



Gordon

A very convenient AWS lambda
automation framework

David Melamed



PyWeb-IL - Dec 5, 2016

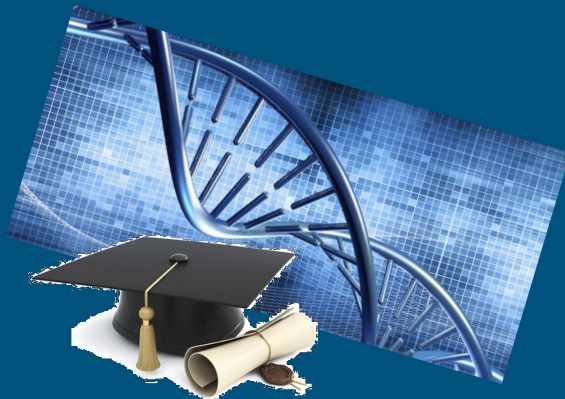
Agenda

- What is serverless? Why do we need this?
- AWS lambda basics
- Gordon, one way to automate lambda
- Demo

Who am I?



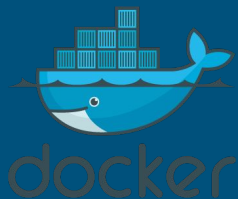
Moved to Israel 8 years ago



PhD in Bioinformatics



Technical Leader



Where do I work?



Leader in the Cloud Access Security Broker (CASB) market

Founded: 2011

Corporate Headquarters: Waltham, Mass. (U.S.A.)

R&D Headquarters: Tel Aviv

Employees: 160 (30 in TLV)

Trusted by major brands:

Acquired by Cisco



360K
APPS



10 M
USERS



1.3 B / month
ACTIVITIES



Traditional architectures

- **Monolithic architecture**
Easy to develop, test, deploy BUT hard to maintain & scale
- **Service-oriented architecture**
Easier to scale BUT harder to test & deploy
- **Microservices**
Easier to deploy & scale BUT harder to develop & test

The need for serverless



- Fully managed
 - No provisioning
 - Highly available
 - Cheaper (pay per invocation & sec.)
-
- Easy to develop
 - Easy to deploy
 - Easy to maintain
 - Hard to test



Available on each major platform

- AWS lambda
- Azure functions
- Google Cloud functions (alpha)

AWS Lambda in a nutshell

- Use containers behind-the-scenes
- Support for versioning, aliases and test events
- Supports Python, NodeJS, Java and C# 
- Supports environment variables 
- In-place edit for Python if no dependency
- Triggered by S3, SNS, SES, CloudWatch event...
- External call through API Gateway
- Pay-as-you-go per invocation and execution time (1,000,000 free requests)
- Logs automatically stored in CloudWatch Logs
- Monitoring through CloudWatch
- Limitations: 5 min max, 100 parallel executions (before throttle)



AWS Lambda hello-world example

```
def handler(event, context):  
    print "Hello world"
```

Gordon - python lambda automation framework



- Written in Python
- Uses CloudFormation behind-the-scenes
- Supports lambda function in python, javascript, java
- Supports multiple event sources: S3, events, dynamodb, API Gateway, SNS (*)
- Supports parameters & stages
- Supports for VPC
- Supports for secrets (**)
- Supports versioning and auto-update of the current alias
- Ability to run the lambda locally
- Custom build process

Gordon 5 commands

- startproject
- startapp
- run
- build
- apply

Full documentation: <http://gordon.readthedocs.io/en/latest/index.html>

Gordon project

- Structure includes apps and settings
- Start a project using scaffold by running:

```
gordon startproject my_project
```

- Project settings:
 - Code bucket
 - List of apps
 - Contexts
 - Lambda triggers: CloudWatch & S3 events
 - VPCs definition (SG/subnets)

```
my_project
├── settings.yml
├── my_app
│   ├── my_app_func
│   │   └── code.py
│   └── settings.yml
└── my_second_app
    ├── my_second_app_func
    │   └── code.py
    └── settings.yml
```

Gordon app

- Structure includes the app code and settings
- Add an app to the project by running:
`gordon startapp my_app`
- Settings for app:
 - List of lambda functions
 - For each lambda, code location & entrypoint, runtime, timeout, vpc IP, customer build, role

Gordon parameters & contexts

- Parameters in YAML file per “stage”
- Injected in a .context file available in lambda
- Support for secrets through ansible-vault (*)

(*) <https://github.com/dmelamedcl/gordon/pull/1>

Gordon run (local test)

- Ability to run locally the lambda function by running:

```
echo '{}' | gordon run my_app.my_lambda_func
```

- Simulate the event injected in lambda:

```
echo '{"param1": 12}' | gordon run my_app.my_lambda_func
```

- Simulate context injected to lambda:

```
echo '{}' | GORDON_CONTEXT=$(pwd)/path/to/context.json gordon run my_app.my_lambda_func
```

- Issue: not possible to use real context in local testing

Gordon build

- Builds the CloudFormation templates and artifacts (zip packages):

```
gordon build [--debug]
```

- Issue: how to deal with binaries / compiled libs?

- Solution: Custom build possible, i.e. using Docker ([lambdaci](#))

build:

- `cp -Rf * {target}`
- `docker run --rm -v {target}:/var/task -v /tmp/.pip-cache:/pip-cache lambdaci/lambda:build-python2.7 pip install -r requirements.txt -t {target} {pip_install_extra}`
- `cd {target} && find . -name "*.pyc" -delete`

Gordon apply

- Uploads the artifacts to s3
- Run the CloudFormation templates (which will create missing resources)
- Support of environments using:

```
gordon apply --stage=prod --region=us-east-1
```

- Support of secrets using:

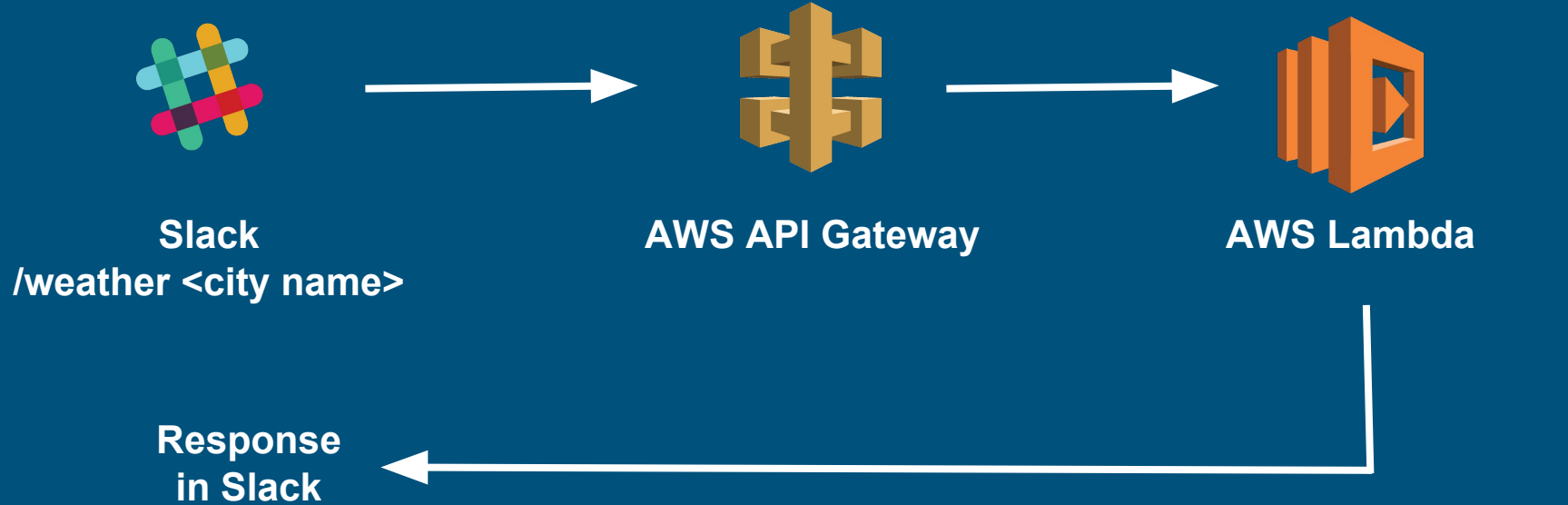
```
VAULT_PASSWORD_prod=my_pass gordon apply --stage=prod
```

DEMO TIME



Build a slack command returning the current weather in 5 min

Weather app architecture



A few alternatives to Gordon

- Serverless (Javascript) <https://serverless.com/>
- Apex (Go) <http://apex.run/>
- Chalice (Python) <https://github.com/awslabs/chalice>
- Sparta (Go) <http://gosparta.io/>
- SAM <https://github.com/awslabs/serverless-application-model/blob/master/versions/2016-10-31.md>



want to work for us?

#infra software engineer

#data scientist

#senior software engineer

#cybersecurity analyst

Interested? Send an email to dmelamed@cisco.com