Security in Infrastructure as Code

Florian Angermeir | 2023-01-05

Infrastructure as Code (IaC)

What is IaC

Technique to provision, manage and configure infrastructure, as code under version control

Applications

- Virtual Machines, Networks, Databases
- Configuration as Code (e.g. Installed packages, Environment variables)
- …

Infrastructure as Code (IaC)

Motivation

- Manual actions or individual scripts -> time consuming, error prone
- Results in snowflake systems
- Change management
- No fast reaction to changes -> Hinders fast development flow

Advantages

- Versioning
- Repeatable, Consistent -> Reproducible
- Testable
- Automatable -> fast setup, easy scaling
- Enhanced collaboration/Expertise shift -> DevOps Enabler

Tool List

Cloud Provider dependent

- AWS CloudFormation
- Azure Resource Manager
- Google Cloud Deployment Manager

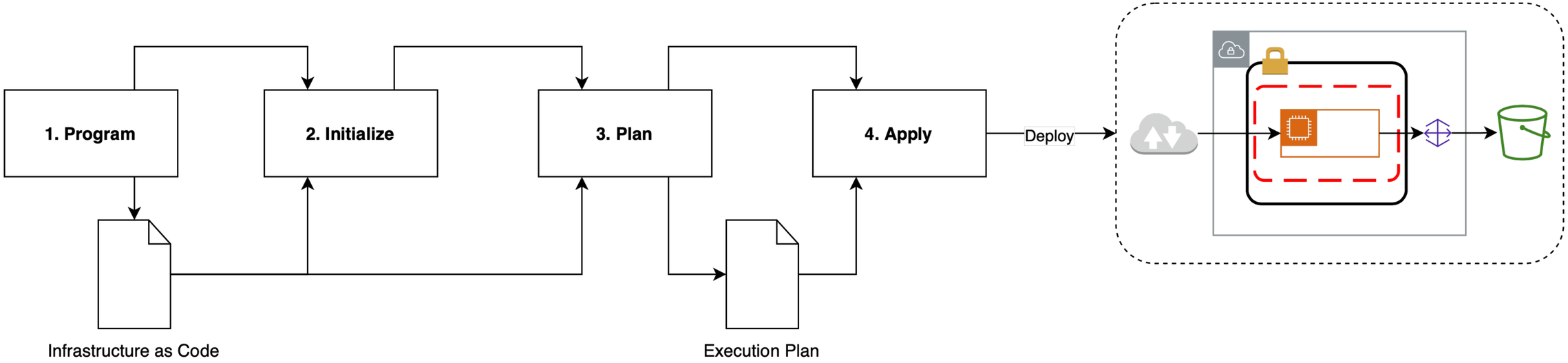
Cloud Provider “independent”

- Terraform
- Pulumi

Configuration as Code tools

- Puppet
- Chef
- Ansible

Terraform Workflow



1. **Program** Infrastructure is codified
2. **Initialize** Download requirements specified in code
3. **Plan** Evaluate configuration changes and output plan how to achieve changes
4. **Apply** Deploy configuration changes

Terraform Workflow

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = "t3.micro"
}

resource "aws_security_group" "allow_all_traffic" {
  name        = "allow_all_traffic"
  description = "Allows worldwide traffic"
  vpc_id      = var.vpc_id

  egress {
    description = "Traffic to worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }

  ingress {
    description = "Traffic from worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }
}

resource "aws_s3_bucket" "s3_bucket" {
  bucket = "s3_bucket"
}

resource "aws_vpc_endpoint" "gateway_endpoint_for_s3" {
  vpc_id       = var.vpc_id
  service_name = "com.amazonaws.eu-central-1.s3"
}
```

```
# aws_security_group.allow_all_traffic will be created
  + resource "aws_security_group" "allow_all_traffic" {
      + arn                    = (known after apply)
      + description            = "Allows worldwide traffic"
      + egress                 = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = "Traffic to worldwide"
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 65535
            },
        ]
      + id                     = (known after apply)
      + ingress                = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = "Traffic from worldwide"
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 65535
            },
        ]
      + name                   = "allow_all_traffic"
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + revoke_rules_on_delete = false
      + tags_all               = (known after apply)
      + vpc_id                 = "vpc_id"
    }
```

What can go wrong?

Various reports about what can go wrong (e.g. NSA, Verizon, Cybersecurity Insiders)

World readable S3 Bucket

World writeable S3 Bucket

EC2 Instance open to internet with overprivileged instance role

What can go wrong?

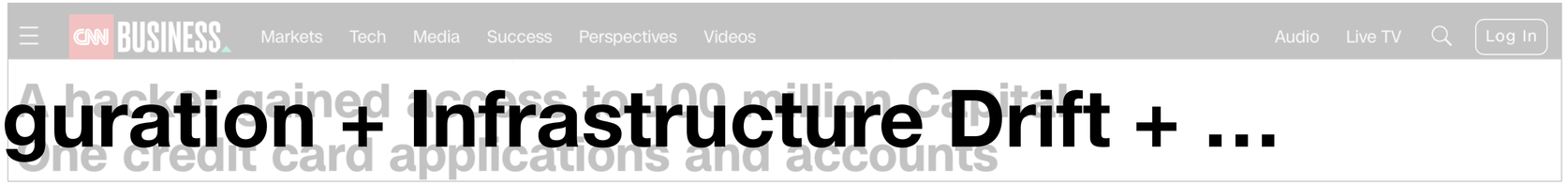
Various reports about what can go wrong (e.g. NSA, Verizon, Cybersecurity Insiders)

World readable S3 Bucket

Insecure Configuration

World writeable S3 Bucket

EC2 Instance open to internet with overly permissive access



Insecure Configuration + Infrastructure Drift + …

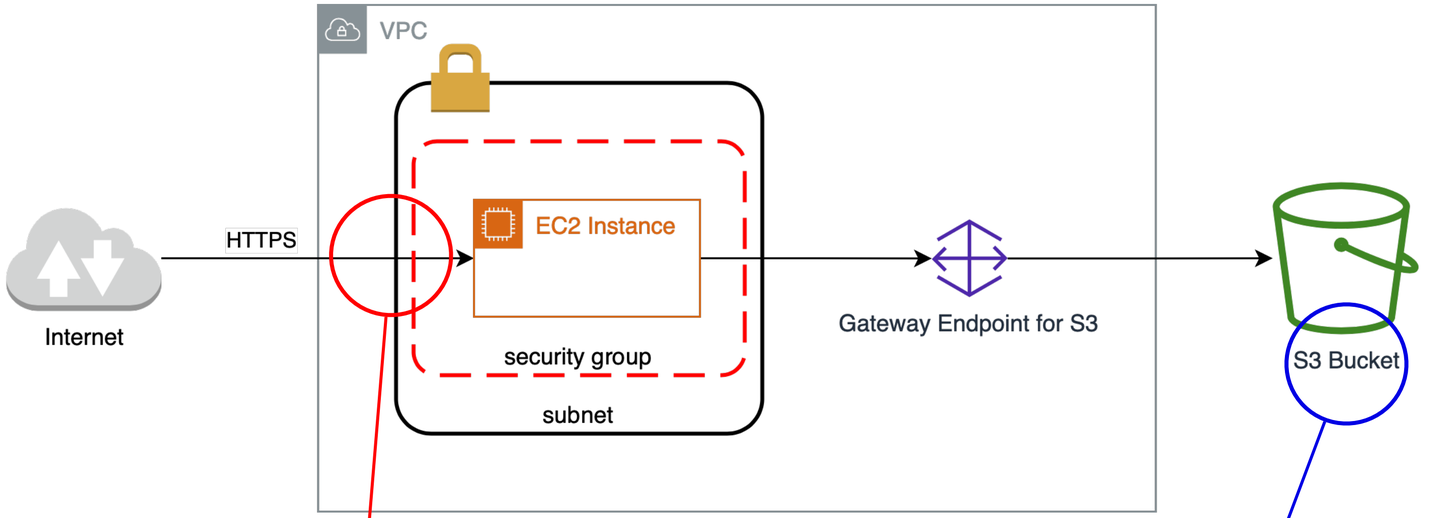
What can go wrong?

1. Insecure configuration of infrastructure
2. Drift in infrastructure
3. …

Insecure Configuration of Infrastructure

Insecure Configuration

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = "t3.micro"
}

resource "aws_security_group" "allow_all_traffic" {
  name        = "allow_all_traffic"
  description = "Allows worldwide traffic"
  vpc_id      = var.vpc_id

  egress {
    description = "Traffic to worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }

  ingress {
    description = "Traffic from worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }
}

resource "aws_s3_bucket" "s3_bucket" {
  bucket = "s3_bucket"
}

resource "aws_vpc_endpoint" "gateway_endpoint_for_s3" {
  vpc_id       = var.vpc_id
  service_name = "com.amazonaws.eu-central-1.s3"
}
```



Firewall too wide open (e.g. allows traffic to ports other than 443)

S3 bucket unprotect, world-readable

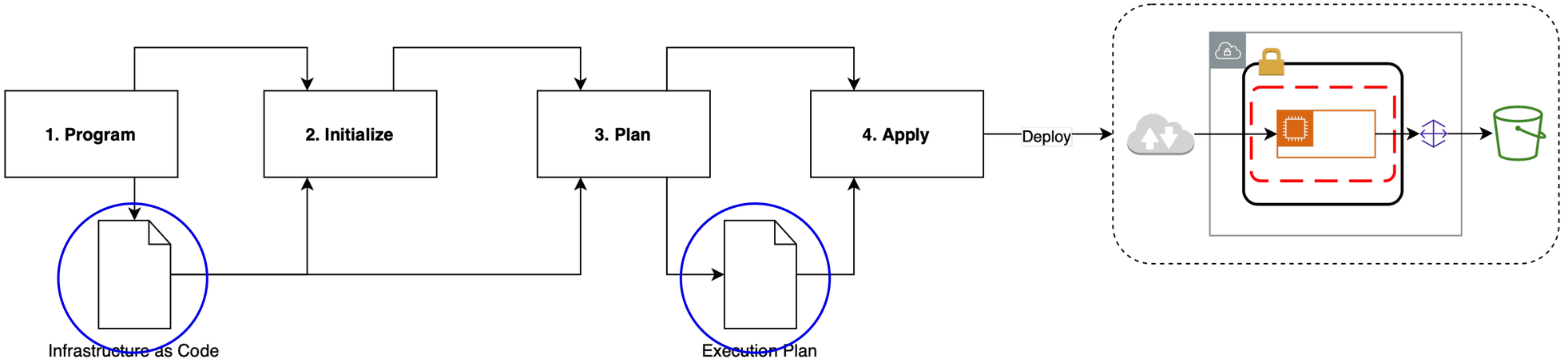
Remediations

Organisation-Wise

- Don’t reinvent the wheel: https://registry.terraform.io
- Build repository of secure modules everybody can use to quickly spin up infrastructure

Project-Wise

- Infrastructure defined as code -> Perform static code analysis on it



Configuration vs. Execution Plan

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = "t3.micro"
}

resource "aws_security_group" "allow_all_traffic" {
  name        = "allow_all_traffic"
  description = "Allows worldwide traffic"
  vpc_id      = var.vpc_id

  egress {
    description = "Traffic to worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }

  ingress {
    description = "Traffic from worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }
}

resource "aws_s3_bucket" "s3_bucket" {
  bucket = "s3_bucket"
}

resource "aws_vpc_endpoint" "gateway_endpoint_for_s3" {
  vpc_id       = var.vpc_id
  service_name = "com.amazonaws.eu-central-1.s3"
}
```

```
# aws_security_group.allow_all_traffic will be created
  + resource "aws_security_group" "allow_all_traffic" {
      + arn                    = (known after apply)
      + description            = "Allows worldwide traffic"
      + egress                 = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = "Traffic to worldwide"
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 65535
            },
        ]
      + id                     = (known after apply)
      + ingress                = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = "Traffic from worldwide"
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 65535
            },
        ]
      + name                   = "allow_all_traffic"
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + revoke_rules_on_delete = false
      + tags_all               = (known after apply)
      + vpc_id                 = "vpc_id"
    }
```

IaC Security Scanners

	Checkov	Kics	Tfsec	Terrascan
Popularity	4.8k Stars	1.2k Stars	5.2k Stars	3.3k Stars
Maintainer	Bridgecrew	Checkmarx	Aquasecurity	Tenable
Scan Basis	Config, Plan	Config, Plan	Config	Config, Plan
Providers	AWS, Azure, GCP	AWS, Azure, GCP	AWS, Azure, GCP	AWS, Azure, GCP
Shipped Policies	>1000	>1500	~250	~250
Extensibility	Python	Rego	Rego	Rego
License	Apache 2.0	Apache 2.0	MIT	Apache 2.0
URL	https://github.com/bridgecrewio/checkov	https://github.com/Checkmarx/kics	https://github.com/aquasecurity/tfsec	https://github.com/tenable/terrascan
Version	v2.1.270	v1.6.2	v1.28.0	v1.15.2

IaC Security Scanners - Showcase

	Checkov	Kics	Tfsec	Terrascan
Critical	0	0	2	0
High	2	6	8	3
Medium	2	3	2	2
Low	2	1	1	0
Info	0	4	0	0
Total	6	14	13	5

- Kics yields most findings, but also output non-security findings (e.g. tagging)
- Even though Tfsec has least information it finds a lot of issues

- But: Where do the differences come from?

IaC Security Scanners - Showcase

	Checkov	Kics	Tfsec	Terrascan		Total	Deduplicated
EC2	2	3	2	3		**10**	**8**
S3	4	4	9	1		**18**	**10**
Network	0	3	2	1		**6**	**5**
Other	0	4	0	0		**4**	**4**

- Seems like these tools have different strengths (e.g. tfsec with S3)
- No tool covers all findings in one area

- Might be that tool policies are of different granularities
- More extensive comparison: https://github.com/iacsecurity/tool-compare

Different granularity?

```
# Finding 1
{
   "query_name":"Unknown Port Exposed To Internet",
   "query_url":"https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security_group",
   "severity":"HIGH",
   "cloud_provider":"AWS",
   "category":"Networking and Firewall",
   "description":"AWS Security Group should not have an unknown port exposed to the entire Internet",
}
# Finding 2
{
   "rule_name":"unrestrictedIngressAccess",
   "description":"Ensure no security groups allow ingress from 0.0.0.0/0 to ALL ports and protocols",
   "severity":"HIGH",
   "category":"Infrastructure Security",
   "resource_name":"allow_all_traffic",
   "resource_type":"aws_security_group"
}
```

Different granularity?

Vulnerable Configuration

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = "t3.micro"
}

resource "aws_security_group" "allow_all_traffic" {
  name        = "allow_all_traffic"
  description = "Allows worldwide traffic"
  vpc_id      = var.vpc_id

  egress {
    description = "Traffic to worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }

  ingress {
    description = "Traffic from worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "-1"
    from_port   = 0
    to_port     = 65535
  }
}

resource "aws_s3_bucket" "s3_bucket" {
  bucket = "s3_bucket"
}

resource "aws_vpc_endpoint" "gateway_endpoint_for_s3" {
  vpc_id       = var.vpc_id
  service_name = "com.amazonaws.eu-central-1.s3"
}
```

Fixed Configuration

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = "t3.micro"
}

resource "aws_security_group" "allow_all_traffic" {
  name        = "allow_all_traffic"
  description = "Allows worldwide traffic"
  vpc_id      = var.vpc_id

  egress {
    description = "Traffic to worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "tcp"
    from_port   = 443
    to_port     = 443
  }

  ingress {
    description = "Traffic from worldwide"
    cidr_blocks = [var.cidr]
    protocol    = "tcp"
    from_port   = 443
    to_port     = 443
  }
}

resource "aws_s3_bucket" "s3_bucket" {
  bucket = "s3_bucket"
}

resource "aws_vpc_endpoint" "gateway_endpoint_for_s3" {
  vpc_id       = var.vpc_id
  service_name = "com.amazonaws.eu-central-1.s3"
}
```

Execution Plan

```
# aws_security_group.allow_all_traffic will be updated in-place
  ~ resource "aws_security_group" "allow_all_traffic" {
      ~ egress                = [
          - {
              - cidr_blocks      = [
                  - "0.0.0.0/0",
                ]
              - description      = "Traffic to worldwide"
              - from_port        = 0
              - ipv6_cidr_blocks = []
              - prefix_list_ids  = []
              - protocol         = "-1"
              - security_groups  = []
              - self             = false
              - to_port          = 65535
            },
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = "Traffic to worldwide"
              + from_port        = 443
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 443
            },
        ]
        id                    = "sg-08ca023c579282023"
        ...
    }
```

WrapUp

- Tools use varying sources (e.g. Source Code vs. Execution Plan)
- Tools scan based on self-defined policies (which are not standardized)

- Seems like different tools have different strengths (e.g. tfsec with S3)
- No tool alone is sufficient to cover all areas

- Choice of employed tool strongly depends on project characteristics and security demands
- There is no one-size-fits-all solution!
- You will have to define policies for tools yourself

Infrastructure Drift

Infrastructure Drift

What is Infrastructure Drift?

Difference between what is configured in code and what is actually deployed

- Resources not existing in code, but in deployment environment
- Resources existing in code, but not in deployment environment
- Resources with different properties in code and deployment environment

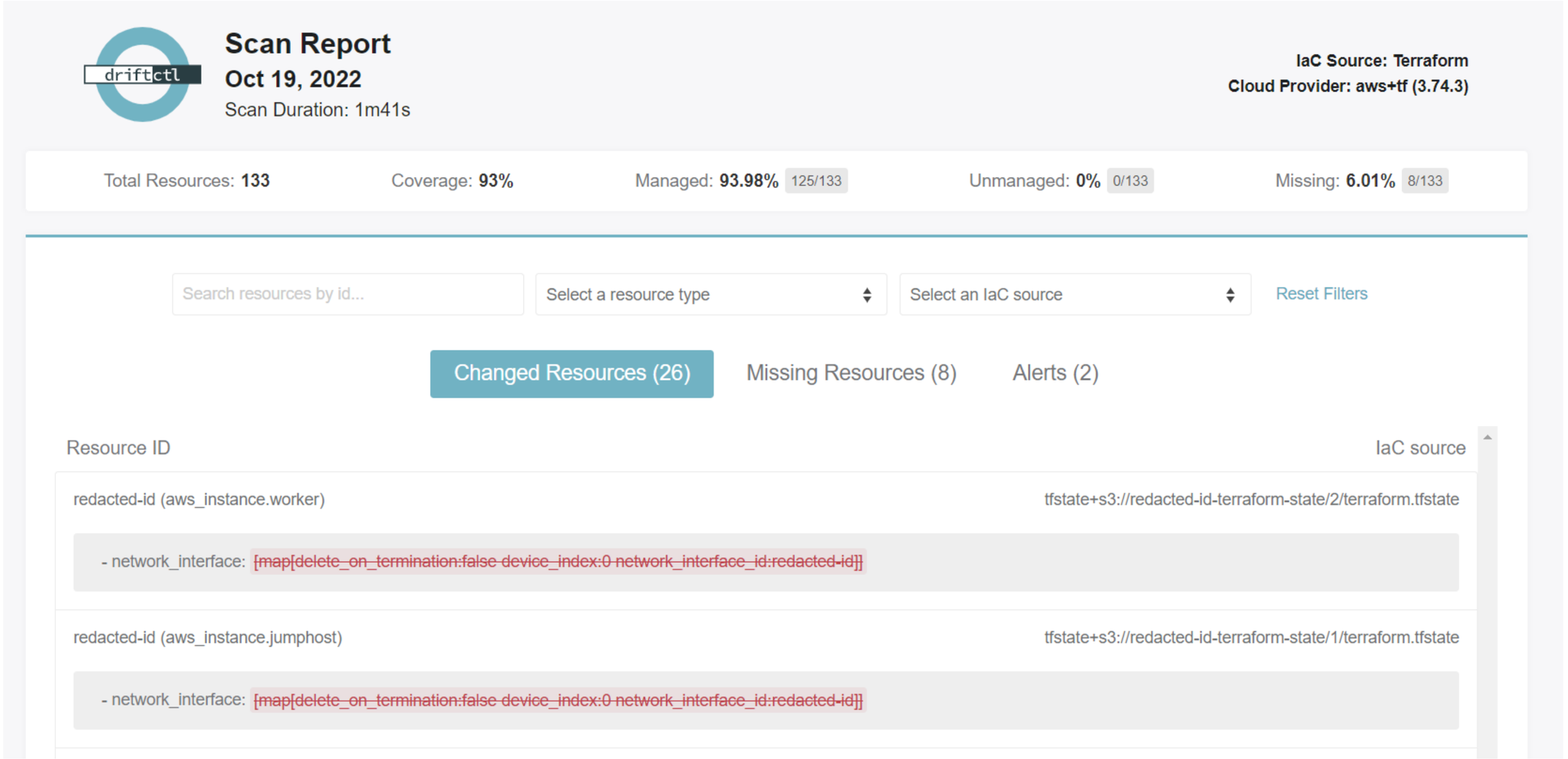
Causes

- Manual intervention: Stakeholder manually creates/updates/deletes resources
- Technical failure: Tool execution fails, computer crashes mid-deployment
- …

Detection Tools

	Bridgecrew	Accurics	Driftctl
Maturity	Mature	Mature	Beta
Maintainer	Commercial	Commercial	Free
Providers	AWS, Azure, GCP	AWS, Azure, GCP	AWS, Azure, GCP
Integrations	GitHub Actions, GitLab CI, VSCode, …	Jenkins, Circle CI, …	GitHub Actions, Gitlab CI, Jenkins, Circle CI
Platforms	External Platform	External Platform	Local CLI Tools

Driftctl


driftctl
Scan Report
Oct 19, 2022
Scan Duration: 1m41s
IaC Source: Terraform
Cloud Provider: aws+tf (3.74.3)
Total Resources: 133
Coverage: 93%
Managed: 93.98% 125/133
Unmanaged: 0% 0/133
Missing: 6.01% 8/133
Search resources by id...
Select a resource type
Select an IaC source
Reset Filters
Changed Resources (26)
Missing Resources (8)
Alerts (2)
Resource ID
IaC source
redacted-id (aws_instance.worker)
tfstate+s3://redacted-id-terraform-state/2/terraform.tfstate
- network_interface: [map[delete_on_termination:false device_index:0 network_interface_id:redacted-id]]
redacted-id (aws_instance.jumphost)
tfstate+s3://redacted-id-terraform-state/1/terraform.tfstate
- network_interface: [map[delete_on_termination:false device_index:0 network_interface_id:redacted-id]]

WrapUp

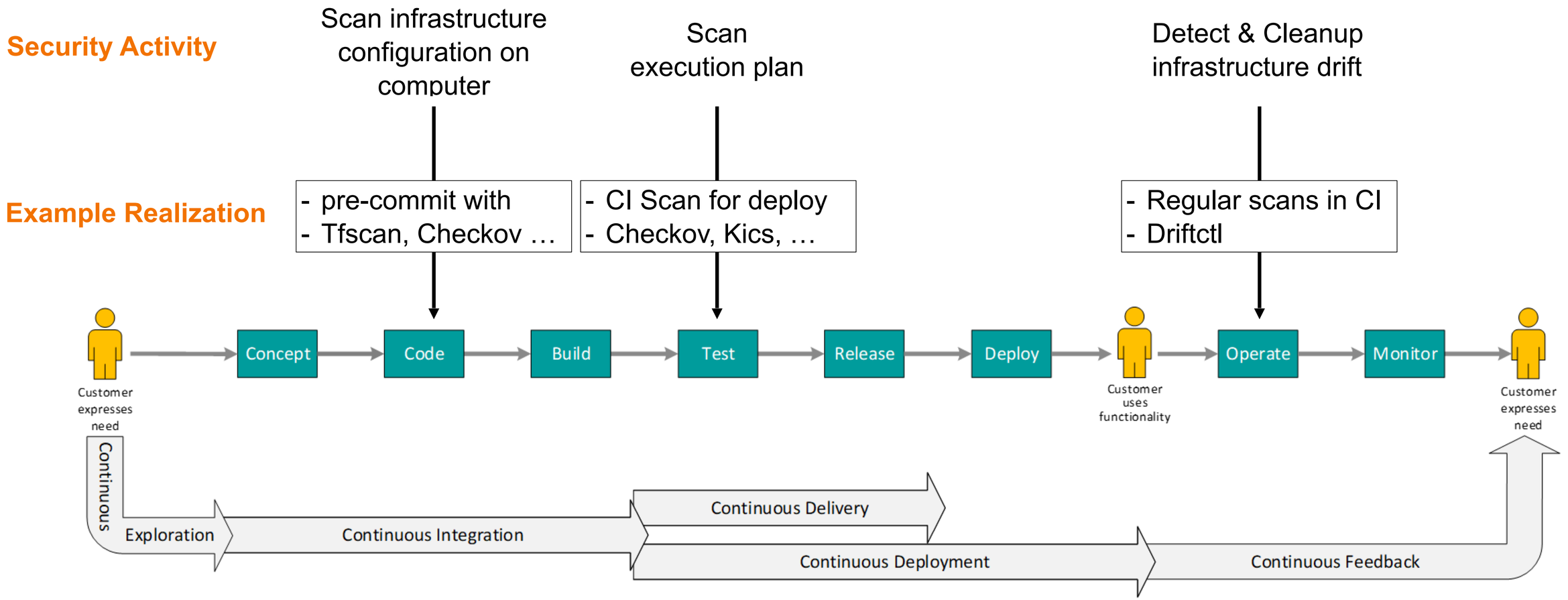
- Infrastructure drift can occur due to various reasons
- Drift is inevitable!

- Poses real security threats
- Difficult to detect

- Only few tools available to detect infrastructure drift
- Hard to solve for non-operations affine people

Summary

IaC Security in DevOps



Further Resources

…on how insecure configurations get into production

- https://github.com/bridgecrew/terragoat
- https://github.com/tenable/KaiMonkey

…comparing IaC scanners

- https://github.com/iacsecurity/tool-compare

…exemplary IaC security implementation

- https://github.com/angrymeir/Security_in_IaC_Example

Contact

Florian Angermeir

florian.angermeir@tum.de

https://angermeir.me