Cloud Native Applications in a Telco World
How Micro Do You Go?

Azhar Sayeed
Chief Architect
Red Hat, Inc.

Dejan Leskaroski
Director, Product Management
Affirmed Networks

The industry leader in virtualized mobile networks.
Agenda

- Microservices – definition and benefits
- Why Containers?
- 5G – driver for cloud native approach
- Application decomposition
  - Network functions – design choices
  - Orchestration
- Summary
CLOUD & VIRTUALIZATION JOURNEY
DIGITAL TRANSFORMATION FOR TELCO & IT

Development Process
- Waterfall
- Agile
- DevOps

Application Architecture
- Monolithic
- N-Tier
- Microservices

Deployment & Packaging
- Physical Servers
- Virtual Servers
- Containers

Application Infrastructure
- Datacenter
- Hosted
- Cloud
Why Micro-services? – Micro-services & Containers

Microservices Architecture is independent from containers

- Microservices architecture is about writing applications so that components can be independently updated and delivered to complete the product
  - May use containers for each of the components
  - Monolith vs componentized
  - Each component can evolve independently
- Network Functions can also be re-factored

Containers – Encapsulating micro services
VALUES OF MICROSERVICES

FAST TIME TO MARKET

Small autonomous services can be developed and delivered faster

EFFICIENCY

Automating delivery and monitoring of small services is easier

SCALABILITY

Fine grained scalability is easier and uses less resources
Containers - An Evolution in Application Deployment

Definition: Software packaging concept that typically includes an application and all of its runtime dependencies. Where hypervisors provide a logical abstraction of a full system (hardware, BIOS, OS), Containers provide an abstraction of the user space and share the same OS, services, and hardware.

- Enable efficiency and automation for microservices, but also support traditional applications
- Enable faster and more consistent deployments from Development to Production
- Enable application portability across 4 infrastructure footprints: Physical, Virtual, Private & Public Cloud
A Word About 5G
RAN Evolution

Virtualized RAN: Virtualized BBU (LTE/4G), CU/DU (5G)
**PACKET CORE EVOLUTION**

- **Box / Device centric LTE/4G**
- **5G - Cloud Based**
  - CP-DP Separation
  - UPF is controlled by AMF and SMF
  - Data plane extensibility

---

**OpenStack or KVM**
- vBBU
- vMME
- vSGW
- vPGW
- OpenStack

**Control Plane – Mobility, Sessions & Service Management**
- Localized GW or Central GW Data Plane

---

**5G RAN 4G RAN**

---

**NG - Core**

---

**Virtualized CO infrastructure**

Virtualized RAN: Virtualized BBU (LTE/4G), CU/DU (5G) NG-CORE with UPF and NGSI, AMF etc
NG-Core and cRAN – require micro services models

- DU, CU and vBBU – Containers attractive to deliver control and data plane functions
- 5G NG-Core - Separate Control and User Plane allows flexible deployment of functions
  - => Orchestration models needed to place functions - Kubernetes
  - => Granularity functions or services for flexibility – Microservices and re-usability
- Support for Edge computing and efficient UPF re-selection/redirection
  - => Common orchestration of core and other functions
  - => Re-usability of components
- Support Network Slicing based on modular design and multi slice connectivity from UEs
  - => Resource partitioning from edge to core
- Stateless functions – compute decoupled from storage
What does Cloud-Native mean?

“Cloud native computing uses an open source software stack to deploy applications as microservices, packaging each part into its own container, and dynamically orchestrating those containers to optimize resource utilization. Cloud native technologies enable software developers to build great products faster”

- Scale elastically
- Resilient to failures
- Instrumented to provide insights
- Repeatable
- Automated
- Utilize – cloud storage, queuing, caching, messaging etc

https://www.cncf.io/
Platform for delivery of Microservices

- Load Balancing, Scaling / Elasticity
- Discovery
- Logging
- Monitoring
- Resilience
- Tracing
- Authentication
- Invocation Messaging / IPC
- Build, Deployment Pipeline
- Container Mgmt.

PaaS (cloud-based)
Middleware services
Application lifecycle management
Mobile
Cloud Native Enablers - Recap

**Orchestration**
- Externalized clustering, load balancing, and connectivity management

**Platform-as-a-Service (PaaS)**
- Logging, Tracing, Performance Monitoring, API Management and much more

**HTTP APIs**
- Standardized integration technology; Publishable (OpenAPI 3.0 / Swagger)

**Stateless Applications**
- Ease of Life Cycle Management – "Cattle not Pets"

**Containers**
- Dynamic orchestration tools, fast instantiation, efficient deployment unit
Microservice Decomposition
Balancing Performance & Flexibility

Coarse Grained

- U-plane packet processing
- Call-Control State Machines
- Protocol Handlers, IP Routing
- FCAPS, Operational Support

Fine Grained

Ultra-High Performance

Ultra-Agile Software Releases

Business Value: Affirmed has the right architecture that strikes the right balance between performance and agility.
Affirmed Cloud Native Ecosystem

Virtualization PaaS (Optional)

Deployed Anywhere, Deployed in Working Cluster:

Cloud Native PaaS

IaaS
PaaS Slicing & Multi-Tenancy

Cloud Deployed PaaS Slices

- Istio
- MongoDB
- Jaeger
- Grafana
- Kibana
- Prometheus
- Elastic
- Jaeger
- Istio
- MongoDB
- etcd
- kubernetes
- NF Microservice Federation
- NF Microservice Federation
- K8s Clusters
Cloud Native Components & Common Microservices

Service Specific Microservices
(e.g., building blocks of AMF, SMF, UPF, etc.)

Data Center IP Network

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices

Acuitas EMS

Cloud VNF Manager

Cloud Native Components

Security

SDN Control

Service Specific Microservices

(e.g., building blocks of AMF, SMF, UPF, etc.)

Networking & Routing Common Microservices

Protocol Handling & Load Balancing Common Microservices
Network Function Deployment Modes

- Cloud Native Network Function Mgr (CNFM)
- K8s Bare Metal
- K8s over VM
- K8s over VM (Cloud VMaaS / Container-aaS / K8s-aaS)
- PaaS Slice
- PaaS Slice
Principles of Microservices Development

- Model around a domain – In our case today packet core gateways - NG-Core
- Culture of Automation – Automated deployment, automated scale and monitoring
- Independent deployment of each microservice
- Active monitoring of services
- Isolation of failures
- Dependency management – declaration and isolation
- Concurrency – process model
- Disposability – fast startup and graceful shutdown
- Logs and monitoring metrics – Event streaming and Telemetry
- Build, Test, Release and Run – Full DevOps
Traditional, stateful, and cloud-native apps

Developer Experience

Enterprise Kubernetes++ container orchestration

Container Linux – provides Isolation and Security
Summary

• Refactoring a monolith necessary for building cloud native applications
• Leverage existing micro services to build new applications
• Containerization can provide isolation of microservices
• How micro do you go?
  • Independence
  • Scale
  • Efficiency
  • Lifecycle management
• Microservices necessary for building next generation cloud native functions – Network functions and gateways for 5G
• Red Hat OpenShift has already built in capabilities and DevOps environment for delivering cloud native applications
THANK YOU