T · ·Mobile·

On-Premise Kubernetes at T-Mobile

James Webb
MTS, Platform Engineering

Brendan Aye
Sr. MTS, Platform Engineering

T · ·Mobile·

Who We Are

T-Mobile Platform Engineering

- **Started as a three-person team in May 2016**
  - Goal: Bring Cloud Foundry to the Enterprise
- **Now 25 strong supporting IT facing IaaS, CaaS & PaaS platforms wearing many hats**
  - Infrastructure Engineers
  - System Administrators
  - Developers
  - Platform Administrators
  - Product Managers
  - Customer Success Engineers
- **Part of a larger organization supporting all On-Premise IT infrastructure for T-Mobile**

What We Manage

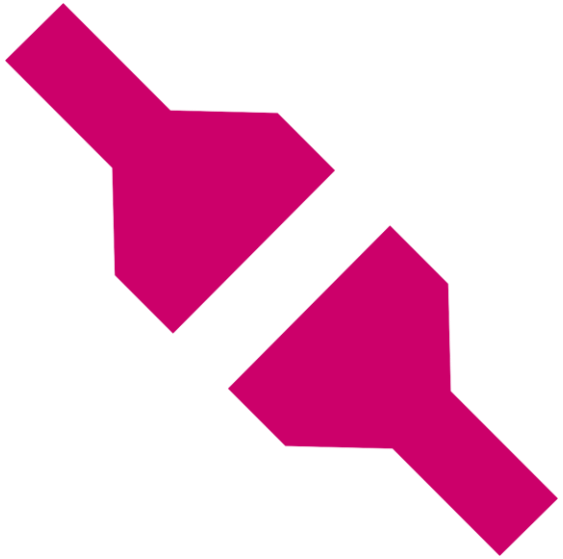
- **PaaS (Pivotal Cloud Foundry)**
  - 12 customer facing foundations in two data centers
  - 34,000 application instances (containers)
    - Roughly 40% production/60% non-production
  - 300M+ production transactions/day
  - Associated platform hosted data services (MySql, RabbitMQ, Gemfire, Redis, …)
- **CaaS (Kubernetes)**
  - 24 clusters, both single and multi tenant
    - Mix of open source and vendor deployments
  - 5 live applications (some turning up just this week)
  - ~1M production transactions/day
- **IaaS (BOSH)**
  - For platform and customer needs

Business Impact of PaaS

- **Speed to market**
  - DevOps teams can onboard and push apps to production same day
  - Some teams went from 6 months dev to prod cycle to weeks
- **Increased application performance & reliability**
  - Average 43% reduction in app response time
  - 83% fewer incidents, resolved 67% faster
- **Deployment agility**
  - 10x increase in planned deployments
  - Daytime changes, blue/green deployments, canary testing
- **Developer efficiency**
  - Platform abstractions let developers focus on development
  - No longer need to manage OS patching, load balancing, certificates – all built in to the platform
- **Workload consolidation**
  - In some cases 12x efficiency gain in HW footprint
  - Adjacent workloads benefit from proximity

If PaaS Is So Great….

- **The CaaS gap**
  - No standard offering for teams to run containers
    - Shadow Docker
  - Not everything is a good fit for Cloud Foundry
    - Non-native containers
      - Vendor supplied docker containers becoming more common
      - Lift & shift
    - Non-HTTP/HTTPS traffic management limited
    - No persistent storage
      - NFS volume services available for PaaS, but a trap
      - Platform data services meet some, but not all application needs
  - No platform orchestration
      - Complex/stateful application management needs to be external



T · · Mobile ·

Why On-Premise?

- **Data Center Gravity**
  - Data
    - Beyond the Chandrasekhar limit
  - Network
    - Latency matters
  - Security
    - On-Premise controls and patterns well understood
  - Organizational
    - Lack of public cloud expertise
    - Most compliance
  - Cost
    - Strong economies of scale in data center - *if* we execute
    - Capex vs Opex
- **Destiny**
  - Own it
- **Public Cloud available**
  - Public Cloud team offers K8S and many other services

T · · Mobile ·

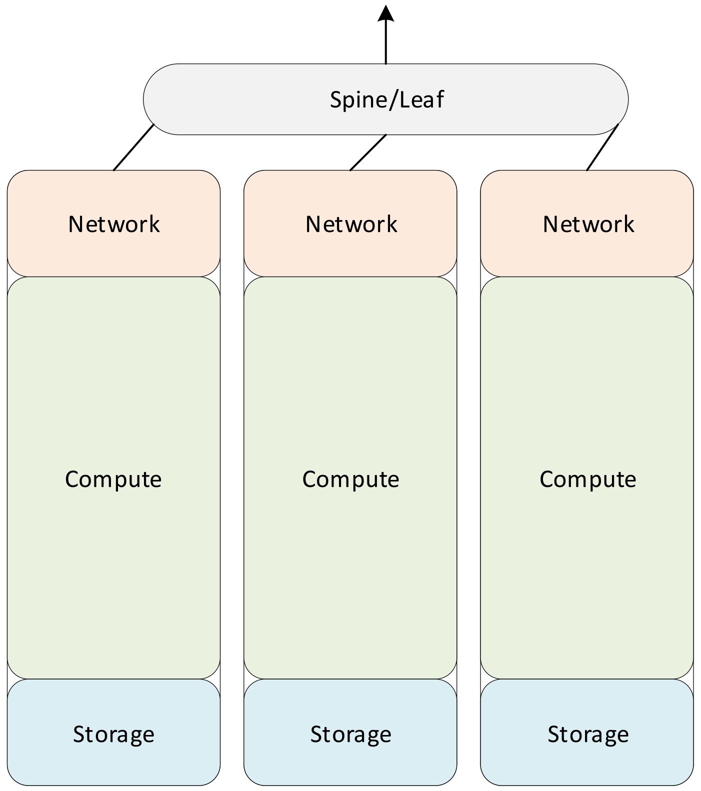
CaaS Requirements

- **Platform Team:**
  - Highly Available at every level
    - Control Plane (etcd/API)
    - Worker Nodes
    - Authn/Authz
  - Automated Deployment
    - Control Plane (OpsMan/Bosh)
    - Cluster builds
  - No Downtime Lifecycle Management
    - K8S Upgrades
    - OS Patching
    - Infrastructure Maintenance
  - LDAP Integration
  - API Configurability
  - Automated Ops

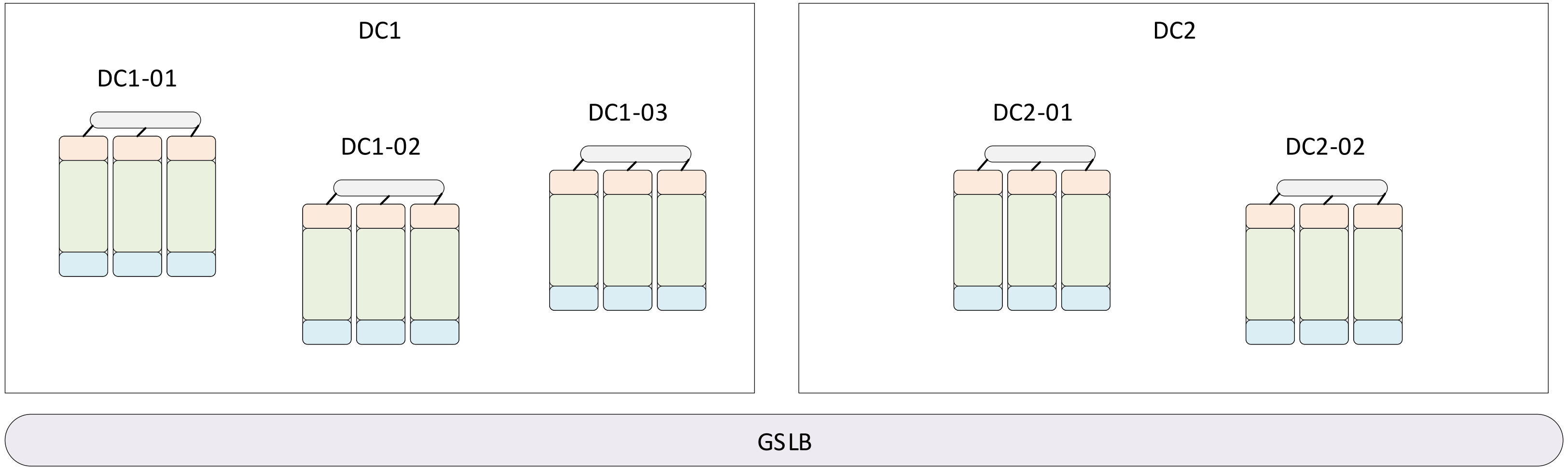
- **DevOps Teams:**
  - Native K8S Experience
  - Container Orchestration
  - PaaS-like support experience
    - Out of the box cert/load balancing
    - OS Patching
    - Infrastructure Maintenance
    - Persistent Storage
    - Single AZ
    - Cross AZ replication
    - Cross Cluster replication
  - TCP Ingress
    - Service type LoadBalancer
  - Centralized Logging/Metrics
    - APM + Platform

“Region” Architecture

- **Each Region a set of 3 availability zones**
  - Shared nothing architecture
  - Each AZ is a single rack with independent:
    - Network
    - Compute
    - Storage
- **Isolated behind a spine/leaf pattern**
    - High-bandwidth/low latency east-west network
    - Intra-AZ traffic isolated behind border leaves
- **Region max capacity**
  - ~55 TB Memory
  - ~2200 Cores
  - ~2 PB Storage



Mutli-DC



- **Multiple regions in each data center**
  - Separate regions for production and non-production
  - Near/Near/Far deployment strategy for applications with a data center preference
  - GSLB available for active/active & active/passive cross-region deployments

T · · Mobile ·

K8S Dial Tone

- **Automated cluster deploy with PKS**
  - **Concourse install of PKS framework**
    - Infrastructure (Compute, Network, Storage)
  - **Concourse deployment of cluster**
    - Select a plan, define # of workers and deploy
- **Post cluster configuration**
  - **Once cluster is deployed we T-Mobilize the cluster**
    - Monitoring/Telemetry (Prometheus)
    - Persistent Storage (Portworx)
    - Ingress (NGINX)
    - Logging (send to Splunk)
    - External Load Balancers configured for API, HTTPS Ingress & TCP Ingress
- **Day 2 Ops**
  - **Org/User Mangement**
    - GitOps (in progress)
  - **Support**

Cluster Ownership

- **Enable, but don’t burden platform customers**
  - **Platform Engineering team manages:**
    - Infrastructure (Compute, Network, Storage)
    - Cluster installs, upgrades, decommissions
    - Base cluster tooling and capabilities (monitoring, logging, ingress, persistent storage, ...)
  - **Multi-tenant clusters**
    - More efficient use of resources
    - Namespace isolation for DevOps teams
    - Provide ingress with default certificate for HTTPS, but customers can also bring their own cert
  - **Single tenant clusters**
    - Sensitive environments
    - High utilization customers
    - Advanced customers who need more control

Early Successes

- **Live Apps**
  - Critical order management, retail store and call centers apps live
  - https://maps.t-mobile.com
- **Upgrades/Patching**
  - Seamless upgrade from 1.10 -> 1.11
  - Automation allowed for same day, no impact patching of recent API CVE
- **Persistent Storage**
  - In use by platform team and customers

Challenges

- **TCP Ingress**
  - Fully automated type LoadBalancer still elusive
  - Workarounds in place, but high support overhead
- **Adoption/Velocity**
  - Developer community starting from scratch with K8S
  - Cloud Native COE ramping up to help
- **Lack of API configurability**
  - On the PKS roadmap

Lessons Learned

- **Own what you can**
  - Not realistic, so friends close, enemies closer
- **Limit blast radius any of one install (cluster, region, database, foundation, ...)**
  - Too big to fail isn't just a Wall Street problem
- **Upgrade/patch often**
  - Customers informed, not consulted
  - Automate repaves to happen even when they're not needed
  - Don't let individual apps dictate schedules
- **Set expiration dates**
  - Don't let clusters become pets
  - Encourage customers to be able to deploy to multiple targets
- **Automate everything**
  - Well, of course
- **Create a community for your customers to interact with support teams and each other**
  - Slack is our first stop for help

On the Horizon

- Hosted Data Services
- Istio/Envoy
- Knative
- Operators
- Federation

Tool Chest

Q&A

T · · Mobile ·

T · · Mobile ·®