THE PATH TO DATA PLANE MICROSERVICES

Ray Kinsella
February 2018
Email : ray.kinsella [at] intel.com
IRC: mortderire

Contributions from

- Kuralamudhan Ramakrishnan (Intel)
- John DiGiglio (Intel)
- Dana Nehama (Intel)

Legal Disclaimer

General Disclaimer:

Technology Disclaimer:

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].

Performance Disclaimers:

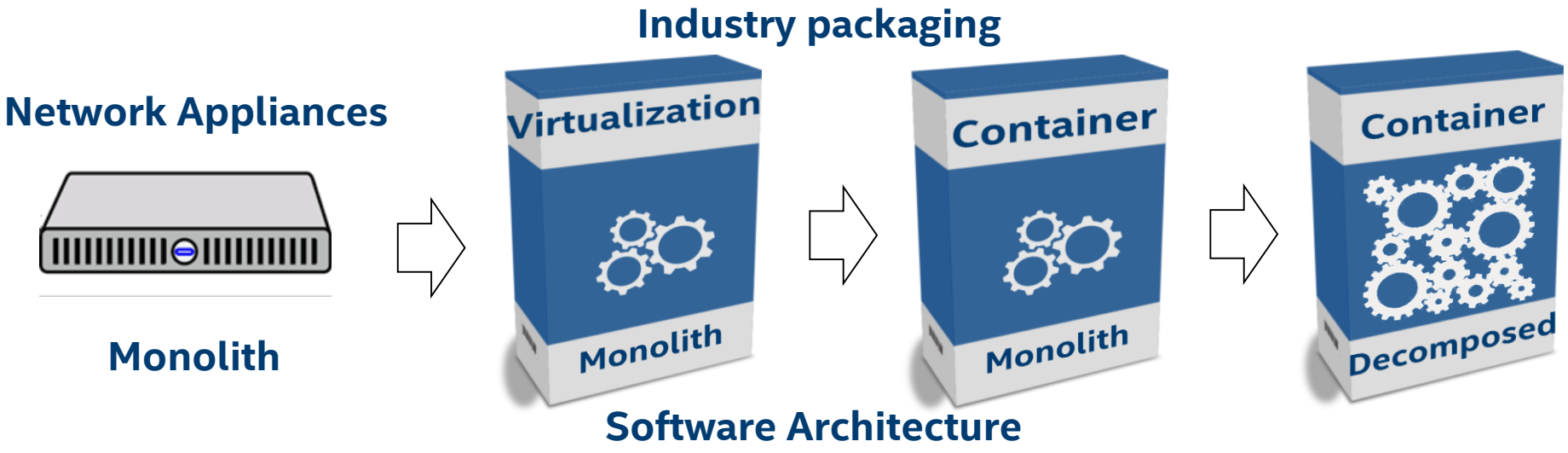
Cost reduction scenarios described are intended as examples of how a given Intel- based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Agenda

- **Network Function Evolution**
- **Containerized Network Functions**
- **Cloud Native Network Functions**
- **Summary**

Network Function Evolution



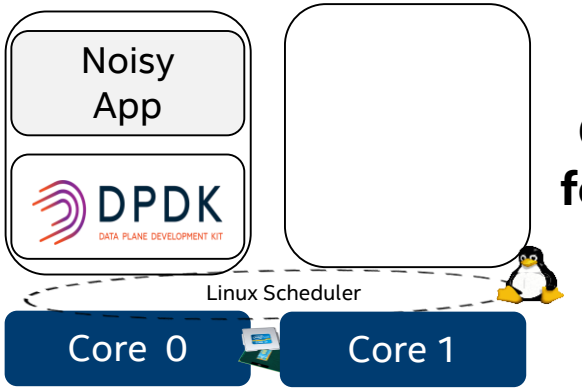
What is driving Containers?

The 12 Factor APP

12 factors (solid principle for Cloud Software Architecture)	Codebase	One codebase tracked in revision control, many deploys
	Dependencies	Explicitly declare and isolate dependencies
	Config	Store configuration in the environment
	Backing Services	Treat backing services as attached resources
	Build, release, run	Strictly separate build and run stages
Application Decomposition	Processes	Execute the app as one or more stateless processes
	Port binding	Export services via port binding
	Concurrency	Scale out via the process model
	Disposability	Maximize robustness with fast startup and graceful shutdown
	Dev/prod parity	Keep development, staging, and production as similar as possible
	Logs	Treat logs as event streams
	Admin processes	Run admin/management tasks as one-off processes

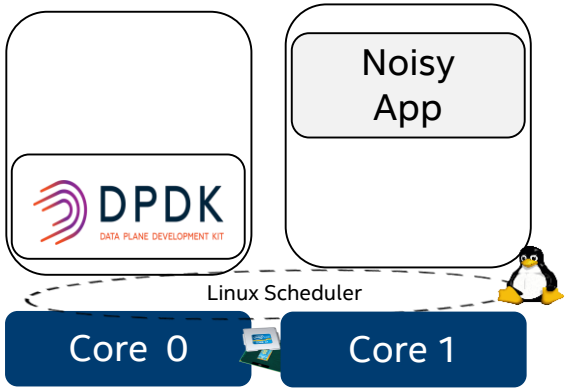
https://12factor.net/

Compute



CPU Manager for Kubernetes

Core Isolation

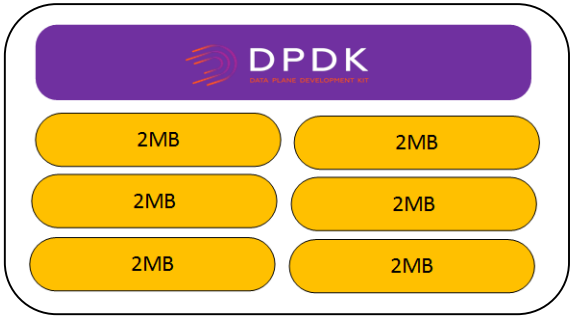


Memory

Kubernetes

Hugepage Enhancements for Kubernetes

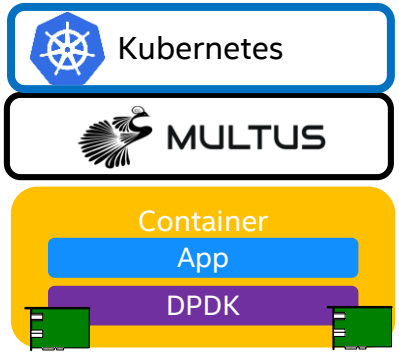
Container



Also…

- Platform (node) feature discovery
- Platform telemetry
- and more.

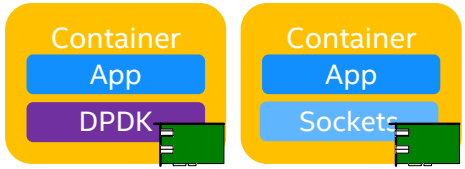
Container I/O



Multus

What is it?
CNI provider for Kubernetes to enable multiple interfaces per Container.

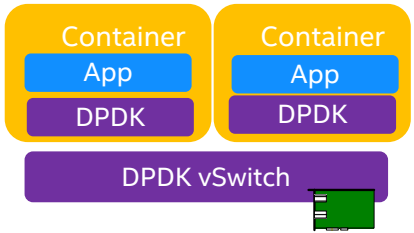
- Functional separation of control and data network planes
- Link aggregation
- Network segregation



SR-IOV

What is it?
Virtual Function Passthrough for Container

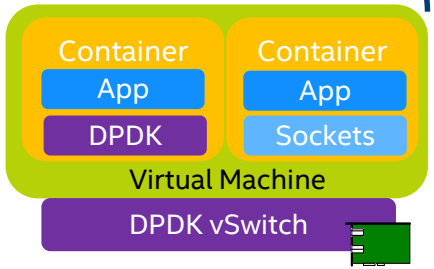
- **High Throughput**
- **Low Density**



Virtio-User

What is it?
Container Virtual Interface for **Bare-metal** deployments

- **Supports**
  - Virtual Network Functions
- **Good Throughput**
- **Good Density**



Master-VM

What is it?
Container Virtual Interface for **Virtualized** deployments

- **Supports**
  - Virtual Network Functions
  - Socket Based Applications
- **Good Throughput**
- **Good Density**

Cloud Native Network Functions

Microservices Enabling

Container Enabling

Data plane Microservices

Resource sharing API

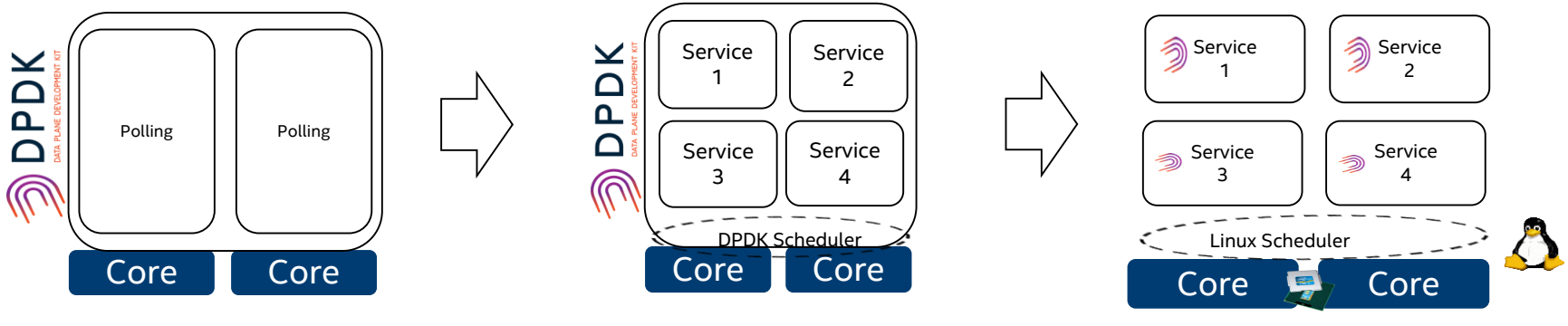
CPU Sharing

I/O Sharing

Memory Sharing

Same APP, same DPDK API(s), deployment specific behaviour!

Compute: Lightweight scheduling models



Monolithic

In-Process Scheduler

Multi-Process Scheduler

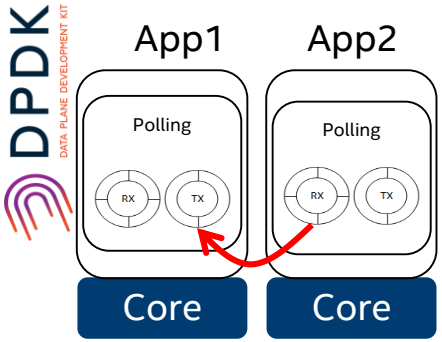
Memory: Lightweight memory models

Monolithic

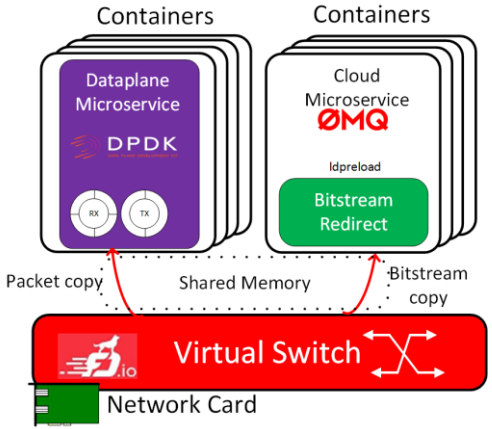
Dynamic Allocation

4K page allocation

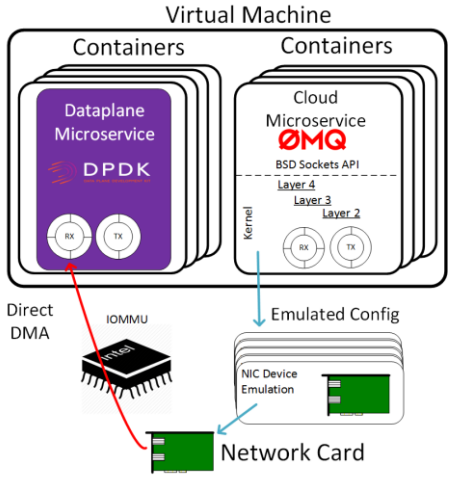
Cloud Native I/O



Current



Scalable Virtual Switch



Accelerated Virtual Switch

Scalable and accelerated Virtual Switching

Data plane microservice models

Model	In-process Microservices	Multi-process Microservices	Multi-node Microservices
Why?	Highest Performance	Multi-process scaling	Multi-node scaling
Scheduling	DPDK Scheduler	Cooperative OS	Cooperative OS
Memory	Monolithic	Dynamic	Dynamic
Transport	Mem Ring	Mem Ring, vSwitch, HW accelerated	RPC a over reliable protocol
High Availability	No	Yes	Yes
Live Migration	No	No	Yes

Summary

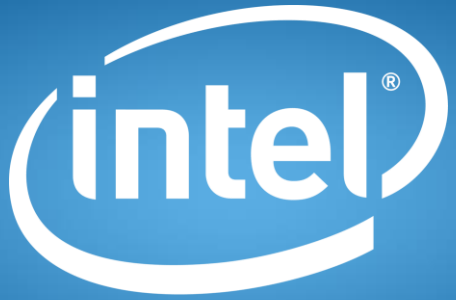
Containerized Network Functions are here!

- Support available on Intel's github (github.com/Intel-Corp) and upstream in Kubernetes.
- See our "Experience Kit" (intel.ly/2rXMswf) for related deployment guidelines; source code, app notes etc.

Cloud Native Network Functions are coming!

- Emerging support in DPDK and FD.io, e.g. new memory model targeted for DPDK 18.02 release.

E-mail: ray.kinsella [at] intel.com IRC: mortderire



experience
what's inside™