Telematics Container Cloud Platform
TCCP Architecture & Deployment

2018-03-05: Rick Davis
The power of Verizon

Verizon Network Solutions

Partner Ecosystem

IoT Ecosystem

Oath

Telematics Portfolio

Verizon Security
Agenda

1. Goals
2. Architecture
3. Automation
4. Collaboration & Lessons Learned
5. Q&A
Goals

What do we want to do?
Business Drivers

Increase Development Velocity
- Monolith to Microservice
- Latest Software Frameworks
- Containerized
- 100% CICD

Increase Infrastructure Velocity
- Site Rebuild & App Deploys < 12hr
- Repeatable/Predictable
- 100% CICD

Future Proofing
- Flexible & Extensible
- Easily Scalable
- Cloud Adaptable & Agnostic
Infrastructure Goals

Architecture
• Cost Conscious
  • Open Hardware
  • Open Source Software, Commercial Support Optional

• Global Deployment Capability w/Vendor Support

• Public & Private Cloud Adaptable

Automation
• Server & Network Provisioning
• Integrated Validations
• Infrastructure-as-Code

Extensibility
• Platform with Swappable Components
• Avoid Lock-in, Everywhere
  • Hardware, Software, and Cloud
Architecture

How are we going to do it?
Architecture

eBGP Clos Fabric
- One Protocol/Simplified OPS
- Quagga with ECMP on Hosts
- Anycast Support

Spine/Leaf Topology
- Dedicated Storage Network
- Dedicated Border/Peering

Simple Scalability
- 100’s of Racks/Site

Open Hardware & Software, Commercial Support Optional
Hardware Platforms

Dell EMC DSS 9000 Servers & Storage
- Open Compute based infrastructure avoids lock-in
- Rack-Level Hyperscale Infrastructure
- Rack-Level Management – IPMI/Redfish
- Flexible Compute & Storage Configurations
- OEM support & reliability, ODM pricing

Dell EMC Open Networking Switches
- Swappable Network OS via ONIE
- Latest high performance platform (e.g. Tomahawk 4x25G/100G)
- Commodity silicon avoids vendor lock-in (Z9100 ~= FB Wedge100)
- OEM support & reliability, ODM pricing
Logical Architecture

Service-by-Service Failover

Service-by-Service Isolation
- VXLAN/MACVLAN Isolation
- Line Rate ACL’s in Silicon

Anycast Load Balancing

Ceph Block/Object/File Store
- Persistent Container Storage via REX-Ray

Independent Clusters
- Failure Domain Isolation
- Site-to-Site Replication in App/DB Control
Docker Swarm – Our Perspective (Early 2016)

**Decision Factors**

- Developer Simplicity & Talent Pool
- Blue/Green Deployment & Rolling Upgrades
- Network, Volume, Logging Drivers
- API Compatibility & Ecosystem Tools
- Windows/Mac/Linux Hosts
- Windows & Linux Containers

- **Reduced Complexity**
- **Swappable Components**
- **Large Ecosystem**
Sounds good...what if it doesn’t work?

Swap Components

- Need cloud?
  - Swap provisioning to cloud
- Need Kubernetes?
  - Enable in Docker or swap it
- Need etcd?
  - Add it or swap it

Handover

- Certify hardware with other VZ platforms
- Handover few racks for platform deployment
- Convert apps
- Validate, update CICD
- Deploy in PROD
- Convert remaining racks

Migrate & Repurpose

- Validate/convert apps and infrastructure services
  - Service Discovery, Logging, Monitoring, etc, etc
- Validate, update CICD
- Deploy in PROD
- Repurpose existing racks
Automation

How do we keep doing it?
## Automation Goals

<table>
<thead>
<tr>
<th>Simplified</th>
<th>Standardized</th>
<th>End-to-End</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Automated</td>
<td>Ansible First</td>
<td>Server AND Network</td>
</tr>
<tr>
<td>Easily Repeatable &amp; Predictable</td>
<td>Bash/Python Second</td>
<td>• Configuration AND Provisioning</td>
</tr>
<tr>
<td>Zero Touch &amp; Zero Downtime</td>
<td>Same CI Pipeline, Every Site</td>
<td>Public &amp; Private Cloud Adaptable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Automated Maintenance &amp; Healing</td>
</tr>
</tbody>
</table>
CI Pipeline & Deployment Management

GitLab Flow based model

Git Tags = Immutable

Maintenance Automation

Fully Automated: Eliminate human error

Code/OPS/Change Review: Merge Request

Branch Protections: RBAC

Information contained herein is provided AS IS and subject to change without notice. All trademarks used herein are property of their respective owners.
Simple Scalability – Adding Racks

Add rack group (P4)
Update device details
Execute CI Pipeline
Simple Scalability – Adding Sites

Add datacenter config
Add racks
Execute CI Pipeline
Collaboration & Lessons Learned

What should we have done?
Collaboration

**Collaboration**
- Issues/Merge Requests, Milestones, and Boards
- GitLab Flow Based Model
- Source Controlled CI
- Reusable repositories

**Communication**
- Wiki & GitLab Pages
- ChatOps
- CI Analytics

**Flexibility**
- Single Solution, Multiple Integrations
- Push & Pull Git Mirroring
- Granular Permissions by Team/Group/Project
- Multi-project CI pipelines
Lessons Learned

Modularize
- Separate repositories by function
- Consolidate roles
- Logically segregate CI pipeline

Standardize
- Programming language
- Single source of truth – Ansible inventory
- Adoption of open source roles

Operationalize
- Ensure idempotency
- Tools/Alerts/Dashboards in roles

Document
- Document
- Change processes
- Code is your documentation
Thank you.
Z9100 Port Mapping & Buffer Zone Utilization

SET1: 7 Compute
- Uplink 1, 2 & 3
- Rack 1, ToR1
- Rack 3, ToR1
- Rack 5, ToR1
- Rack 7, ToR1

SET2: 7 Compute
- Uplink 1, 2 & 3
- Rack 1, ToR2
- Rack 3, ToR2
- Rack 5, ToR2
- Rack 7, ToR2

SET3: 7 Compute
- Uplink 4, 5 & 6
- Rack 4, ToR1
- Rack 4, ToR2

SET4: 7 Compute
- Uplink 4, 5 & 6
- Rack 2, ToR1
- Rack 6, ToR1
- Rack 2, ToR2
- Rack 6, ToR2

© Verizon 2018 All Rights Reserved
Information contained herein is provided AS IS and subject to change without notice. All trademarks used herein are property of their respective owners.
MACVLAN & Anycast is Easy :)