Microservices on the Edge: The Infrastructure Impact

Ram (Ramki) Krishnan: Industry Consultant, SupportVectors, Co-chair IETF NFV Research Group

Chris Wright: Vice President and Chief Technologist, Office of Technology at Red Hat
Presentation Outline

- Enterprise Microservices Backgrounder
  - Enterprise Infrastructure Architecture Impact
- Microservices on the Edge
  - Edge Infrastructure Architecture Impact
  - Microservices for Virtual Network Functions – New Potential Models
- Common Infrastructure Architecture for Microservices
  - Containers, Resource Modelling, SLA Monitoring and Policy Abstractions
  - Open Source/Standards Efforts Next Steps
Enterprise Microservices - Backgrounder

Classic Application Architecture
Any organization will produce a design whose structure is a copy of the organization's communication structure -- Melvyn Conway, 1967

Key Microservice Architecture Tenants
- Service split based on business need
- Decentralized governance – different processes and data stores
- Module reuse - share common modules such as logging, monitoring
- Loosely coupled - scale independently, new service flexibility
- Standardize the APIs across microservices

Adapted from: https://martinfowler.com/articles/microservices.html
Enterprise Microservices: Real-time Transaction Travel-booking Example

**Individual services:**
Seven tiles in the figure.

**Interaction:**
Arranged to show which microservices can interact with other microservices.
bookFlights service – receives external customer request.

**Independent scale:**
The services' different vertical heights represent how they are used in different quantities in relation to one another.

**Loosely coupled – flexible to add new service:**
Example -- add discount coupon service

Infrastructure Architecture Impact – An Exemplary Deployment Model

**Storage Intensive Nodes**
e.g. Red Hat Ceph, Microsoft Azure storage
HW Acceleration e.g.: Compute/Network – RDMA (RoCE, InfiniBand etc.), Network/Storage – x86 AES-NI, Intel Quick Assist, Cavium (ARM) ThunderX2, Customizable FPGA etc. (TLS, Secure storage etc.)

**Compute Intensive Nodes**
e.g. Machine Learning, 3D application streaming
HW Acceleration e.g.: GPU, customizable FPGA (Parallel floating point etc.), RDMA (RoCE, InfiniBand etc.),

**General Purpose Nodes**
e.g. Web/Middle Tier applications
HW Acceleration e.g.: Network crypto – x86 AES-NI, Cavium ThunderX2 (TLS etc.)

**Memory Intensive Nodes**
e.g. SAP Hana, Microsoft SQL server, Big Data Apache Spark
HW Acceleration e.g.: Compute/Network – RDMA (RoCE, InfiniBand etc.), Network crypto – x86 AES-NI, Cavium ThunderX2, Customizable FPGA etc. (TLS etc.)

**Takeaways**
- Towards a Converged infrastructure -> Flexible node personality is important
- HW acceleration key for deterministic performance, especially for latency sensitive workloads -> Reconfigurable components are highly desirable
Infrastructure Architecture Impact: Real-time Transaction Travel-booking Example

**General Purpose Nodes**
- HW Acceleration e.g.: Network crypto – x86 AES-NI, Cavium ThunderX2, Customizable FPGA etc. (TLS, IPSEC etc.)

**Memory Intensive Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.), Network crypto – x86 AES-NI, Cavium ThunderX2, Customizable FPGA etc. (TLS etc.)

**Storage Intensive Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.), Network /Storage crypto – x86 AES-NI, Intel Quick Assist, Cavium ThunderX2, Customizable FPGA etc. (TLS, Secure storage etc.)

**App Tier – Book Flight Microservice Aggregator**
- Web Front End – Book Flight Customer Input

**App Tier – Create Customer Microservice**
- Database Tier – Create Customer Trigger

**App Tier – Adjust Inventory Microservice**
- Database Tier – Adjust Inventory Trigger

**Network Fabric**
- Storage Tier – Create Customer Trigger
- Storage Tier – Adjust Inventory Trigger
- Database Tier – Adjust Inventory Trigger
- App Tier – Create Customer Microservice

**Takeaways**
- No. of hops proportional to number of microservices, bursty nature of data (Storage I/O block operations, HW Protocol (TCP etc.) offload batching, CPU batch processing etc.) -> service assurance challenge for latency sensitive applications
- HW acceleration is key for deterministic performance -> challenge managing heterogeneity
- Dynamic service creation -> challenge managing dynamic scaling in a shared heterogenous infrastructure
- Database decoupling/scale/PACLEC requirements -> challenge in choosing the right database
Up Next

• Enterprise Microservices Backgrounder
  • Enterprise Infrastructure Architecture Impact
• Microservices on the Edge
  • Edge Infrastructure Architecture Impact …
Edge Computing – Use Case Summary

Use cases from MEC -- http://www.etsi.org/technologies-clusters/technologies-multi-access-edge-computing

• Video analytics

• Location services

• Internet-of-Things (IoT)
  • Examine in detail a low-latency service such as air quality measurement

• Augmented reality

• Optimized local content distribution

• Data caching
Edge Computing IoT Microservices: Real-time Analytics Air Quality Measurement Example

Alerting Microservice: Trigger air quality alerts - leverage statistics and machine learning jobs.


Event reporting Microservice: Process dynamic events from Mobile and Web applications.

Data Reception, Storage & Transformation Job: Receive raw sensor data from IoT device - store in file system. Perform data validation and transform data into (JSON) format.

Contextual Enrichment Job: Add device specific data to transformed JSON format.


Takeaways
- Microservices architecture key to distributed computing across smart sensors, IoT gateways, Edge DC, Cloud DC
- HW acceleration key to deterministic performance and reducing edge node footprint

Adapted from: http://airboxlab.github.io/streaming/microservices/iot/spark/real-time/2016/08/29/streaming-microservices.html
Infrastructure Architecture Impact:
Real-time Analytics IoT Air Quality Measurement Example

**General Purpose Nodes**
HW Acceleration e.g.: Network crypto – x86 AES-NI, Cavium ThunderX2, Customizable FPGA etc. (TLS, IPSEC etc.)

**Compute Intensive Nodes**
(Spark ML etc.)

**Memory Intensive Nodes**
(SQL/NoSQL DB, Spark etc.)
HW Accln.: x86 AVX, ARM Cortex M4

**Storage Intensive Nodes**
(HDFS etc.)
HW Acceleration e.g.: Storage crypto – x86 AES-NI, Intel Quick Assist, Cavium ThunderX2, Customizable FPGA etc. (TLS, Secure storage etc.)

**Data Reception and Storage Microservice**
HW Accln.: MQTT (TLS etc.) decryption

**AI Tier - Machine Learning Job**
HW Accln.: x86 AVX, ARM Cortex M4

**Analytics Tier – Statistics Streaming Job**

**Network Fabric**

**Analytics Tier – Alerting Streaming Job**
HW Accln.: Machine Learning model evaluation

**Storage Tier – Statistics Streaming Job**
HW Accln.: Secure storage, Storage integrity check

**Storage Tier – Machine Learning Job**
HW Accln.: Secure storage, Storage integrity check

**Alerting Microservice**

**Takeaways (similar to enterprise travel booking example)**
- No. of hops proportional to number of microservices, bursty nature of data (Storage I/O block operations, CPU batch processing etc.) -> service assurance challenge for latency sensitive applications such as real-time alerting
Up Next

- Enterprise Microservices Backgrounder
  - Enterprise Infrastructure Architecture Impact
- Microservices on the Edge
  - Edge Infrastructure Architecture Impact
  - Microservices for Virtual Network Functions – New Potential Models ...
Potential Microservices Architecture for NAT VNF

**Deployment Model**
- Read/Write intensive NAT tables (key-value pair hash table) Memory intensive nodes
- Packet processing - General purpose nodes, - Optional NAT table caching

**General Purpose Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.), SR-IOV

**Memory Intensive Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.)

**Network Fabric**

Adapted from: http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/hotmiddlebox/p49.pdf

**Takeaways**
- Benefits: Packet processing decoupled from database management
- Challenges: Tables are in RAM with higher Capex than classic solution, Additional network hop per packet
Potential Microservices Architecture for Stateless Firewall VNF

**Deployment Model**
- Read intensive Firewall tables (key-value pair hash tables for different + optionally TCAM) - Storage intensive nodes
- Packet processing - General purpose nodes, Firewall table caching, counter batch update
- PACELC theorem in action – Firewall table caching – consistency vs latency tradeoff

**General Purpose Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.), SR-IOV

**Storage Intensive Nodes**
- HW Acceleration e.g.: Compute /Network – RDMA (RoCE, InfiniBand etc.), Lookup - TCAM

**Firewall Packet Processing Microservice**

**Firewall Table Storage (SSD etc.) Microservice**

**Network Fabric**

**Takeaways**
- Benefits: Packet processing decoupled from database management, Lower Capex than classic solution
- Challenges: Additional network hop per packet batch
Up Next

• Enterprise Microservices Backgrounder
  • Enterprise Infrastructure Architecture Impact

• Microservices on the Edge
  • Edge Infrastructure Architecture Impact
  • Microservices for Virtual Network Functions – New Potential Models

• Common Infrastructure Architecture for Microservices
  • Containers ...
Containers – FCAPS framework (1)

Key Microservice Tenant - App and Database separation
• Containers can be created/destroyed on the fly and ideal for apps
• Stateless apps are desirable for containers – does not preclude stateful applications (e.g. classic VNFs)

“F” in FCAPS – Fault Management
• PACELC theorem availability vs consistency tradeoff

“C” in FCAPS – Configuration Management
• Open source implementations for microservice, e.g. Kubernetes/Mesos service implementation
• Open source HW acceleration integration – work in progress

“A” in FCAPS – Accounting Management for billed infrastructure
• Open source implementations for microservice, e.g. Kubernetes Datadog integration
• Open source HW acceleration integration – work in progress
Containers – FCAPS framework (2)

“P” in FCAPS – Performance Management
- PACELC theorem latency vs consistency tradeoff – Recall firewall VNF example
- SW isolation (memory, CPU, storage etc.) in a virtualized infrastructure – supported by Linux Kernel
- HW isolation/monitoring (cache etc.) – Intel RDT [Ref. 1] cache partitioning/monitoring etc.
- Performance Monitoring with HW acceleration (e.g. SR-IOV, RDMA) – work in progress

“S” in FCAPS – Security Management
- SW security – Linux Namespaces, SELinux, AppArmor etc.
- HW security - *difficult to match VMs*
  - Containers (or processes) in VMs - two hardware indirection tables for virtual address translation
  - Native Containers on Host OS - single hardware indirection table for virtual address translation
  - Intel Clear Containers [Ref. 2] – HW security similar to VMs but other challenges
- HW security requirements – dictated by deployment model
  - SaaS – Typical deployment model is native containers on Host OS
  - PaaS/IaaS – Typical deployment model is Containers (or processes) in VMs

Ref. 2: https://clearlinux.org/features/intel%C2%AE-clear-containers
Containers and NFV (3)

Practical Deployment

- NFV deployments are starting out as SaaS
- Occasionally need to run third party apps
- Viable for a predominantly containerized deployment as long as there are no performance issues; third party apps can be run as VMs

Next Steps

- Call for participation in NFVRG
  - Expand on current draft -- https://www.ietf.org/archive/id/draft-natarajan-nfvrn-containers-for-nfv-03.txt
  - Detailed security best practices leveraging Selinux, AppArmour etc.
Up Next

• Enterprise Microservices Backgrounder
  • Enterprise Infrastructure Architecture Impact

• Microservices on the Edge
  • Edge Infrastructure Architecture Impact
  • Microservices for Virtual Network Functions – New Potential Models

• Common Infrastructure Architecture for Microservices
  • Containers
  • HW Acceleration Resource Modelling and SLA monitoring ...
Some of the important Modelling Aspects of HW Accelerators with constrained resources

HW capabilities: Features supported by the accelerator
• E.g. Crypto Acceleration (AES-NI, Intel QuickAssist etc.)
  • Different crypto algorithms (AES-CBC etc.), Protocols (IPSEC, TLS etc.)

HW capacity: Operations per second
• E.g. Crypto Acceleration (Intel QuickAssist etc.) bandwidth

HW Topology: How the accelerators are interconnected from the CPU perspective
• E.g. Multi-GPU <-> CPU PCI-e interconnect topology

SW capabilities: OS Kernel driver and user space library integration
• E.g. Linux/Windows OS support, Libcrypto/Libssl library support
Small buffer switch can be modelled as a HW Accelerator – important for low-latency SLA monitoring/enforcement for RDMA based-protocols such as RoCE

- As an example, OCP switch designs [Ref. 1] use Broadcom Trident (Alpha Networks SNX-60x0-486F etc.) and Broadcom Tomahawk (Facebook Backpack, Edgecore Networks AS7300-54X etc.)

- Broadcom Trident family and Tomahawk family have different internal buffering architectures, i.e. different HW topologies
  - Trident has a single shared buffer pool for all ports
  - Tomahawk has multiple buffer pools, one per port group

- Dynamic switch buffer pool utilization with topology knowledge is also a key metric for SLA monitoring besides egress queue depth etc.

Ref. 1: http://www.opencompute.org/wiki/Networking/SpecsAndDesigns
HW Acceleration Resource Modelling is a key area where the community can bring value

- Can leverage the industry efforts on related topics
  - **OpenStack Enhanced Platform Awareness** -- [https://01.org/sites/default/files/page/openstack-epa_wp_fin.pdf](https://01.org/sites/default/files/page/openstack-epa_wp_fin.pdf)
  - **Kubernetes GPU support** -- [https://github.com/kubernetes/community/blob/master/contributors/design-proposals/gpu-support.md](https://github.com/kubernetes/community/blob/master/contributors/design-proposals/gpu-support.md)

Low-latency network SLA monitoring/enforcement is another key area for additional IETF contributions

- Can leverage several IETF drafts in the area
  - [https://datatracker.ietf.org/doc/draft-krishnan-opsawg-in-band-pro-sla/?include_text=1](https://datatracker.ietf.org/doc/draft-krishnan-opsawg-in-band-pro-sla/?include_text=1)
  - [https://tools.ietf.org/html/draft-brockners-inband-oam-requirements-03](https://tools.ietf.org/html/draft-brockners-inband-oam-requirements-03)
  - More …
Up Next

- **Enterprise Microservices Backgrounder**
  - Enterprise Infrastructure Architecture Impact

- **Microservices on the Edge**
  - Edge Infrastructure Architecture Impact
  - Microservices for Virtual Network Functions – New Potential Models

- **Common Infrastructure Architecture for Microservices**
  - Containers, HW Acceleration Resource Modelling and SLA monitoring
  - **Policy Abstractions** ...
Policy Abstractions

The right infrastructure Policy Abstractions are key to using the HW acceleration resource modelling and delivering low-latency SLAs

- The industry favored implementation model in OpenStack, Kubernetes etc.
  - JSON/YAML for policy language
  - Policies managed by the infrastructure orchestrator admin (OpenStack, Kubernetes etc. admin)

- This is a key area where the community and IETF can bring value
  - Can leverage the industry efforts on related topics
Policy Abstractions – Example OpenStack JSON Policy

For "low-latency" workloads:
• At least 8GB of free ram
• At least 8 free vCPUs
• NUMA awareness
• X86 AES-NI for crypto

['or', ['and', ['=', '$user.type', 'low-latency'], ['>', '$host.free_ram_mb', 8*1024], ['>', '$host.vcpus_total' - '$host.vcpus_used', 8], ['=', '$host.crypto.x86-aes-ni', 'True'], ['not', ['=', '$host.numa_topology', 'None']]]]
Up Next

• Enterprise Microservices Backgrounder
  • Enterprise Infrastructure Architecture Impact

• Microservices on the Edge
  • Edge Infrastructure Architecture Impact
  • Microservices for Virtual Network Functions – New Potential Models

• Common Infrastructure Architecture for Microservices
  • Containers, Resource Modelling, SLA Monitoring and Policy Abstractions
  • Open Source/Standards Efforts Next Steps
• Containers – Contribution to NFVRG and beyond
  • Expand on current draft (https://www.ietf.org/archive/id/draft-natarajan-nfvrг-containers-for-nfv-03.txt) based on discussion points
  • Detailed security best practices leveraging Selinux, App Armour etc.

• HW Acceleration Resource Modelling/Policy Abstractions - key value add area for community/IETF
  • Kubernetes GPU Support -- https://github.com/kubernetes/community/blob/master/contributors/design-proposals/gpu-support.md
  • Kubernetes Resource QoS -- https://github.com/kubernetes/community/blob/master/contributors/design-proposals/resource-qos.md
  • RDMA-based Distributed Tensorflow on Apache Spark -- https://yahoosng.tumblr.com/post/157196488076/open-sourcing-tensorflow-on-spark-distributed-deep

• Low-latency network SLA monitoring/enforcement – key contribution area leveraging current work
  • https://datatracker.ietf.org/doc/draft-krishnan-opsawg-in-band-pro-sla/?include_text=1
  • https://tools.ietf.org/html/draft-brockners-inband-oam-requirements-03