

MANAGING AWS ACCOUNTS ON A MASSIVE SCALE

Ivan Petrushevski, AWS Cloud Architect



QUICK INTRO



Ivan Petrushevski

AWS Cloud Architect

Passion for:
Automation, IaC, DevOps, GitOps

linkedin.com/in/petrushevski

EUROPEAN LEADER IN PUBLIC CLOUD

- We offer a fully managed public cloud transformation, from infrastructure to application development
- We are the preferred partner for our strategic hyperscale partners AWS, Microsoft and Google
- Nordcloud is a Managed Services Provider for all 3 hyperscale partners - being in the elite group of less than ten companies worldwide with this same status
- We are recognized in Gartner's Magic Quadrant for Public Cloud Infrastructure Managed Services Providers 2017 & 2018
- Among TOP 5 providers globally in DevOps and Mode 2 delivery on public cloud (Gartner)

Key numbers

- Customers: +100 *
- Deployed projects: +500
- Founded 2011
- Growth 2015-2018 50 %
Revenue CAGR

* 40% of OMX40

Highly skilled organization

350+ cloud experts with
250+ certifications and
200+ business and technical accreditations.

Local presence

Norway, Finland, Sweden, Poland, Netherlands, UK, Austria,
Switzerland, Denmark, Germany

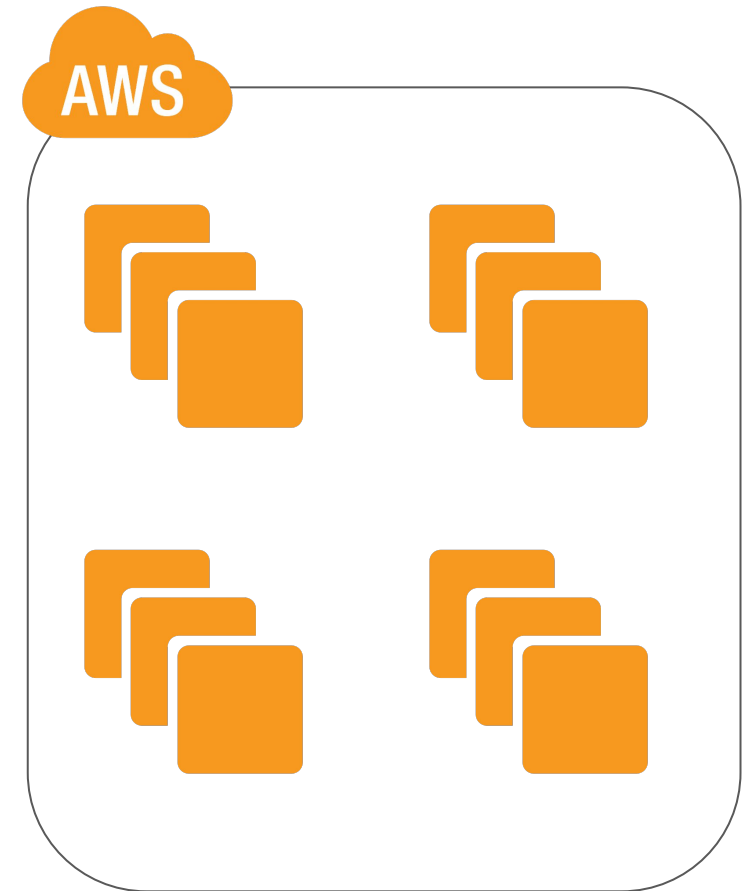


AGENDA

1. The need for multiple AWS accounts
2. Provisioning accounts
 - a. Traditional approach
 - b. Cloud native approach
3. Deploying infrastructure and application stacks

AWS ACCOUNT

- AWS account is a container of resources
- Email is “root” user
- Billing attached
 - Directly
 - Or consolidated in another account
- Standalone or part of AWS Organization
- Does NOT cost you money to create it



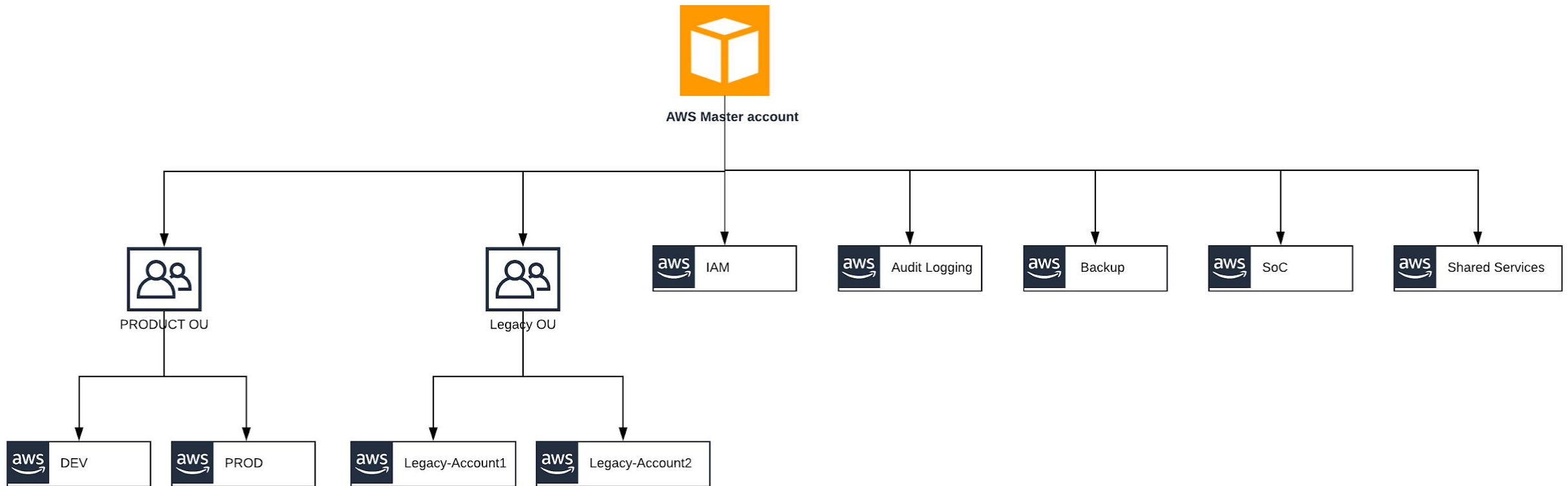
WHY MORE AWS ACCOUNTS?

- **Environment** isolation
 - development
 - production
- **Resource** isolation
 - Shared services
 - Backups
 - Sandboxing
- **Business or product** isolation
 - Organizational units
- **Teams** isolation
 - Product teams
 - Tech stack teams (Network, Development)



“Everything is about **reducing** blast radius.”

GENERIC AWS ORGANIZATION



PROVISIONING NEW AWS ACCOUNTS - VENDING MACHINE

TRADITIONAL APPROACH OF ACCOUNT CREATION

Speed versus Control


Developers

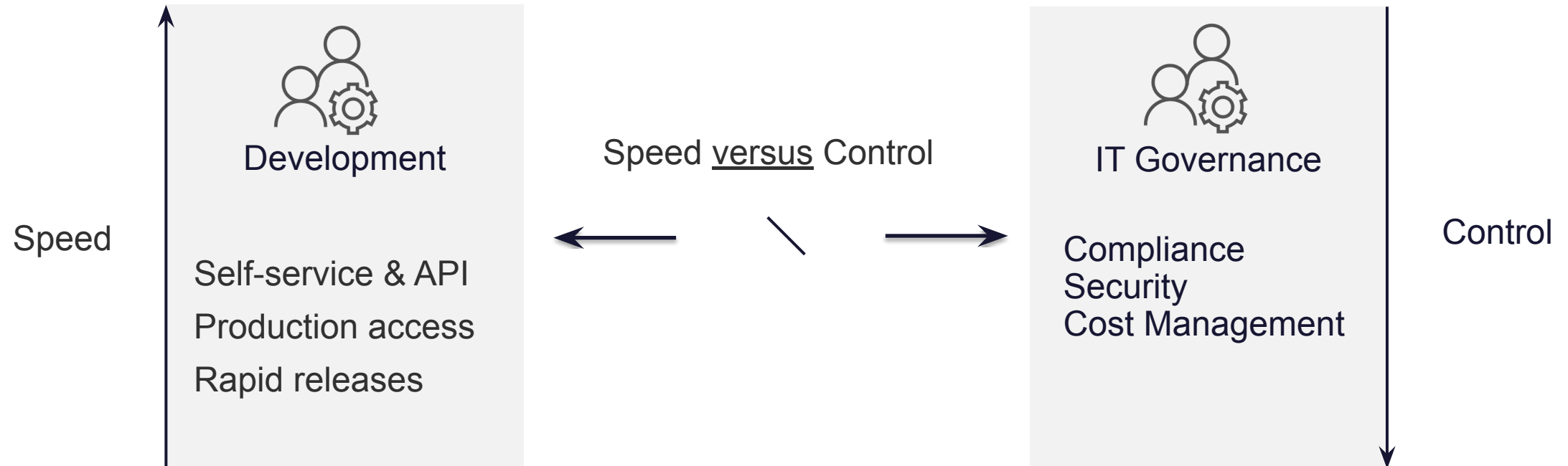

Operations



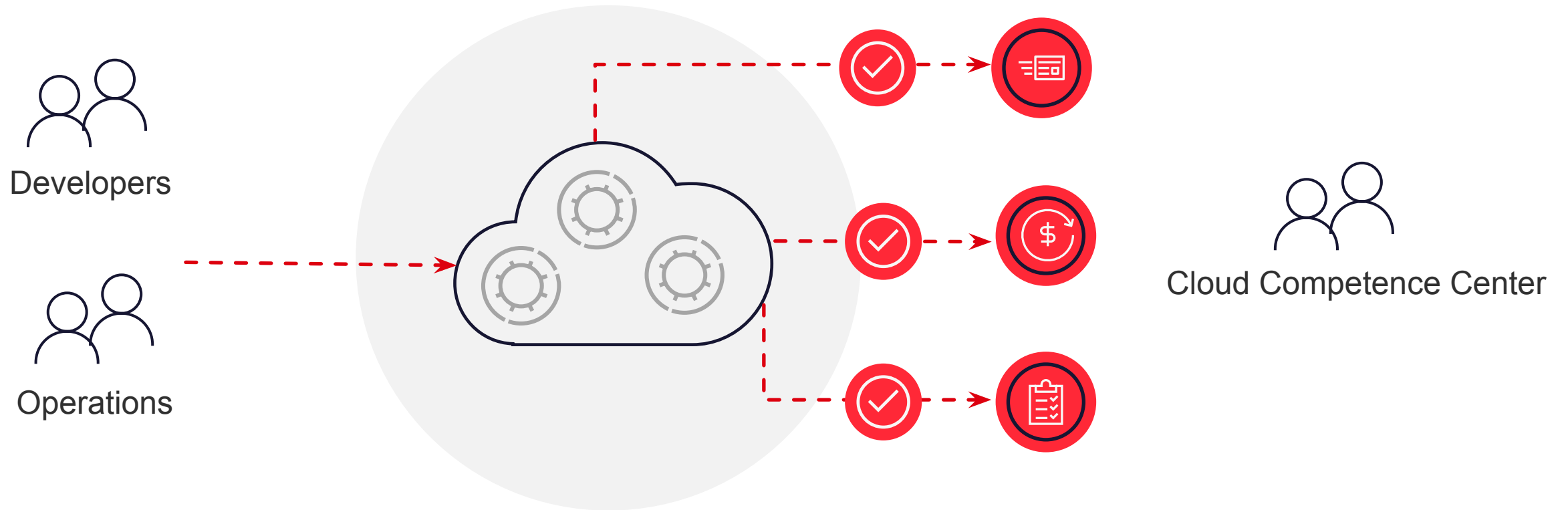
Cloud Competence Center



TRADITIONAL APPROACH



CLOUD NATIVE APPROACH

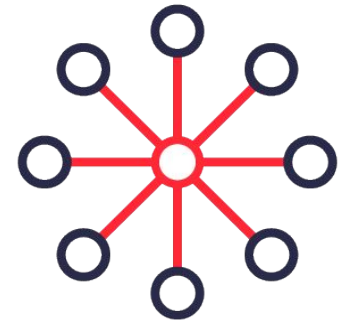


Speed with Control - automated foundation for provisioning

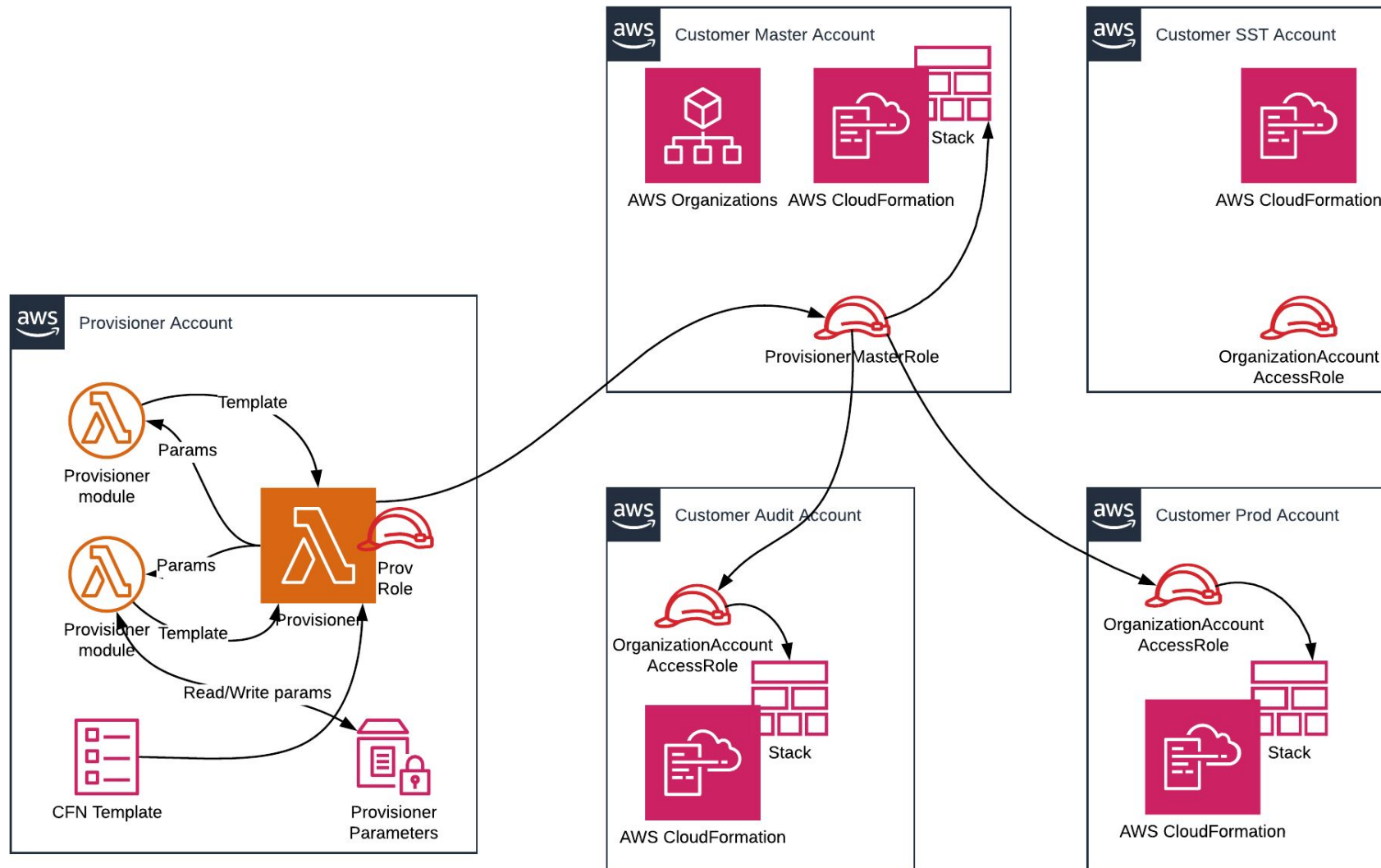
PROVISIONER

PROVISIONER MAIN FEATURES

- Manage **AWS Organization**
- Create static (CloudFormation) or dynamic (Lambda) **templates and blueprints** that can be used to **provision** hundreds of AWS accounts
- Setup **core infrastructure** that will be automatically deployed to every Account in the Organizational Unit (OUs)
- Design and run sophisticated, **multi-account deployments**



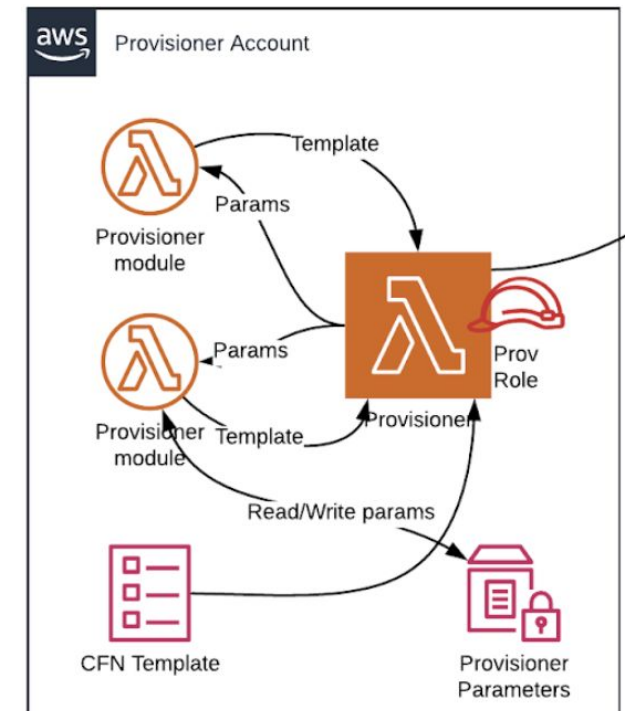
HOW PROVISIONER WORKS



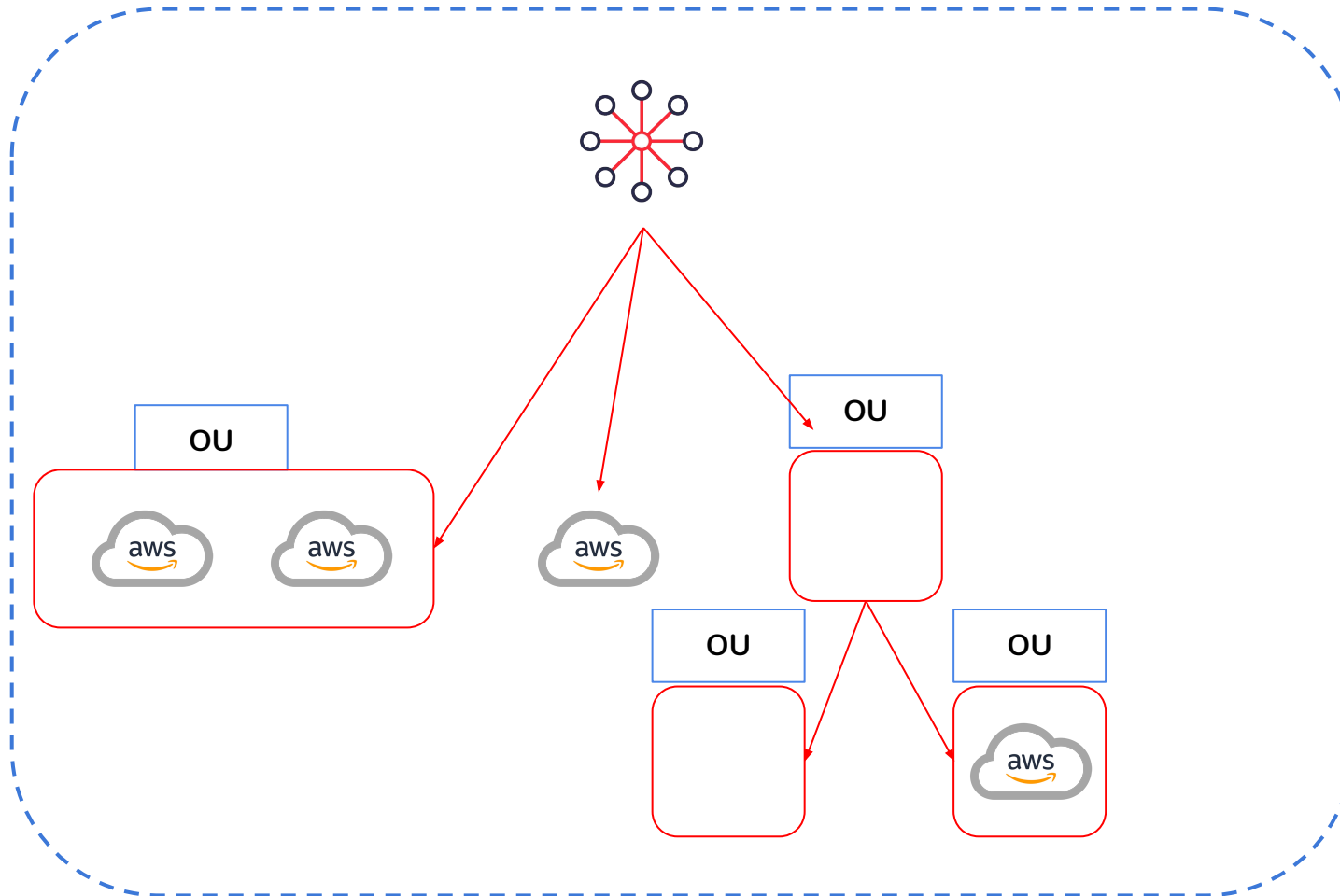
HOW PROVISIONER USES OBJECTS

If you want to define a state - run **THIS** Cloudformation yaml on every account in **XYZ** OU, provisioner needs to know

- 1) That **THIS** cloudformation template needs to be launched → create **template** object in provisioner with i.e. S3 URL to CFN stack
- 2) Combine 1+ templates into a single → **blueprint** object that represents a set of templates to be provisioned during a single operation
- 3) Define via → **deployment** object which **blueprint** object should be deployed on accounts in **XYZ** OU



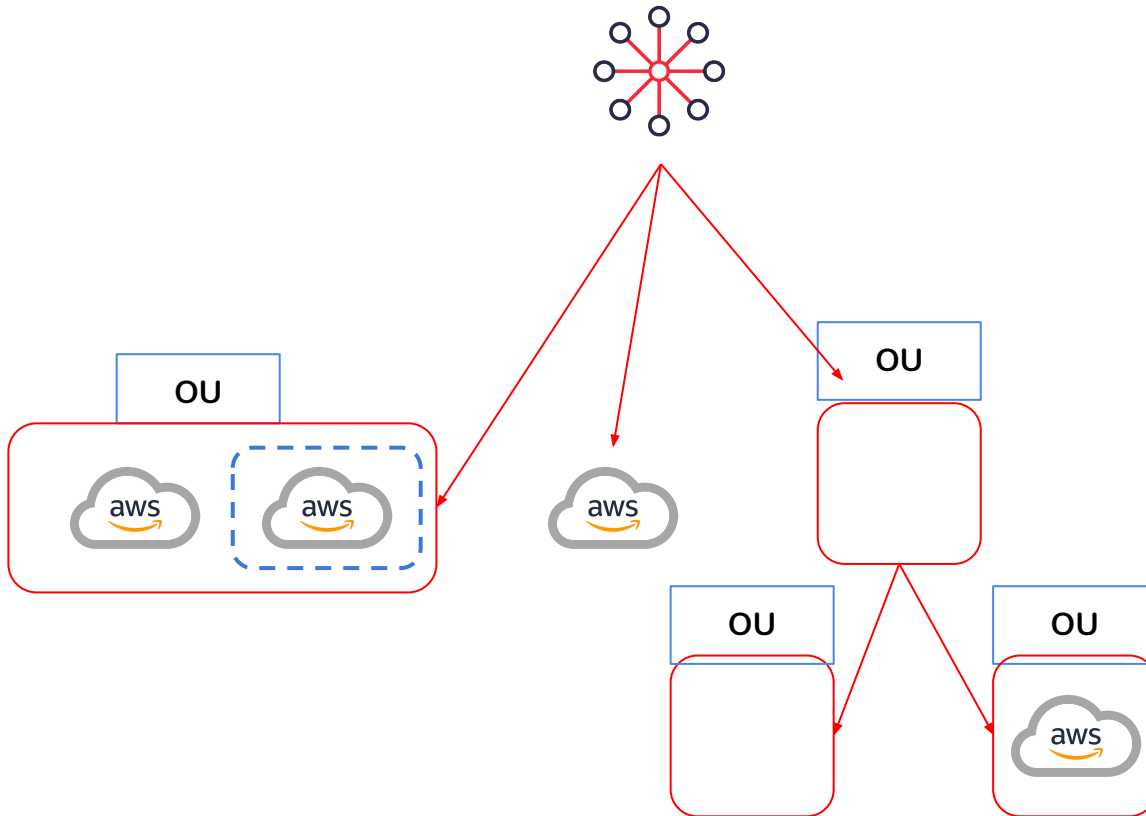
ORGANIZATION (provctl apply -f org.yaml)



```
apiVersion: v1
kind: Organization
metadata:
  name: TheBestOrganization
spec:
  orgAccountId: 123456789012
  access:
    - roleArn: RoleForProvisioner
      externalId: poufnyajdi
```

ACCOUNT (provctl apply -f account.yaml)

```
apiVersion: v1
kind: Account
metadata:
  name: my-iam-account
spec:
  organizationalUnit:
    name: core
    accountAlias: my-iam-account
    email: real.email+iam@demo.cloud
```



PROVISIONER ACCOUNT OBJECT

```
apiVersion: v1
kind: Account
metadata:
  name: my-iam-account
spec:
  organizationalUnit:
    name: core
    accountAlias: my-iam-account
    email: real-email+iam@demo.cloud
```

```
apiVersion: v1
kind: Account
```

...

“GitOps: versioned CI/CD on top of declarative infrastructure. Stop scripting and start shipping.”

- Kelsey Hightower

DEPLOY INFRASTRUCTURE AND APPLICATION STACKS

WHAT PROVISIONER DOES

Provisioner is an **orchestration** tool.

It **takes**

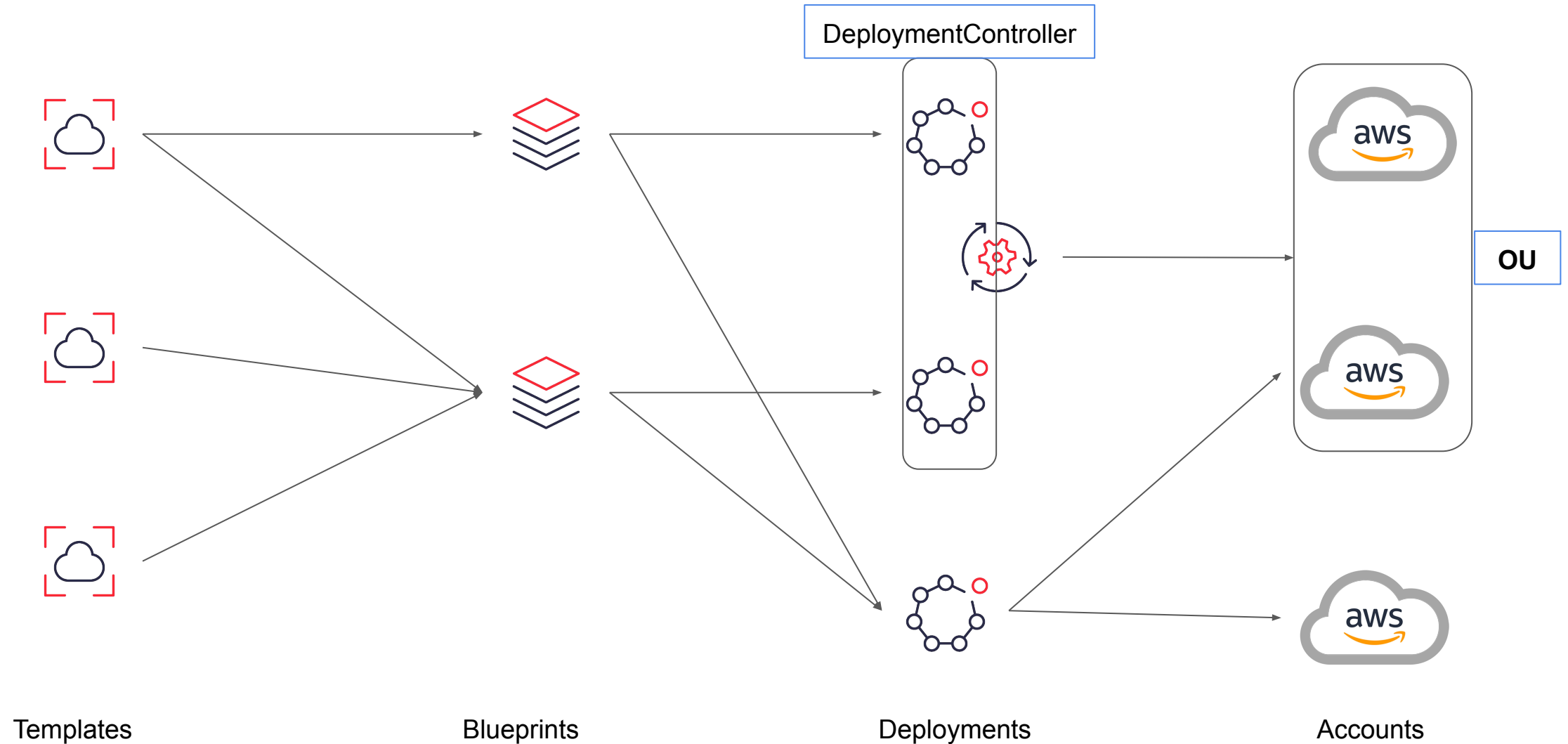
- Static templates
 - Cloudformation template (yaml/json)
- Dynamic templates

Provisioners sends defined parameters to Lambda function and receives in response Cloudformation template

and **deploys** Cloudformation stacks on particular standalone AWS Accounts or AWS Accounts within AWS Organizations OU



3000 FT VIEW OF DEPLOYMENT



PROVISIONER TEMPLATE OBJECT

```
apiVersion: v1
kind: Template
metadata:
  name: iam-account-ReadOnlyRole
  version: 3
  annotations:
    environment: prod
spec:
  Source: ./cf-templates/iam-account/ReadOnlyRole.yaml
```


PROVISIONER BLUEPRINT OBJECT

```
apiVersion: v1
kind: Blueprint
metadata:
  name: iam-account
  version: 6
spec:
  templates:
    - name: iam-account-ReadOnlyRole
      version: 3
    - name: iam-account-CloudAdminRole
      version: 2
    - name: iam-account-SandboxRole
      version: 1
```

PROVISIONER DEPLOYMENT OBJECT

```
apiVersion: v1
kind: Deployment
metadata:
  name: iam-account
spec:
- name: manage-iam-roles
  account:
    name: nordcloud-iam-account
    blueprint:
      name: iam-account
      version: 6
    defaultSettings:
      regions:
      - eu-north-1
        parameters:
      - name: IdentityProvider
        value: arn:aws:iam::123456789012:saml-provider/gsuite
```

PROVCTL GET ACCOUNTS

```
Ivans-MBP:ipetrushevski:~$ provctl get accounts
```

NAME	ALIAS	ACCOUNT ID	ORGANIZATIONAL UNIT	STATUS
demo-husq-simple-dep-test	demo-husq-simple-dep-test	48	demo-husq-prov-simple	succeeded
rnd_prov_test_account	rnd-prov-test-account5	32	demo-ou-3	succeeded
aws-op-demo-monday	aws-op-demo-monday	54	demo-ou-3	succeeded
rnd-prov-test-async	rnd-prov-test-async	90	demo-ou-child	succeeded
demo-husq-audit		35	root	succeeded
aws-demo-audit		16	root	succeeded
nordcloud-test-organization	aws-nordcloud-organization-test	61	root	succeeded
test-new-account-kp	testnewaccountkp0089	62	root	succeeded
kamil-test-22	kamil-test-1	69	account-policy-demo-ou	succeeded
kamil-test-account-234	kamil-trest-cos-tam	51	root	succeeded
test-account-12345		60	a	failed
test-9995	nordlcoud-test-9995	36	demo-ou-child-333	succeeded
test-new-account-kp-2		49	root	succeeded
aws-nc-demo-tue	aws-nc-demo-tue	31	root	succeeded
aws-nc-mc-demo	aws-nc-mc-demo	20	demo-ou-child	succeeded
test-new-account-kp-2222	testnewaccountkp0089112	10	root	succeeded
demo-mc-test-0003TestKor	demo-mc-test-0003testkorr	24	root	succeeded
test-new-account-kp-22222	testnewaccountkp00891122	54	root	succeeded
rnd-prov-test-async-final-api007	rnd-prov-test-async-final-api0010	93	root	succeeded
rnd-prov-test-async-final-api		15	root	succeeded

PROVCTL GET TEMPLATES

```
Ivans-MBP:ipetrushevski:~$ provctl get templates
```

NAME	VERSION	TYPE
demo-husq-simple-lambda	1.0	S3_URL
demo-husq-simple-sqs	1.0	S3_URL
foundation_audit	0.0.a12a97b4	LAMBDA_ARN
foundation_soc	0.0.a12a97b4	LAMBDA_ARN
prep_audit	0.0.a12a97b4	LAMBDA_ARN
prep_soc	0.0.a12a97b4	LAMBDA_ARN
source	0.0.a12a97b4	LAMBDA_ARN
sqsTemplate 33	1	S3_URL
sqsTemplate1	1	LOCAL
sqsTemplate	1	LOCAL
test-aws-template-x	0.0.d6f5ec4e	S3_URL
test-aws-template-x	1.1	S3_URL
test-aws-template-xx	0.0.d6f5ec4e	S3_URL
test-aws-template	1.0	LAMBDA_ARN
test-blueprint-dynamic-640847a0	0.0.1	LAMBDA_ARN
test-local-template	1.0	S3_URL
test-local-template	2.0	S3_URL
test-s3-template-2	1.0	S3_URL

PROVCTL GET TEMPLATE BODY

```
Ivans-MBP:ipetrushevski:~$ provctl get template iam-account-ReadOnlyRole:3 --body
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Creates IAM ReadOnlyRole for Federated Principal in IAM Account",
  "Parameters": {
    "IdentityProvider": {
      "Type": "String",
      "Description": "Provide the arn of the Federation Identity Provider"
    }
  },
  "Resources": {
    "Role": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": "ReadOnlyRole",
        "Description": "Read only role used to jump in other AWS accounts with ReadOnly access",
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
```

PROVCTL GET BLUEPRINTS

```
Ivans-MBP:ipetrushevski:~$ provctl get blueprints
```

NAME	VERSION	TEMPLATES NUM
blueprint-sqs-lambda	1.0	2
demo-husq-foundation-audit	1.0	1
demo-husq-foundation-soc	1.0	1
demo-husq-prep-audit	1.0	1
demo-husq-prep-soc	1.0	1
demo-husq-service	1.0	1
demo-husq-simple	1.0	2
test-blueprint-dynamic	1.0.0	2
test-manual	10	1

MORE COMMANDS...

Change/choose Provisioner Organization:

provctl change-org

Create/update resources defined in template.yaml:

provctl apply -f template.yaml

Describe template named s3bucket:

provctl describe template s3bucket

Delete Blueprint WebApp1:

provctl delete blueprint WebApp1

List all deployments with version and type:

provctl get deployments

DEMO - OU AND ACCOUNT CREATION

[OU-And-Account-Creation.mov](#)

The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'provisioner-demo' with a subdirectory 'account-structure' containing files like 'account.yaml', 'organizational-unit.yaml', and 'deployments'. The main editor area shows the content of 'organizational-unit.yaml' with the following YAML code:

```
1  apiVersion: v1
2  kind: OrganizationalUnit
3  metadata:
4    name: nic-demo-ou
5    annotations:
6      environment: demo
7      purpose: test
8  spec:
9    parent:
10     name: root
11
```

The terminal at the bottom shows the execution of the following commands:

```
-rw-r--r--@ 1 ipetrushevski  staff  156 Feb  3 17:11 organizational-unit.yaml
Ivans-MBP:ipetrushevski:~/Documents/Testings/provisioner-demo/account-structure$ provctl apply -f organizational-unit.yaml
Provisioner will apply Objects defined in following files to the nordcloud-test-organization organization:
- organizational-unit.yaml

Do you want to perform these actions?
Only 'yes' will be accepted to approve.

Enter a value: yes

OrganizationalUnit "nic-demo-ou" applied.
Ivans-MBP:ipetrushevski:~/Documents/Testings/provisioner-demo/account-structure$ provctl get organizationalUnits | grep nic-demo-ou
nic-demo-ou      ou-0p8o-p3apbjc  root      0      0
Ivans-MBP:ipetrushevski:~/Documents/Testings/provisioner-demo/account-structure$
```

The terminal output shows the successful creation of the organizational unit 'nic-demo-ou' and its details.

DEMO - S3 BUCKET DEPLOYMENT

[deployment.mov](#)

The screenshot shows a Nordcloud IDE interface. On the left, a project explorer shows a file tree for 'visioner-demo [master]' with files like .git, account-structure, account.yaml, organizational-unit.yaml, deployments, blueprint.yaml, deployment.yaml, and s3-cf-template.yaml. The main editor displays the 'deployment.yaml' file with the following content:

```
1  apiVersion: v1
2  kind: Deployment
3  metadata:
4    name: nic-demo-deployment
5    annotations:
6      environment: demo
7      purpose: test
8  spec:
9    - name: deploy-s3-demo
10      account:
11        name: "nic-demo-account"
12      blueprint:
```

Below the editor, a terminal window shows the output of the deployment:

```
Name:          nic-demo-deployment
Status:        PENDING
deploy-s3-demo
  AccountName:  nic-demo-account
  Blueprint:    nic-demo-blueprint v.: 1
  Default settings:
    Regions:    eu-north-1,
  Parameters:
    - BucketName: nic-conf-demo-bucket
  State:
    Status:     pending
    Templates states:
Metadata:
  environment: demo
  purpose: test
```

The bottom status bar shows the file path 'deployments/deployment.yaml', a red error icon, and various settings like 'LF', 'UTF-8', 'YAML', 'git+', 'master', 'No remote', 'GitHub', and 'Git (0)'.

THANK YOU!

Let's stay in touch

ivan.petrushevski@nordcloud.com

[linkedin.com/in/petrushevski](https://www.linkedin.com/in/petrushevski)

