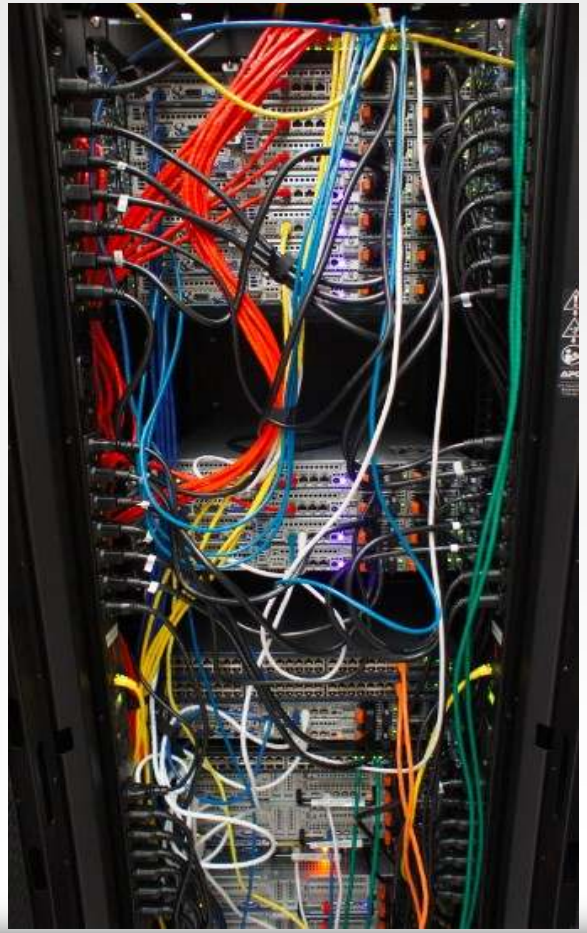
Infrastructure as code with Amazon Web Services

Julien Simon
Principal Technical Evangelist, AWS
julsimon@amazon.fr
@julsimon

Agenda

- Infrastructure as code
- AWS CloudFormation
- Hashicorp Terraform
- Troposphere



```
"Conditions" : {
  "HaveNoOtherRoles"  : { "Fn::Equals" : [{"Ref" : "OtherRoles"}, ""]},
  "HaveEbs"           : { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "EbsVolumeSize"}, "0"]}] },
  "HaveEbsSnapshotId" : { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "EbsSnapshotId"}, ""]}] },
  "HaveAdditionalTagKey": { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "AdditionalTagKey"}, ""]}] },
  "HaveAdditionalTagValue": { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "AdditionalTagValue"}, ""]}] },
  "HaveSSL": { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "SSLPort"}, "0"]}] },
  "IsHTTP" : { "Fn::Equals" : [{"Ref" : "ElbProtocol"}, "HTTP"]},
  "HaveSpotPrice" : { "Fn::Not" : [{ "Fn::Equals" : [{"Ref" : "SpotPrice"}, ""]}]}
},
"Resources": {
  "AutoScalingGroup": {
    "Type": "AWS::AutoScaling::AutoScalingGroup",
    "UpdatePolicy" : {
      "AutoScalingRollingUpdate" : {
      "MaxBatchSize" : "1",
      "MinInstancesInService" : "0",
      "PauseTime" : "PT15M",
      "WaitOnResourceSignals": "true"
      }
    },
    "Properties": {
      "LaunchConfigurationName": { "Ref": "LaunchConfig" },
      "LoadBalancerNames": [ { "Ref": "ElasticLoadBalancer" } ],
      "MinSize": { "Ref": "MinPoolSize" },
      "MaxSize": { "Ref": "MaxPoolSize" },
      "AvailabilityZones": { "Fn::FindInMap": ["AZConfig", "AvailabilityZones", "all"] },
      "VPCZoneIdentifier": { "Ref": "EC2SubnetsIds" },
      "Tags" : [
        { "Fn::If": [
            "HaveAdditionalTagKey",
            {
              "Key" : { "Ref": "AdditionalTagKey" },
              "Value": {
                "Fn::If": [
                  "HaveAdditionalTagValue",
                  {"Ref": "AdditionalTagValue"},
                  ""
                ]
              },
              "PropagateAtLaunch": "true"
            },
            {"Ref" : "AWS::NoValue"}
          ]
        },
        { "Key" : "Name", "Value" : { "Fn::Join" : [ "-", [ { "Ref" : "ServiceName"}, {"Ref" : "EnvironmentName"
        { "Key" : "cost", "Value" : { "Ref" : "Cost" }, "PropagateAtLaunch": "true" },
        { "Key" : "environment", "Value": { "Ref" : "EnvironmentName"}, "PropagateAtLaunch": "true" }
      ]
    }
  }
```

Why infrastructure as code rocks

Automated: save time & reduce human error

Predictable: build the same infra every time

Traceable: keep track of all changes

Testable: make sure best practices are built-in

You don’t get all of this with scripting

Typical use cases

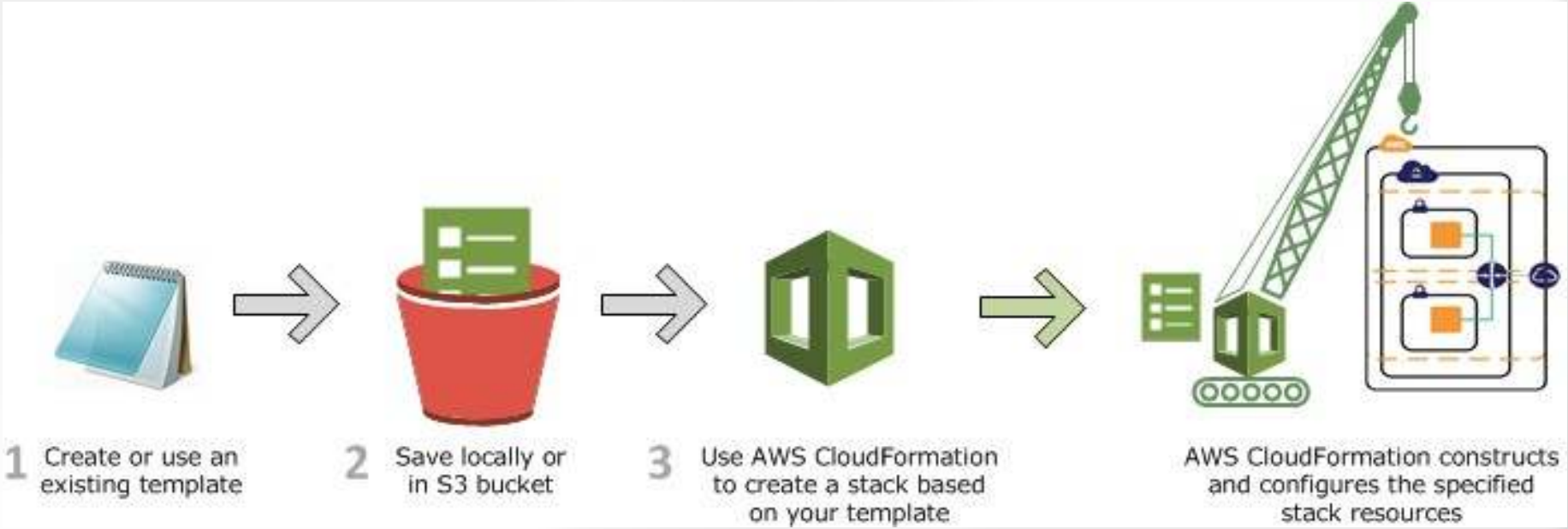
- Building as many environments as you need
  - Development, staging, pre-production, production
  - Same architecture, different sizing → template + parameters
- Deploying in a different region
- Performing green / blue deployments
- Preparing for Disaster Recovery

AWS CloudFormation

AWS CloudFormation

- Fundamental service used to automate creation, configuration and destruction of AWS resources (VPC, EC2, RDS, etc.)
- Infrastructure is described in a template
  - JSON or YAML file. Not a script!
  - Resources, Parameters, Outputs, etc.
- CloudFormation ingests the template and builds a stack of AWS resources
- Pricing: no charge

AWS CloudFormation



CloudFormation Template

```
{
  "AWSTemplateFormatVersion" : "version date",

  "Description" : "JSON string",

  "Metadata" : {
    template metadata
  },

  "Parameters" : {
    set of parameters
  },

  "Mappings" : {
    set of mappings
  },

  "Conditions" : {
    set of conditions
  },

  "Resources" : {
    set of resources
  },

  "Outputs" : {
    set of outputs
  }
}
```

The CloudFormation CLI in one slide

```
$ aws cloudformation validate-template
--template-body file://template.json

$ aws cloudformation create-stack
--template-body file://template.json --stack-name MyTemplate

$ aws cloudformation get-template --stack-name MyTemplate

$ aws cloudformation update-stack --stack-name MyTemplate
--template-body file://template.json

$ aws cloudformation delete-stack --stack-name MyTemplate
```

Change sets

- CloudFormation used to be ‘fire and forget’
  - Or sometimes ‘fire and remember all your life’ ;)

- Change sets have been introduced to preview effects of stack creations and stack updates

- Please use them!
  - `aws cloudformation create-change-set`
  - `aws cloudformation describe-change-set`
  - `aws cloudformation execute-change-set`
  - `aws cloudformation delete-change-set`

Example: creating a new stack with a change set

```
$ aws cloudformation create-change-set
--stack-name Template0 --template-body file://template0.json
--change-set-type CREATE --change-set-name createTemplate0


{
. . .
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "createTemplate0",
    "Changes": [
        {
            "ResourceChange": {
                "Action": "Add",
                "ResourceType": "AWS::EC2::Instance",
                "Scope": [],
                "Details": [],
                "LogicalResourceId": "Ec2Instance0"
            },
            "Type": "Resource"
 . . .
```

CloudFormation demo

Starting stuff, updating it, deleting it

https://github.com/juliensimon/aws/tree/master/CF/cfdemo


Terraform

Terraform

- Open source project initiated by Hashicorp

  https://github.com/hashicorp/terraform
- Infrastructure is described in a configuration, written in Hashicorp Configuration Language (HCL)
  - Why HCL? https://github.com/hashicorp/hcl#why
- Support for many cloud / SaaS providers, but not an abstraction layer for “multi-cloud” (whatever that is)
- What I like about Terraform
  - Very easy to preview modifications
  - Ability to take drift into account (could be risky, though)

The Terraform CLI in one slide

```
$ terraform plan

$ terraform apply

$ terraform show

$ terraform plan -destroy

$ terraform destroy
```

Terraform demo

Same thing, but different

https://github.com/juliensimon/aws/tree/master/CF/terraform/


Troposphere

Troposphere

- Open source project
  https://github.com/cloudtools/troposphere

- Write Python code that generates a CloudFormation template (JSON), then use CloudFormation to build the stack

- What I like about Troposphere: more natural to write and debug than JSON / YML / HCL

Troposphere demo

Because why not?

https://github.com/juliensimon/aws/tree/master/CF/troposphere/

Summing things up

	CloudFormation	Troposphere	Terraform
Project nature	AWS service	Open Source (1700+ stars)	Open Source (7000+ stars)
Service coverage	Best. Used internally by EB, ECS, SAM, etc.	Very good for most services.	Very good. Many other cloud / SaaS providers.
Language	JSON, YML	Python (to JSON)	HCL
Interface	CLI, GUI		CLI
Multi-user	Yes (IAM)		No
Drift detection	No		Yes (‘refresh’)
Existing resources	Yes (with CloudFormer)	Yes (with CloudFormer, cfn2py)	Yes (with terraforming)
Continuous deployment	Yes (with CodePipeline)		No (build your own)
Preview updates	Yes (change sets)		Yes (‘plan’)
Rollbacks	Yes		No (in Open Source version)
Multi-account & multi-region	Yes (with Lambda)		Yes
Other features	Quick Starts	Access to Python libraries	Terraform console

Now what?

- CloudFormation, Terraform and Troposphere allow you to run automated and predictable infrastructure builds in any AWS region.

- Wait, there's more work!
  - Automate AMI builds - Aminator (Netflix), Packer (Hashicorp)
  - Automate service provisioning - Puppet, Ansible, Chef, AWS OpsWorks
  - Automate application deployment - Code*

- DevOps is a long journey, just take one step at a time ☺

Additional resources

http://aws.amazon.com/cloudformation
https://aws.amazon.com/fr/blogs/mt/
https://aws.amazon.com/fr/new/#management-tools

https://www.terraform.io/
https://github.com/cloudtools/troposphere

https://github.com/stelligent/cfn_nag
https://github.com/dtan4/terraforming

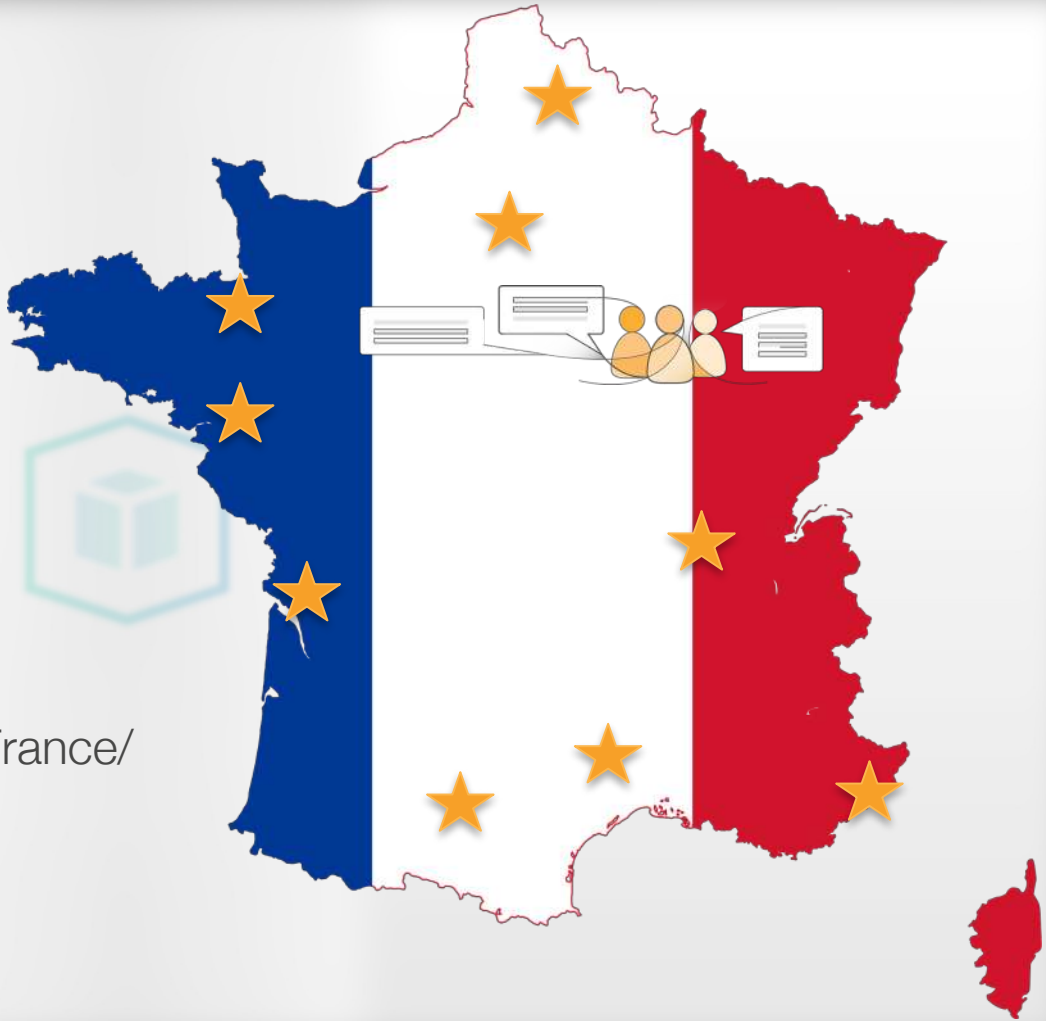
http://aws.amazon.com/free
http://console.aws.amazon.com

AWS User Groups

Lille
Paris
Rennes
Nantes
Bordeaux
Lyon
Montpellier
Toulouse
Côte d'Azur (new!)

facebook.com/groups/AWSFrance/

@aws_actus



https://aws.amazon.com/fr/events/webinaires/

Mars	Avril
Mardi 28 mars - 15h30	Mardi 25 avril - 15h30
EC2, pas juste des instances (en savoir plus)	Les bases de données relationnelles (en savoir plus)

Merci !

Julien Simon
Principal Technical Evangelist, AWS
julsimon@amazon.fr
@julsimon