How to be a Network Engineer in a Programmable Age
An evolution that goes beyond Infrastructure as Code and Automation

Hank Preston, Principal Engineer
NetDevOps Evangelist
ccie 38336 R/S

@hfpreston
github.com/hpreston
Topics to Cover

• The Network Engineer of Old
• The Four Ages of Networking
• Cloud to the Rescue
• Enter NetDevOps
• Today’s Network Engineer

The Network Engineer Evolves
Ye ole Network Engineer

What is it? "WR MEM"?

erm...

copy run start

oi...!
Meet Carl the Network Engineer

**Network Skills**
- Spanning-Tree
- Routing Protocols
- QoS
- VPN Design
- Spanning-Tree
- VOIP
- Fibre Channel
- Security Policy
- MPLS
- Did we mention Spanning-Tree?

**Programming Skills**
- TCL
- EEM
- Expect Scripts
The Network...
The Network...
The Network...
The Network...
The Network...
The OSI Model of Networking...

Please don’t ask about this...

- L7: Application
- L6: Presentation
- L5: Session

Oh Yeah... We Got this

- L4: Transport
- L3: Network

Black Magic

- L2: Data Link
- L1: Physical
Networking through the ages...
The Four Ages of Networking....

Stone Age
Spanning Tree
VLANs
The Four Ages of Networking.....
The Four Ages of Networking.....

Stone Age
Spanning Tree
VLANs

Bronze Age
Routing Protocols
WAN Design
IP-magedon

The Renaissance
SDN
OpenFlow
Controllers
Overlays
MP-BGP
VXLAN
Micro-Segmentation
White Box
The Four Ages of Networking.....

Stone Age
- Spanning Tree
- VLANs

Bronze Age
- Routing Protocols
- WAN Design
- IP-magedon

The Renaissance
- SDN
- OpenFlow
- Controllers
- Overlays
- MP-BGP
- VXLAN
- Micro-Segmentation
- White Box

Programmable Age
- Cloud
- Python
- REST / APIs
- NETCONF / YANG
- “Fabrics”
- Network Function Virtualization (NFV)
- Containers
- DevOps
- NetDevOps!
“Digitization” of the Enterprise

I want an agile bimodal hybrid cloud so we can develop containerised serverless trustless microservices applications to take us Digital to avoid disruption from any unicorns. Oh.. and I want DevOps...Two of those...
**App Economy**
User Expectations and Agility

**Internet of Things**
If it isn’t connected, don’t bother...

**Tech Unicorns**
Low barrier of entry for disruptors
The Cloud You Plan to Build 😊
The Cloud You Plan to Build 😊

The Cloud You End Up With 😞
The New Infrastructure Stack

Development Environment
- Vagrant, Docker, Vim, Slack, Spark, Git

Operating System
- CoreOS, Rancher, RedHat, Ubuntu, Microsoft

Infrastructure
- UCS/ACI, HP, vSphere/NSX
The New Infrastructure Stack

<table>
<thead>
<tr>
<th>Development Environment</th>
<th>Vagrant, Docker, Vim, Slack, Spark, Git</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delivery Pipeline</td>
<td>GitHub, BitBucket, Jenkins, Team City, Drone, Puppet, Ansible, Chef</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cloud Management and Automation</th>
<th>UCS Director, vRealize, OpenStack, AWS, CloudCenter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>CoreOS, Rancher, RedHat, Ubuntu, Microsoft</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>UCS/ACI, HP, vSphere/NSX</td>
</tr>
</tbody>
</table>

Users and Developers

Architects and Operators
The New Infrastructure Stack

<table>
<thead>
<tr>
<th>Users and Developers</th>
<th>Development Environment</th>
<th>Vagrant, Docker, Vim, Slack, Spark, Git</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delivery Pipeline</td>
<td>GitHub, BitBucket, Jenkins, Team City, Drone, Puppet, Ansible, Chef</td>
</tr>
<tr>
<td></td>
<td>Scheduling and Placement</td>
<td>Docker/Swarm, Kubernetes, Mesosphere, Tectonic, Rancher, Rocket</td>
</tr>
<tr>
<td></td>
<td>Container Layer</td>
<td>HAProy, Cassandra, RabbitMQ, Hadoop, Consul</td>
</tr>
<tr>
<td></td>
<td>Applications and Middleware</td>
<td>UCS/ACI, HP, vSphere/NSX</td>
</tr>
<tr>
<td></td>
<td>Cloud Management and Automation</td>
<td>UCS Director, vRealize, OpenStack, AWS, CloudCenter</td>
</tr>
<tr>
<td></td>
<td>Operating System</td>
<td>CoreOS, Rancher, RedHat, Ubuntu, Microsoft</td>
</tr>
<tr>
<td></td>
<td>Infrastructure</td>
<td>UCS/ACI, HP, vSphere/NSX</td>
</tr>
</tbody>
</table>

Architects and Operators
## The New Infrastructure Stack

<table>
<thead>
<tr>
<th>Users and Developers</th>
<th>DevOps Engineers</th>
<th>Architects and Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Development Environment</strong></td>
<td>Vagrant, Docker, Vim, Slack, Spark, Git</td>
<td><strong>Infrastructure</strong></td>
</tr>
<tr>
<td><strong>Delivery Pipeline</strong></td>
<td>GitHub, BitBucket, Jenkins, Team City, Drone, Puppet, Ansible, Chef</td>
<td><strong>Cloud Management and Automation</strong></td>
</tr>
<tr>
<td><strong>Scheduling and Placement</strong></td>
<td>Docker/Swarm, Kubernetes, Mesosphere, Tectonic, Rancher, Rocket</td>
<td><strong>Operating System</strong></td>
</tr>
<tr>
<td><strong>Container Layer</strong></td>
<td>HAProxy, Cassandra, RabbitMQ, Hadoop, Consul</td>
<td></td>
</tr>
<tr>
<td><strong>Applications and Middleware</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The New Infrastructure Stack

<table>
<thead>
<tr>
<th>Development Environment</th>
<th>Vagrant, Docker, Vim, Slack, Spark, Git</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delivery Pipeline</td>
<td>GitHub, BitBucket, Jenkins, Team City, Drone, Puppet, Ansible, Chef</td>
</tr>
<tr>
<td>Scheduling and Placement</td>
<td>Docker/Swarm, Kubernetes, Mesosphere, Tectonic, Rancher, Rocket</td>
</tr>
<tr>
<td>Container Layer</td>
<td>HAProxy, Cassandra, RabbitMQ, Hadoop, Consul</td>
</tr>
<tr>
<td>Applications and Middleware</td>
<td>UCS Director, vRealize, OpenStack, AWS, CloudCenter</td>
</tr>
<tr>
<td>Cloud Management and Automation</td>
<td>CoreOS, Rancher, RedHat, Ubuntu, Microsoft</td>
</tr>
<tr>
<td>Operating System</td>
<td>UCS/ACI, HP, vSphere/NSX</td>
</tr>
<tr>
<td>Infrastructure</td>
<td></td>
</tr>
</tbody>
</table>
Network Stakeholders

**Network Builders**
- Traditional networking teams
- Design, Build, and Maintain the Network
- Responsible for Care and Feeding

**Network Consumers**
- The users of the network
- Looking to consume network “services”
- The network is a “utility” – It should just work
CARL, WHAT IS THIS “NETDEVOPS” I KEEP HEARING ABOUT?

YOU’LL LOVE IT CAP! NETDEVOPS BRINGS THE CULTURE, TECHNICAL METHODS, STRATEGIES, AND BEST PRACTICES OF DEVOPS TO NETWORKING!

Enter NetDevOps!
Moving to a NetDevOps Culture and Mindset

Organizations Today have a “Culture of Fear”

- Problems occur during change
- Change is seen as failure
- Changes happen rarely
- Changes are big and complicated
- The team isn’t well practiced
- Change is seen as high risk
Moving to a NetDevOps Culture and Mindset

Organizations Today have a “Culture of Fear”

- Change seen as failure
- Changes happen rarely
- Changes are big and complicated
- Problem occurs during change
- The team isn’t well practiced
- Change seen as high risk

NetDevOps will require a “Culture of Change”

- Change was uneventful
- Change was tested and verified
- The team is well practiced
- Each change is small
- Changes are regular activities
- Change seen as success
- Each change is small
- The team is well practiced
- Change was tested and verified
NetDevOps Pipeline: “Treating the Network as Code”

- Network Configuration stored in Source Control
- Changes are proposed in code “branches”
- CICD Build Servers deploy and test proposed configurations
- Successful configurations automatically deployed to “Production”
The NetDevOps Engineers Tool Bag
(Example tools, not comprehensive)

- Distributed Source Control
  (git, Subversion, Mercurial, GitHub, BitBucket, GitLab)

- Build Server
  (GitLab, Jenkins, Team City, Drone)

- Configuration Management
  (Ansible, Puppet, NSO, NAPALM, DIY)

- Network Test Tooling
  (pyATS, TRex, Robot, Behave)

- Telemetry & Monitoring
  (ELK, Grafana, Pipeline, UTM)

- Network Device
  
  - YANG/Native Data Model
  - Configuration Data
  - Operational Data

- Network Virtualization Platforms
  (VIRL/CML, NFVIS, Vagrant)

- Development Environment
  (Vagrant, NSO, VIRL/CML)

- Test Environment
  (VIRL/CML)

- Production Environment
The NetDevOps Engineers Tool Bag
(Cisco Products and Projects)

Distributed Source Control

Build Server

Configuration Management
Network Service Orchestrator (NSO)

Network Test Tooling
pyATS, TRex

Telemetry & Monitoring
Pipeline, UTR

Network Device
DNA Center Platform, APIC, Meraki, IOS XE, IOS XR, NX-OS, Firepower, UCS

Development Environment
NSO, VIRL, VNFs

Test Environment
VIRL/CML

Production Environment

Network Virtualization Platforms
NFVIS
VIRL/CML
NSO
VNFs
5 Stages of Grief

Denial

Anger

Bargaining

Depression

Acceptance
IT'S ALIVE!!!
Carl’s 3 Step Approach to Network Programmability

**Phase 1**
- Python
- REST APIs
- JSON/XML
- git/GitHub

**Phase 2**
- Linux Skills
- Ansible
- Docker
- NETCONF/YANG

**Phase 3**
- Linux Networking
- Container Networking
- NFV

**As Needed**
- Network Controllers
- IOT Networking
- Cloud Networking
- ”DevOps”
Carl has Embraced Programmability!

**Network Skills**
- Layer 2 & 3 Fundamentals
- Quality of Service
- Security and Segmentation
- Linux Networking
- Container Networking
- Cloud Networking
- IOT Networking
- Model Driven Programmability
- Network Function Virtualization

**Platform Skills**
- Linux Administration
- Container Fundamentals
- Micro Service Platforms
- Cloud Fundamentals

**Programming Skills**
- Data Formats (ex: JSON/YAML)
- Python and APIs (ex: REST)
- Source Control (ex: git)
- Configuration Management (ex: Ansible)
I can't keep up Cap. It used to be simple! Now it's all automated, composable, fungible, immutable, policy driven, infrastructure as code....

You ok Bro?

Well.. what's the quote? Oh yeah.. "If you dislike change you're gonna dislike irrelevance even more"

I'VE ALWAYS ENJOYED YOUR PEP TALKS!!!
Review

- We looked back on the history of the network and network engineering
- Traveled through the Four Ages of Networking
- Talked about the impact “Cloud” has had on IT
- Explored how NetDevOps will change Networking
- Considered the skills a network engineer needs today
What do do next?

**NetDevOps Readings**
- Embrace NetDevOps, Say Goodbye to a "Culture of Fear"
- NetDevOps Goes Beyond Infrastructure as Code
- What does "Network as Code" Mean?
- A Network Engineers Journey in Programmability
- NetDevOps and the Rise of the Programmable Network

**NetDevOps Learning Resources**
- Network Programmability Basics Video Course
- Network Programmability for Application Developers

**NetDevOps Videos**
- How to be a Network Engineer in a Programmable Age
- Network as Code in Action
- Benefits of Configuration Management
Got more questions? Stay in touch!

Hank Preston

hapresto@cisco.com
@hfpreston
http://github.com/hpreston

@CiscoDevNet
facebook.com/ciscodevnet/
http://github.com/CiscoDevNet

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public