What *is* DevOps, really?

- DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.
What Kind of apps do we Write?

- node.js
- .NET Core
- Elasticsearch
- MongoDB
So, how Does all of this fit together?

- Introducing the Dockerfile
- Create an image once and run anywhere.
- Lets us change the underlying infra.
So, how do things get deployed?

It all starts with our devs:

- Create feature branch
- Write Code
- Write Tests
- Raise Pull Request
GitHub Workflow

- Create 'feature' branch from 'master'
- Commit changes
- Submit Pull Request
- Discuss proposed changes
- Merge 'feature' branch into 'master'
What happens after code gets merged?
How do we make our deployments easier?
Introducing Deploy Bot

- Serverless Slack Bot
- Written in Python/Flask
- Using the Zappa Framework
- Doesn’t cost anything when it’s not being used
- Deploy from anywhere!
Linux Infrastructure

Nomad Server

1. Fetch CPU Percentage Average from CloudWatch
2. Query cluster for current, min, max job stats
3. Update nomad job counts up, down based on Meta rules

Nomad servers self-discover and register themselves via the `nomad` service in Consul

Nomad ASG servers should be set to min/max 3/3 for staging clusters.
On Production ASG min/max should be 7/7 to ensure quorum during AZ failure.
Note: The nomad UI currently runs as a docker container inside an agent.
Scaling Containers

1. Service is started with metadata metrics per task group
   a. count_max = 10
   b. count_min = 2
   c. scaledown_decrease_count = 1
   d. scaledown_threshold = 20
   e. scaleup_increase_count = 2
   f. scaleup_threshold = 40
   g. threshold_metric = AverageCpuPercent
   h. ...

2. Service runs on each Nomad Agent
   a. Sends metrics to Cloudwatch

3. Lambda executes
   a. Checks metrics against ruleset
   b. Scales up/down
How do we simplify all of this complexity?
What we get

- Windows + Linux containers
- Fast container scale-up
- Fast scale-out
- Cheap instances via Spot, managing spot termination
- Historical metrics job CPU vs deploy versions
- IAM Policies
- Fast deploy/rollback times
- 1hr to build new stacks
We also love to code!

github.com/cvandal/nomad-ui
Windows Infrastructure

Robot Army

Octopus Deploy
Octopus Deploy

- Environments
- Projects
- Project Deployment
- Processes
- Step templates
Octopus Deploy

Environments

Pre-Production

<table>
<thead>
<tr>
<th>VM-CM01</th>
<th>VM-CM02</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cm01.author.preprod.com:10933</td>
<td>Cm02.author.preprod.com:10933</td>
</tr>
<tr>
<td>CM-Master</td>
<td>CM-Master</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VM-CD01</th>
<th>VM-CD02</th>
<th>VM-CD03</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cd01.preprod.com</td>
<td>Cd02.preprod.com</td>
<td>Cd03.preprod.com</td>
</tr>
<tr>
<td>CM-Job</td>
<td>CM-Job</td>
<td>CM-Job</td>
</tr>
</tbody>
</table>

Production

<table>
<thead>
<tr>
<th>VM-CM01</th>
<th>VM-CM02</th>
<th>VM-CM03</th>
<th>VM-CM04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cm01.author.production.com:10933</td>
<td>Cm02.author.production.com:10933</td>
<td>Cm03.author.production.com:10933</td>
<td>Cm04.author.production.com:10933</td>
</tr>
<tr>
<td>CM-Master</td>
<td>CM-Master</td>
<td>CM-Master</td>
<td>CM-Master</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VM-CD01</th>
<th>VM-CD02</th>
<th>VM-CD03</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cd01.production.com</td>
<td>Cd02.production.com</td>
<td>Cd03.production.com</td>
</tr>
<tr>
<td>CM-Slave</td>
<td>CM-Slave</td>
<td>CM-Slave</td>
</tr>
<tr>
<td>CM-Job</td>
<td>CM-Job</td>
<td>CM-Job</td>
</tr>
</tbody>
</table>
Octopus Deploy

Projects

Nikita

Create release

Overview

Release | Staging
---------|--------
0.0.86   | 0.0.86

August 3rd 2017

Nikita

Deploy to Staging (#38)

Task summary | Task log
--------------|---------

Task progress
This task started 3 hours ago and ran for 42 seconds

- Deploy Nikita release 0.0.86 to Staging
- Step 1: Git - Pull (HTTPS)
- Step 2: Give me results
- Step 3: Replace Telegraf Config
- Step 4: replace default website
- Step 5: Test cluster

Overview
Process
Variables
Channels
Releases
Settings

m
Octopus Deploy

Project Deployment

Task log

This task started 3 hours ago and ran for 42 seconds

- Deploy Nikita release 0.0.86 to Staging
  The deployment completed successfully.
  - Step 1: Git - Pull (HTTPS)
  - Step 2: Give me results
  - Step 3: Replace Telegraf Config
  - Step 4: replace default website

WIN-2P28NA2JQS6

Executing script on 'WIN-2P28NA2JQS6'
Name          Application pool  Protocols          Physical Path
-----          ---------------          -------          -----------------------
GiveMeWeather  .NET v4.5          http           C:\Octopus\Applications\GitPull\Staging\Nikita\Git - Pull (HTTPS)\published website
Octopus Deploy

Processes

Deployment process

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Git - Pull (HTTPS)</td>
<td>nikita, nikita-ml4</td>
</tr>
<tr>
<td>2</td>
<td>Give me results</td>
<td>nikita</td>
</tr>
<tr>
<td>3</td>
<td>Replace Telegraf Config</td>
<td>nikita, nikita-ml4, test</td>
</tr>
<tr>
<td>4</td>
<td>replace default website</td>
<td>nikita, nikita-ml4</td>
</tr>
</tbody>
</table>

Add step  Reorder steps
Octopus Deploy

Step Templates

Configure the step using the fields below. Use the Parameters tab to define variables that can be used in scripts and bound fields.

Script

```powershell
if ($env:PATH -notlike "$git") {
    Write-Output "Adding the Git 'bin' directory to the 'Path' environment variable..."
    $path = $env:PATH + "\Program Files\Git\bin"
    [Environment]::SetEnvironmentVariable("Path", $path, "Machine")
}
[System.Reflection.Assembly]::LoadWithPartialName("System.Web")
function Format-UriWithCredentials($url, $username, $password) {
    $url = New-Object "System.Uri" $url
    $url.Scheme += "://"
    if (-not [string]::IsNullOrEmpty($username)) {
        $url = $url + [System.Web.HttpUtility]::UriEncode($username)
    }
    return $url
}
```
Robot Army

Cluster Definition

```yaml
RobotPlatoon stagingNikita
{
    Ensure = "Present"
    Name = "s-nikita"
    Team = "DevOps"
    OctopusProject = "Nikita"
    PlatoonVersion = "ra3"
    Sleepytime = "Yes"
    InstanceType = "t2.nano"
    OctopusVersion = "3.0"
    PublicInstances = $false
    VPC = "Staging"
    HealthCheckEndpoint = "TCP:80"
    minSize = 1
    maxSize = 1
}
```
So, DevOps at Domain

- Incredibly fun
- Nothing like traditional operations
- Lots of coding involved
- Requires a holistic view of what we run and how we’re running it
- Always lots to learn
- Always on the bleeding edge
Thank you.

Domain

Jordan Simonovski
Nikita Gorenbukh