

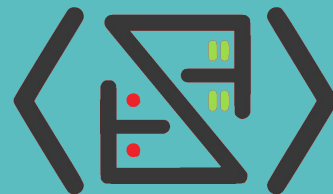
# ANSIBLE

## Going Serverless with Ansible

Ryan Scott Brown - @ryan\_sb  
Senior Software Engineer

# About

Blogs @ Serverless Code  
Community @ The Serverless Zone  
Works @ Ansible by Red Hat



**SERVERLESS CODE**

Tweets @ryan\_sb

- Ansible Philosophy
- Ansible for Cloud Resources
- Playing Well w/ Other Tools
- Mixing Infrastructures
- State of the (Serverless) Union

## Simple Wins

- Extend automation beyond the operations team
- Works on two levels of abstraction
  - Configuration management and provisioning
  - Orchestration and control-plane
- Bring existing tools and infrastructure

## Agentless

- Windows and Linux
- Third-party APIs
- Networking Hardware
- Security mitigation

- AWS
- Azure
- Google Cloud
- OpenStack

## Dynamic Inventory

- Resources are becoming more ephemeral
- Cloud provider API is now the source of truth

## Orchestrating Other Tools

```
- puppet:  
  environment: production  
  facts:  
    serve_with: ssl  
    application_version: v1.49.2
```



# Implementation Patterns

# Pattern 1

## Ansible and Serverless Framework

## Best of Both Worlds

- Helps to have devs with Serverless Framework experience
- Can pass control for shared resources up to Ansible

## Sample Play

- cloudformation:
  - stack\_name: prod-vpc
  - state: present
  - template: base\_vpc.yml
- cloudformation\_facts:
  - stack\_name: external\_stack

## Inside base\_vpc.yaml

Outputs:

VpcId:

Value: !Ref AppVPC

Export: {Name: !Sub “\${Stage}-VPCID”}

SubnetA:

Value: !Ref SubnetA

Export: {Name: !Sub “\${Stage}-VPCSubnetA”}

SubnetB:

Value: !Ref SubnetB

Export: {Name: !Sub “\${Stage}-VPCSubnetB”}

- cloudformation:
  - stack\_name: prod-vpc
  - state: present
  - template\_args:
    - Stage: prod
  - region: us-east-1
  - template: base\_vpc.yml
- serverless\_deploy:
  - stage: prod
  - functions:
    - createUser
    - deleteUser
  - region: us-east-1

## Migration Tips

- Attention to detail - who owns resources needed by both?
- Make sure data is close enough to new & old systems
- Don't couple serverless & legacy too tightly
- Remember Ansible can populate Serverless variable files

# Pattern 2

## Ansible All-In



## Fastest Start

- Ansible skills 100% transferrable
- Add to existing playbooks
- Great for “Lambda Light” workloads

## Build a Function

```
- lambda:  
  name: sendReportMail  
  zip_file: "{{ deployment_package }}"  
  runtime: python2.7  
  timeout: 20  
  handler: handler.handler  
  memory_size: 1024  
  role: "{{ iam_exec_role }}"  
register: lambda
```

## Cron Scheduling

- `cloudwatchevent_rule`:
  - `description`: "Send out incident summary daily"
  - `name`: `incident_summary_schedule`
  - `schedule_expression`: `'rate(1 day)'`
  - `targets`:
    - `id`: `SendSummary`
    - `arn`: `"{{ lambda.configuration.function_arn }}"`
    - `input`: `'{}'`

## Least Mature

- Limited event source availability
  - Kinesis/DynamoDB streams
  - CloudWatch Events
- No direct API Gateway (in v2.2)

# State of the Union

So, how's Jeff?

- Look for more integrations in 2.3
- Serverless Framework module
- Old & new apps can share tooling

# Thank You!

Questions to @ryan\_sb or find me at the social tonight