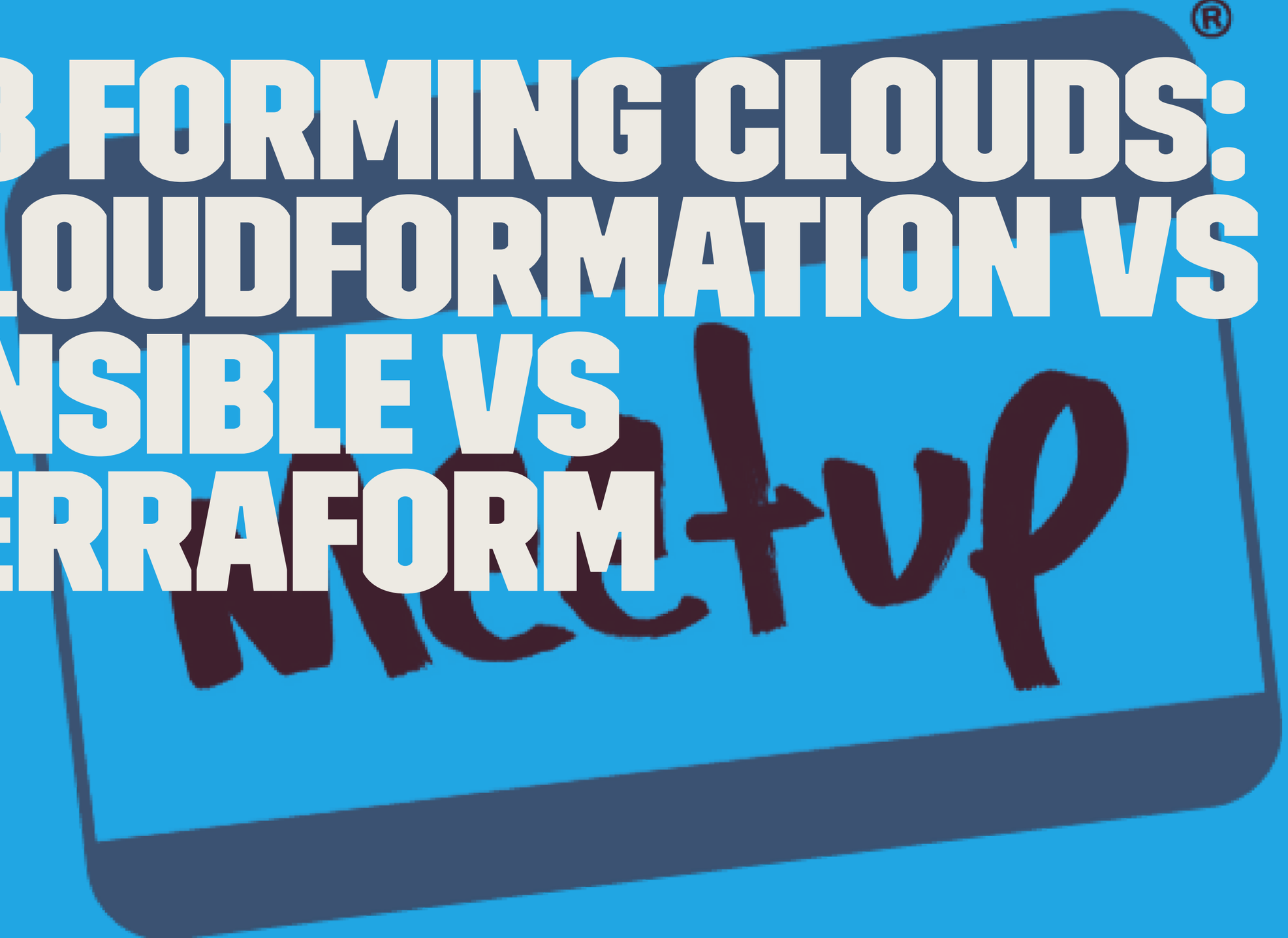
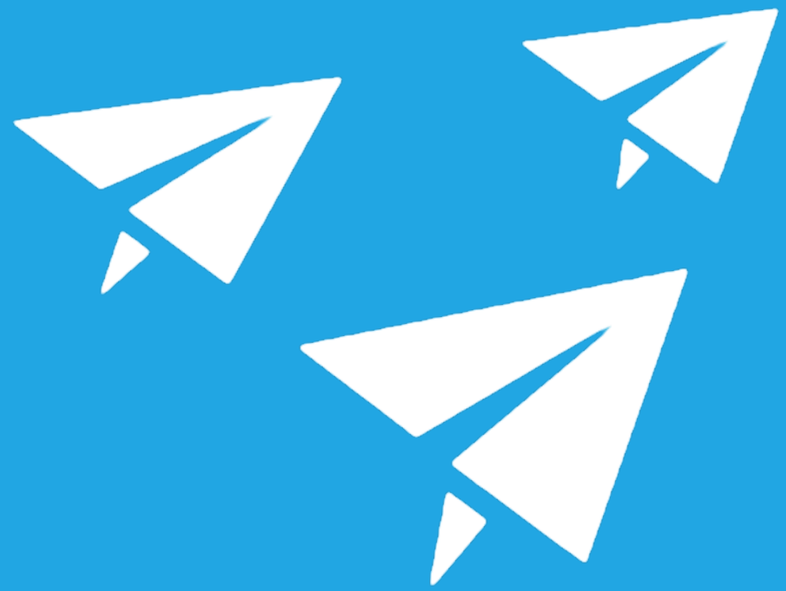
#8 FORMING CLOUDS: CLOUDFORMATION VS ANSIBLE VS TERRAFORM





skyscrapers

The Cloud & Ops Experts

GEERT THEYS

Sales guy @skyscrapers

Still likes to get his hands dirty.

github.com/gtheys
twitter.com/toadi
geerttheys.com

MATTIAS

Cloud Engineer @skyscrapers

Likes to experiment.

github.com/mattiasgees
twitter.com/mattiasgees
blog.mattiasgees.be

CLOUDFORMATION

TERRAFORM

ANSIBLE

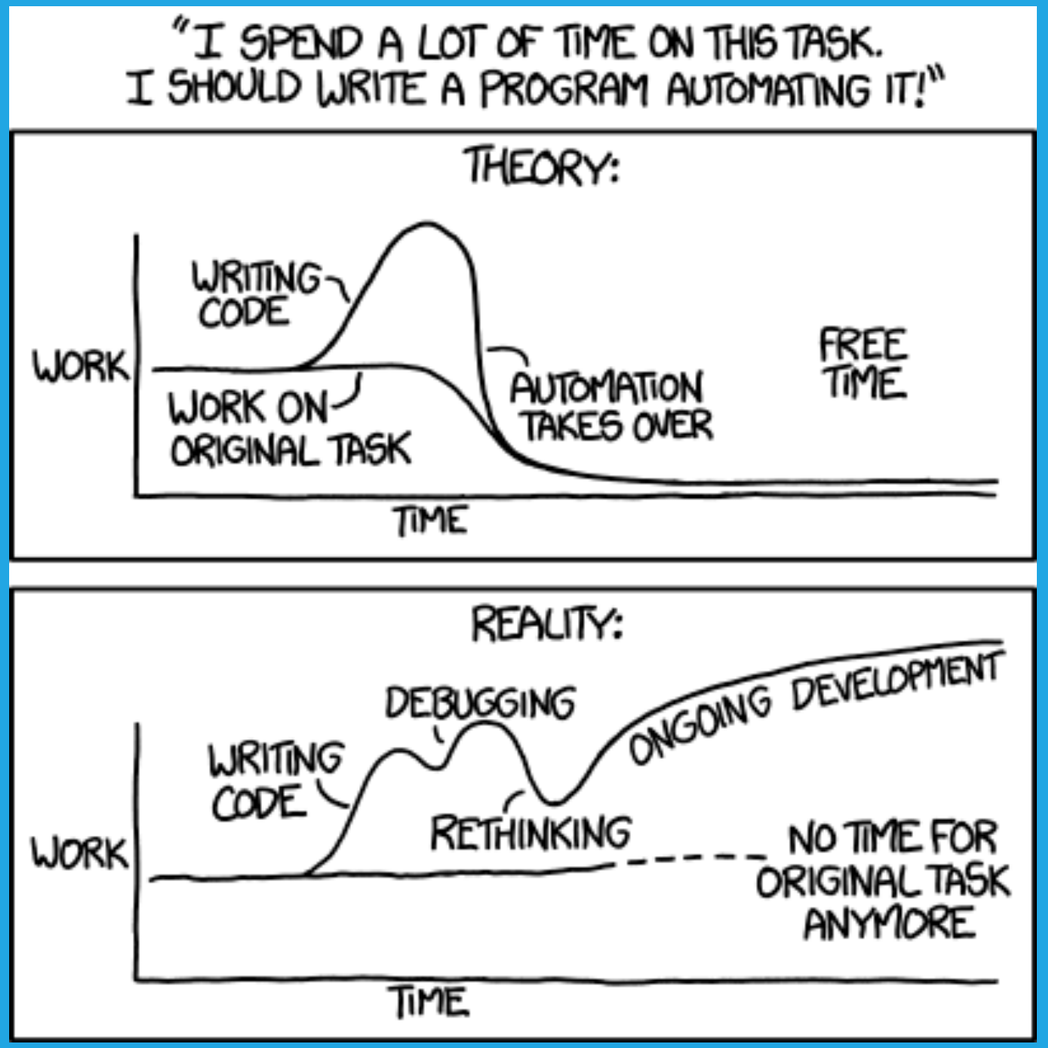
FOR AWS PROVISIONING

ALL DIFFERENT TOOLS

ALL HAVE THEIR ADVANTAGES

AND DISADVANTAGES

BEFORE WE START!



LETS COMPARE

SIMPLE SYNTAX?

EXAMPLE 1:

```
"web01" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "AvailabilityZone" : "eu-west-1a",
    "ImageId" : { "Ref" : "imageami" },
    "InstanceType" : { "Ref" : "webInstanceType" },
    "KeyName" : { "Ref": "opKeyName"},
    "SubnetId" : { "Ref" : "PublicSubnet1a" },
    "BlockDeviceMappings" : [ {
      "DeviceName" : "/dev/sda1",
      "Ebs" : {
        "VolumeType" : "standard",
        "DeleteOnTermination" : "false",
        "VolumeSize" : "8"
      }
    } ],
    "Tags" : [
      {"Key" : "Application", "Value" : { "Ref" : "AWS::StackId" } },
      {"Key" : "Environment", "Value" : { "Ref" : "environmentType" } },
      {"Key" : "Project", "Value" : { "Ref" : "projectName" } },
      {"Key" : "Name", "Value" : { "Fn::Join" : [ "-", [ "web01", { "Ref" : "projectName" }, { "Ref" : "environmentType" }] ] } }
    ],
    "SecurityGroupIds" : [ {"Ref" : "sgweb"} ],
    "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
      "#!/bin/bash\n",
      "/usr/bin/logger -t autobootstrap \"Run apt-get update\"\n",
      "sudo apt-get update\n",
      "/usr/bin/logger -t autobootstrap \"Install nginx\"\n",
      "sudo apt-get install nginx -y\n",
      "/usr/bin/logger -t autobootstrap \"Start nginx\"\n",
      "sudo service nginx start\n"
    ]]}}
  }
},
```

EXAMPLE 2:

```
resource "aws_instance" "web" {
  count = "${var.web_nodes}"
  ami = "${var.ami}"
  instance_type = "${var.instance_type}"
  subnet_id = "${element(aws_subnet.public_subnets.*.id, count.index)}"
  key_name  = "${var.key_name}"
  security_groups  = ["${aws_security_group.sg_web.id}"]
  user_data = "${template_file.metadata_web.rendered}"

  root_block_device {
    volume_type = "standard"
    volume_size = "8"
    delete_on_termination = "false"
  }

  tags {
    Name = "${var.project}-${var.environment}-web0${count.index + 1}"
    Environment = "${var.environment}"
    Project = "${var.project}"
  }
}

resource "template_file" "metadata_web" {
    filename = "templates/metadata.tpl"
}
```

EXAMPLE 3:

```
- name: WebServer | Create the WebServer Instance(s)
  local_action:
    module: ec2
    region: "{{ vpc_region }}"
    group: "{{ web_security_groups[0].sg_name }}"
    keypair: "{{ key_name }}"
    instance_type: "{{ web_instance_type }}"
    image: "{{ imgae_id.ami }}"
    vpc_subnet_id: "{{ item }}"
    assign_public_ip: True
    wait: True
    wait_timeout: 600
    user_data: |
              #!/bin/sh
              sudo apt-get install nginx -y
    instance_tags:
      Name: "{{ vpc_name }}_WEB_Instance"
      Environment: "{{ ENV }}"
      Role: "{{ server_role }}"
      Application: "{{ application }}"
  with_items:
     - "{{ public_subnet_1 }}"
     - "{{ public_subnet_2 }}"
  register: web
```

WHICH ONE WAS THE EASIEST TO READ?

EASY TO INSTALL?

MACOSX HAS THE FANTASTIC BREW:

» brew install awscli

» brew install terraform

» brew install ansible

Warning: None are up to date!

Up to date versions and on linux use pip install

SAFE TO USE?

Terraform and Ansible have a plan or --dry-run mode

PERFORMANT?

- Terraform:
  - Use dependency graph and parallelizes as much as possible
  - Partial refresh before changes
  - Destroy ordering

HOW DO THEY KEEP STATE?

DO I FEEL SAFE

- Terraform:
  - Partial State get's stored on error (eg. sg gets created not the rules, next run will fix this)
  - Create before destroy
- Cloudformation:
  - State is stored on AWS
  - Start to pray when you run it

ISSUES?

- Terraform:
  - Not yet good working on existing interfaces
  - No full coverage of AWS
- Cloudformation:
  - JSON
  - No plan mode

GOOD DEPLOYING MORE ADVANCED INFRASTRUCTURE?

SUPPORT?

DO YOU LOOK COOL WHEN USING IT?

DO I WANT TO USE IT?

DO WE RECOMMEND 1?