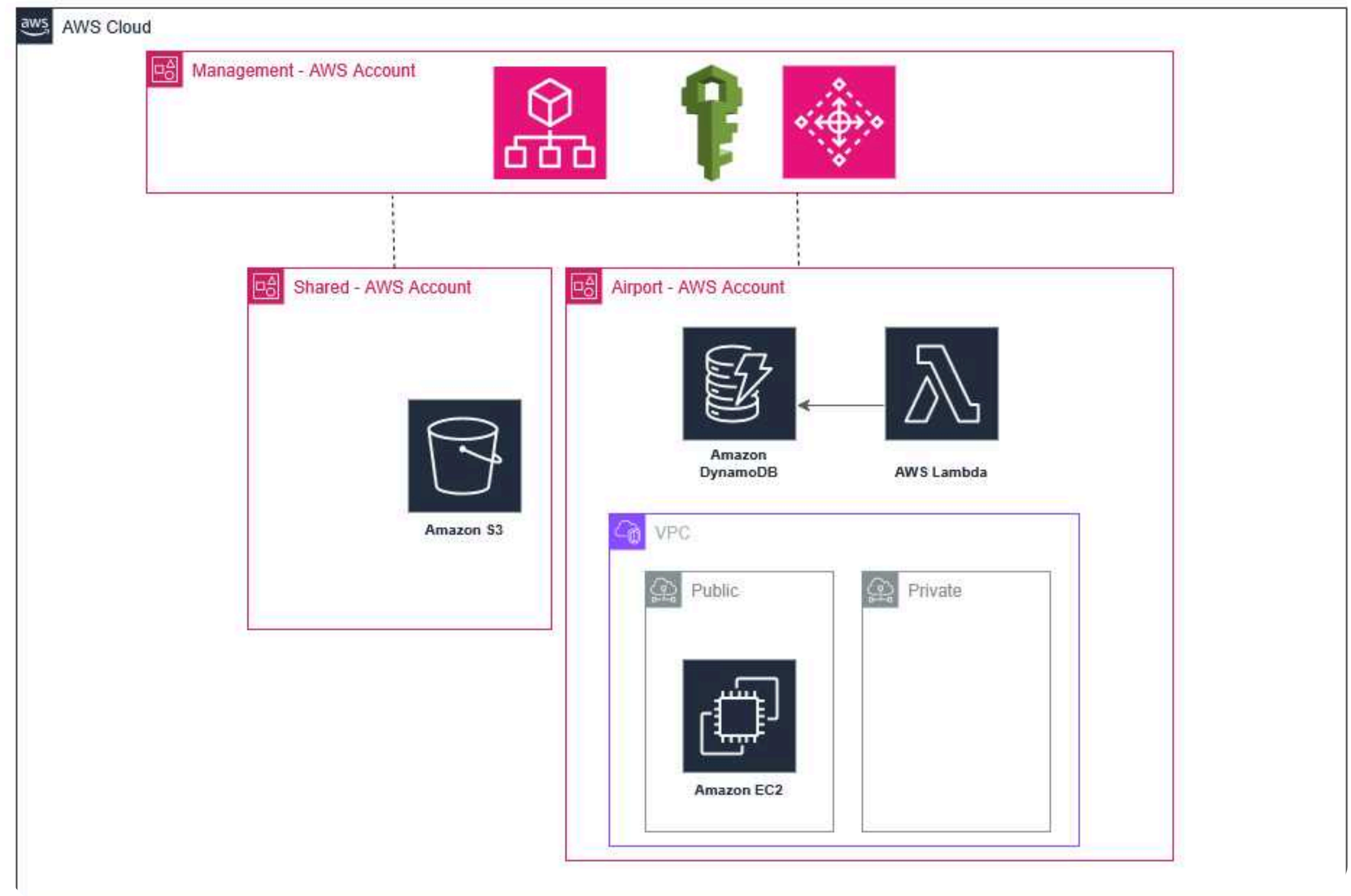
Doel van de PoC is om op een objectieve manier beide tools te kunnen testen, hiervoor werd een volledig nieuwe AWS omgeving opgezet

Multi account

Terraform

Python

Identiek in beide tools



Pulumi demo

Demo

Deploy, destroy en extra's in Pulumi

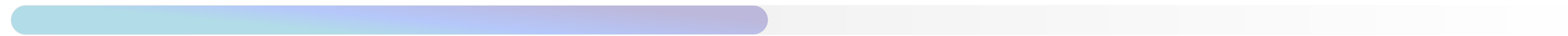
<https://vimeo.com/1198652745>

```
Windows PowerShell
PS C:\Users\gille\OneDrive\Documents\GitHub\bac-aws-pulumi-poc> pulumi up
Previewing update (dev)
View in Browser (Ctrl+O): https://app.pulumi.com/gillesdm/bac-aws-pulumi-poc/dev/preview/07419c6f-985d-4d5e-b67f-df690c87aa95
```

Type	Name	Plan
+ pulumi:pulumi:Stack	bac-aws-pulumi-poc-dev	create...
+ pulumi:providers:aws	shared-provider	create
+ custom:component:S3Component	bac-poc-dev-storage	create
+ aws:s3:Bucket	bac-poc-dev-storage-bucket	create
+ aws:s3:BucketPublicAccessBlock	bac-poc-dev-storage-pab	create
+ aws:s3:BucketPolicy	bac-poc-dev-storage-policy	create
+ aws:s3:BucketObject	bac-poc-dev-storage-fids-html	create
+ pulumi:providers:aws	airport-provider	create
+ custom:component:Ec2Component	bac-poc-dev-compute	create
+ aws:ec2:SecurityGroup	bac-poc-dev-compute-sg	create
+ aws:ec2:Instance	bac-poc-dev-compute-instance	create
+ custom:component:VpcComponent	bac-poc-dev-network	create
+ aws:ec2:Vpc	bac-poc-dev-network-vpc	create
+ aws:ec2:Subnet	bac-poc-dev-network-private-0	create
+ aws:ec2:Subnet	bac-poc-dev-network-public-1	create
+ aws:ec2:InternetGateway	bac-poc-dev-network-igw	create
+ aws:ec2:Subnet	bac-poc-dev-network-private-1	create
+ aws:ec2:Subnet	bac-poc-dev-network-public-0	create
+ aws:ec2:RouteTable	bac-poc-dev-network-rt	create
+ aws:ec2:RouteTableAssociation	bac-poc-dev-network-rta-1	create
+ aws:ec2:RouteTableAssociation	bac-poc-dev-network-rta-0	create
+ custom:component:LambdaApiComponent	bac-poc-dev-serverless	create
+ aws:apigatewayv2:Api	bac-poc-dev-serverless-api	create
+ aws:iam:Role	bac-poc-dev-serverless-role	create
+ aws:dynamodb:Table	bac-poc-dev-serverless-table	create
+ aws:apigatewayv2:Stage	bac-poc-dev-serverless-stage	create
+ aws:iam:RolePolicy	bac-poc-dev-serverless-policy	create
+ aws:lambda:Function	bac-poc-dev-serverless-lambda	create
+ aws:apigatewayv2:Integration	bac-poc-dev-serverless-integration	create
+ aws:lambda:Permission	bac-poc-dev-serverless-permission	create
+ aws:apigatewayv2:Route	bac-poc-dev-serverless-route	create

Methodologie

Verloop van het onderzoek



Fase 3
PoC uitwerken

Fase 4
Praktijktesten

Fase 5
Resultaten

Overzicht

Korte beschrijving
Changes to AWS/Azure/Addex KPI formatting

Beschrijving

Current situation:

- AWS + Azure + Addex cloud security kpis are created each month, but are also dropped for previous files
- Naming convention is not consistent
- Files are not generated at same time of month

Proposal:

- Harmonize file names, starting with YYYY-MM-DD prefix.

Real-life use case

Verzoek vanuit het cybersecurity team om logs met de status van encryptie maandelijks te exporteren naar een Azure storage.

Opgezet met beide tools, resultaten gedeeld met het team.

Ticket succesvol uitgewerkt met de beide tools

The screenshot shows the AWS Lambda console for the function 'aws-config-kpi-aggregator-export'. A green notification bar at the top indicates 'Successfully updated the function "aws-config-kpi-aggregator-export"'. The function overview shows it is triggered by 'EventBridge (CloudWatch Events)'. The function ARN is 'arn:aws:lambda:eu-west-1:123456789012:func:aws-config-kpi-aggregator-export'. The description is empty, and it was last modified 1 hour ago. At the bottom, a green notification bar shows 'Executing function: succeeded (logs)'.

Aanmaken in Pulumi

```
> pulumi up
Previewing update (prod)

View in Browser (Ctrl+O): https://app.pulumi.com/gillesdm/AWS-config-aggregator/prod/previews/02d20c29-3d1a-41be-9e24-d2f6150cc828

Type                Name                Plan
pulumi:pulumi:Stack  AWS-config-aggregator-prod
+ aws:cloudwatch:EventRule  security-kpi-monthly-trigger1  create
+ aws:iam:Role            security-kpi-lambda-role1      create
+ aws:iam:RolePolicy      security-kpi-config-policy1    create
+ aws:iam:RolePolicyAttachment  security-kpi-lambda-basic-exec1  create
+ aws:lambda:Function      security-kpi-config-export1     create
+ aws:cloudwatch:EventTarget  security-kpi-monthly-lambda-target1  create
+ aws:lambda:Permission     allow-eventbridge-monthly-trigger1  create

Resources:
+ 7 to create
8 unchanged

Do you want to perform this update? [Use arrows to move, type to filter]
yes
> no
details
explain
```

Aanmaken in Terraform

Merged feat: added automatic KPI config log export #49
DeMeerleerGilles merged 3 commits into `main` from `ICTCF-3458`

Terraform Plan Results

Summary

Total Changes: 10 to add, 0 to change, 0 to destroy

- [stacks/security/kpi] Create `aws_cloudwatch_log_group.this`
- [stacks/security/kpi] Create `aws_secretsmanager_secret.azure_sas`
- [stacks/security/kpi] Create `module.eventbridge.aws_cloudwatch_event_rule.notifications`
- [stacks/security/kpi] Create `module.eventbridge.aws_cloudwatch_event_target.notifications`
- [stacks/security/kpi] Create `module.lambda.aws_lambda_function.lambda_function`
- [stacks/security/kpi] Create `module.lambda.aws_lambda_permission.allow_trigger["0"]`
- [stacks/security/kpi] Create `module.lambda_role.aws_iam_policy.cf[0]`
- [stacks/security/kpi] Create `module.lambda_role.aws_iam_role.cf`
- [stacks/security/kpi] Create `module.lambda_role.aws_iam_role_policy_attachment.inline[0]`
- [stacks/security/kpi] Create `module.lambda_role.aws_iam_role_policy_attachment.managed["arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole"]`

Cost Estimate

Infracost report

Monthly estimate increased by €0.34

Changed project	Baseline cost	Usage cost*	Total change	New monthly cost
-----------------	---------------	-------------	--------------	------------------

github-actions Bot commented 1 hour ago

Terraform Apply Results

Summary

Status: ✔ Apply successful
Stacks: 1 succeeded, 0 failed

✔ Stack: `stacks/security/kpi`

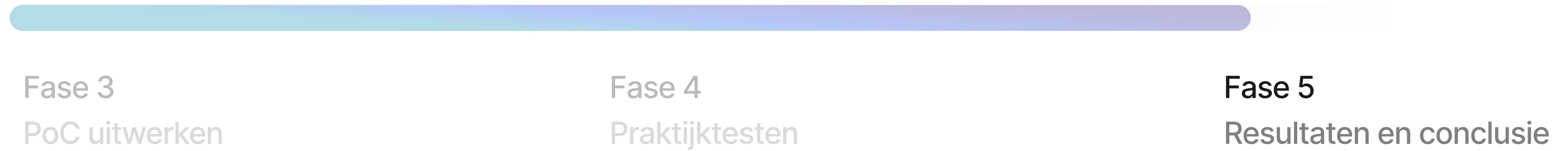
▶ View Apply Output

😊

Pull request successfully merged and closed
You're all set — the branch has been merged.

Methodologie

Verloop van het onderzoek

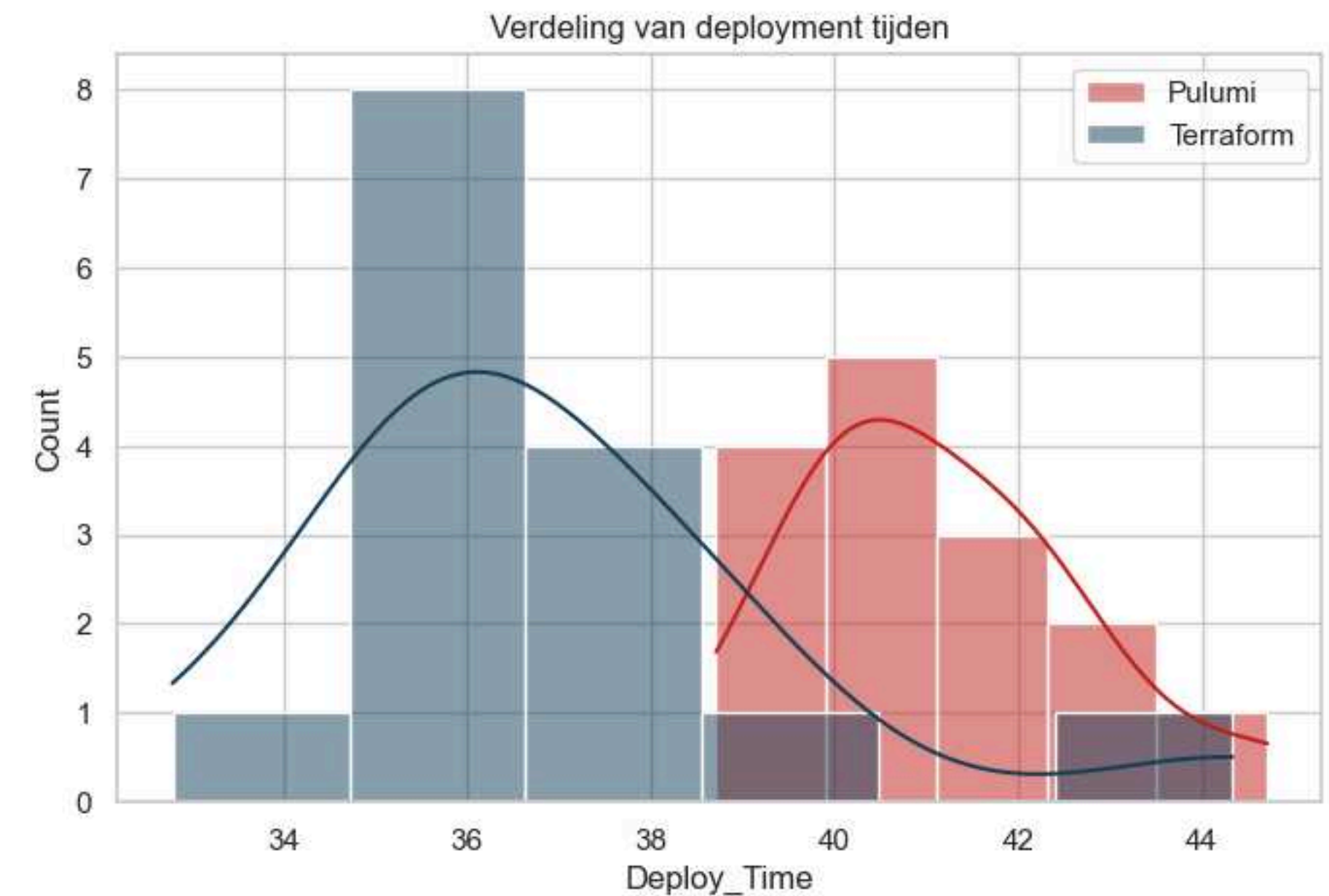
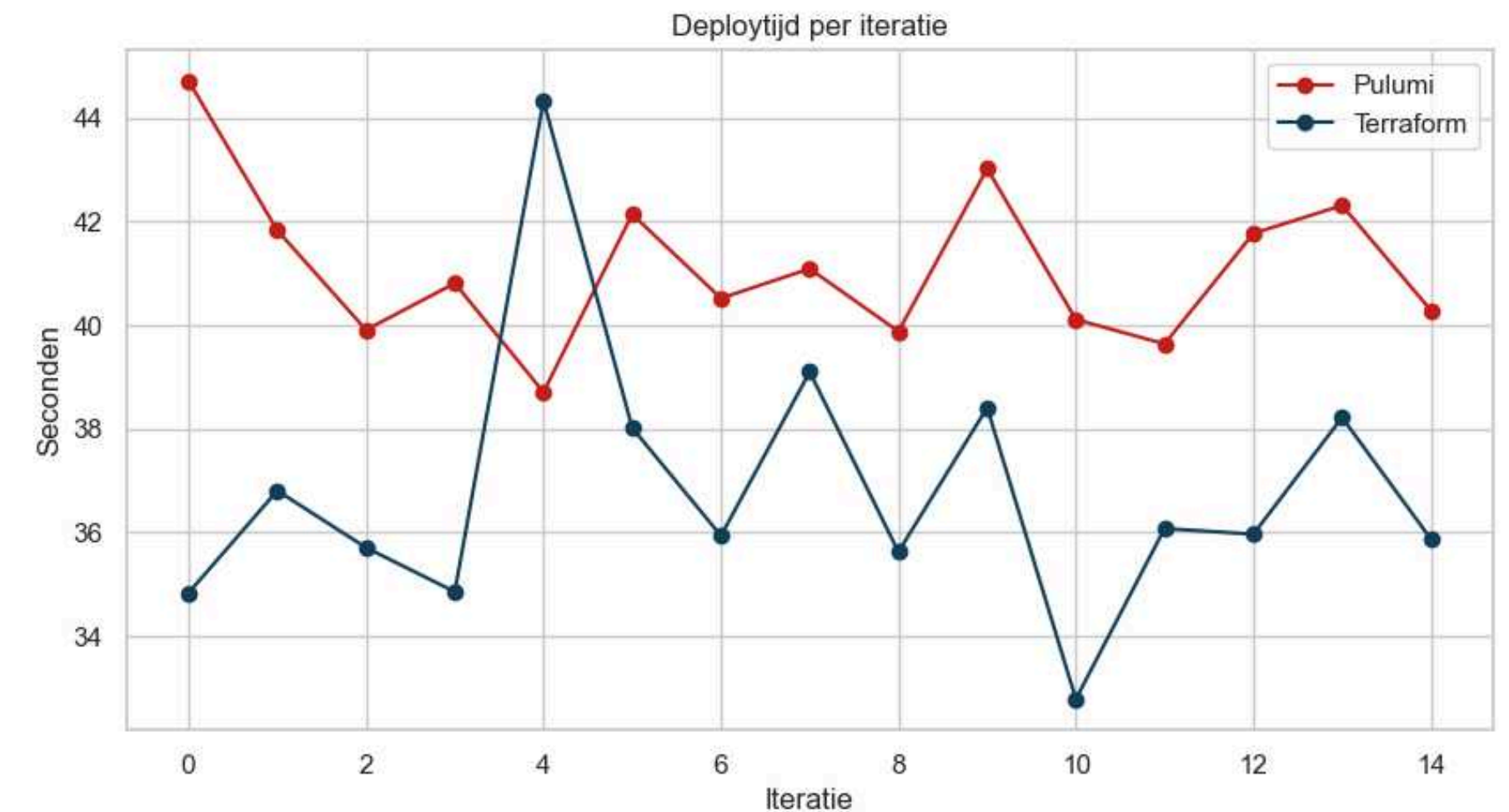


Evaluatiepunt 1

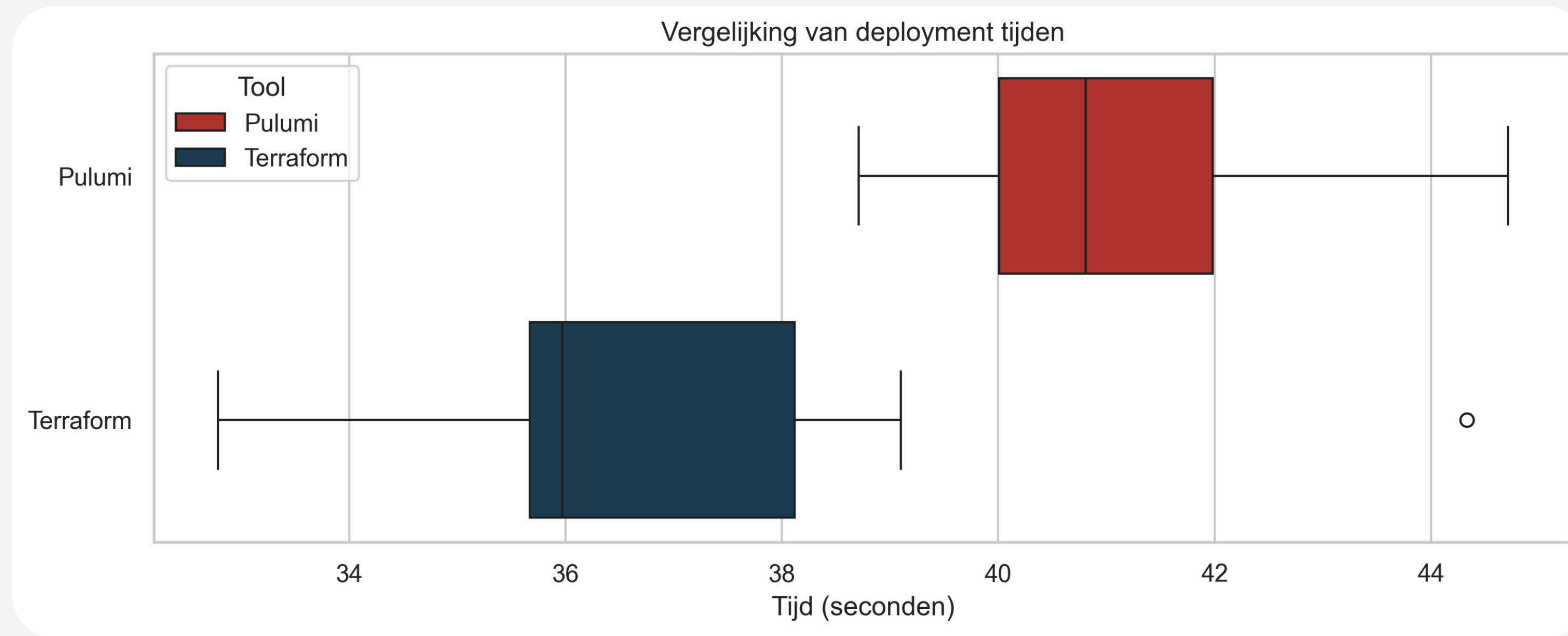
Snelheid en prestaties

- Terraform is statistisch significant sneller dan Pulumi
- Deploy: Terraform was gemiddeld **4,3 seconden sneller** (gemiddeld 36,84s voor Terraform vs. 41,12s voor Pulumi)
- Bij het verwijderen van de infrastructuur was Terraform gemiddeld **5,9 seconden sneller** (25,45s vs. 31,34s)

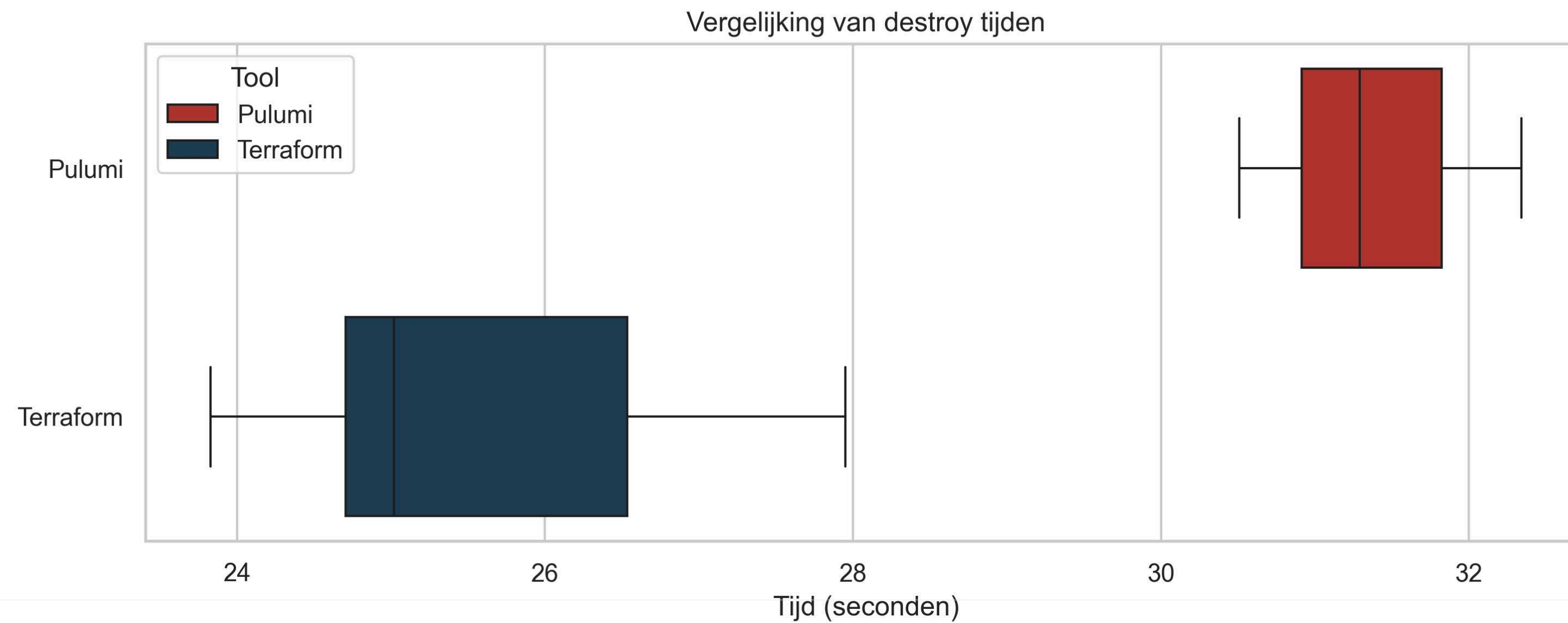
Deze resultaten zijn gebaseerd op 15 testiteraties waarbij dezelfde omgeving werd opgebouwd en afgebroken



Deploy



Destroy



Evaluatiepunt 2

Codecomplexiteit

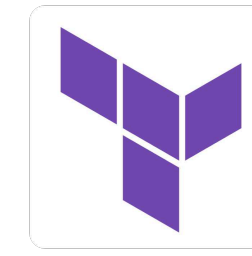
- Terraform gebruikte minder lijnen code dan Pulumi in de uitgevoerde testen, ook zijn er minder files nodig
- Pulumi biedt dankzij de programmeertalen meer flexibiliteit om complexe logica te implementeren, bovendien zijn er meer opties voor herbruikbaarheid van code
- Een nadeel is dat je in Pulumi snel onoverzichtelijke code krijgt

Samenvatting



Pulumi

- Gebruikt programmeertalen: Python, TypeScript, C#, GO, Java
- Gemiddeld tragere prestaties
- Meer testmogelijkheden: unit-, integratietesten, ...
- Moeilijker om aan te leren
- Kleiner ecosysteem, nog in volle groei



Terraform

- Werkt met HCL: Hashicorp Configuration Language
- Gemiddeld snellere prestaties
- Iets beperkter, fouten worden soms maar bij het uitvoeren van een terraform apply ontdekt
- Snel te leren
- Groot ecosysteem (industriestandaard), meer dan 6000 providers

Eindadvies

Eindadvies

Een volledige overstap is op dit moment niet aanbevolen

Terraform kan op dit moment behouden worden, aangezien dit momenteel goed werkt

Het team kan Pulumi wel gebruiken voor projecten waar de tool mogelijk een meerwaarde heeft.



Credits

Rol

Gilles De Meerleer

Onderzoeker

Alexander Veldeman

Promotor

Dieter Adant

Co-promotor

Nick Dehoux

Co-promotor

Bedankt voor het luisteren en het bijwonen van mijn presentatie!
Zijn er nog vragen?