



DRA for Networking

FOSDEM 2025

Douglas Smith

Principal Software Engineer, OpenShift Engineering @ Red Hat, Inc.

Miguel Duarte

Principal Software Engineer, OpenShift Engineering @ Red Hat, Inc.



Doug Smith

- Technical lead for OpenShift Network Plumbing Team in OpenShift Engineering
- Multus CNI maintainer
- Network Plumbing Working Group member
- Blog: <https://dougbtv.com>



Miguel Duarte

- OpenShift Virt network engineer
- Network Plumbing Working Group member
- Blog: <https://maiqueb.github.io>

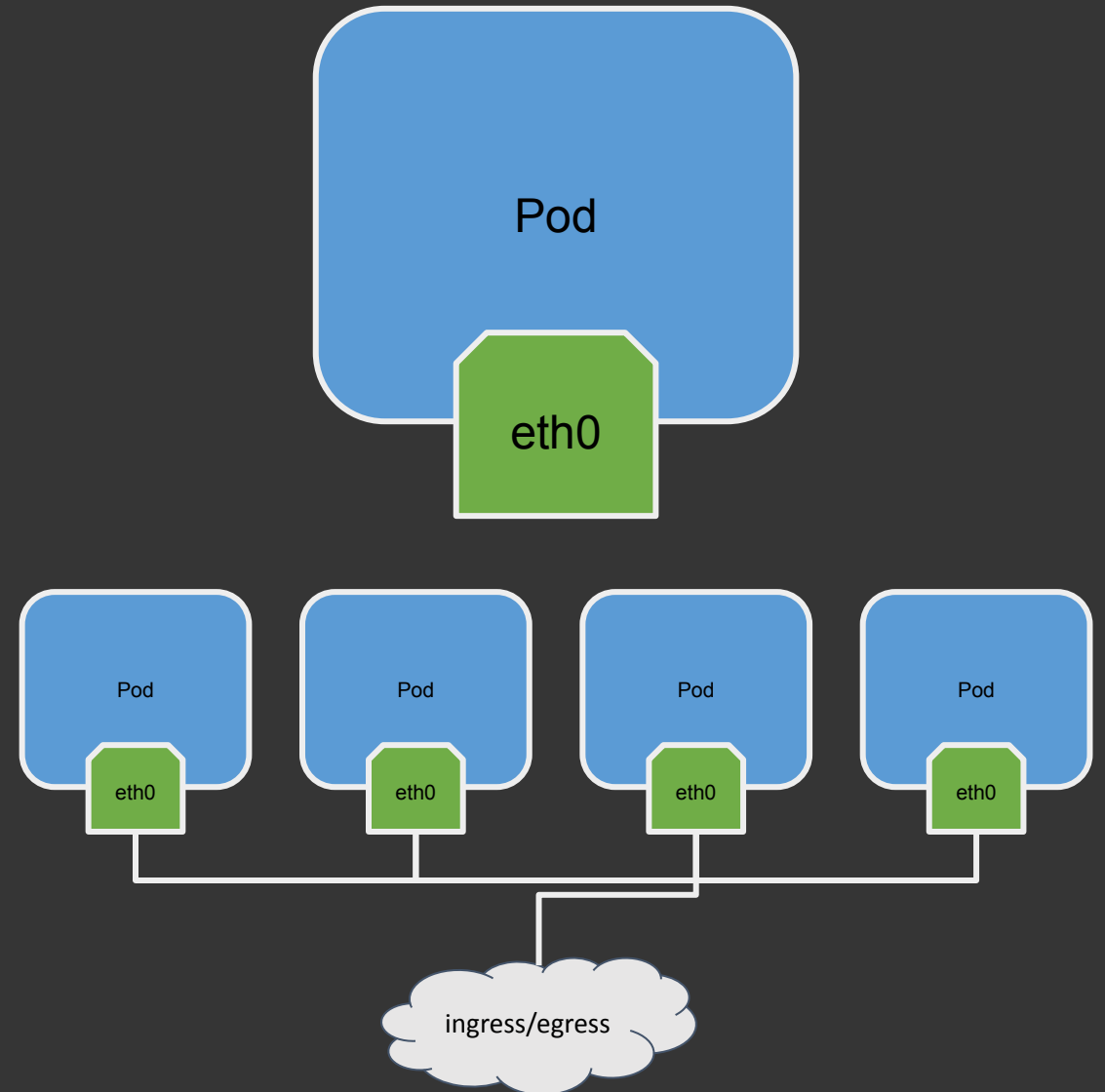
Agenda

- ▶ Problem Statement
 - Multinetworking: As it is today
- ▶ Then the DRA bomb drops
- ▶ A tour of the resources
- ▶ Call to action

The Problem. (and current solutions)

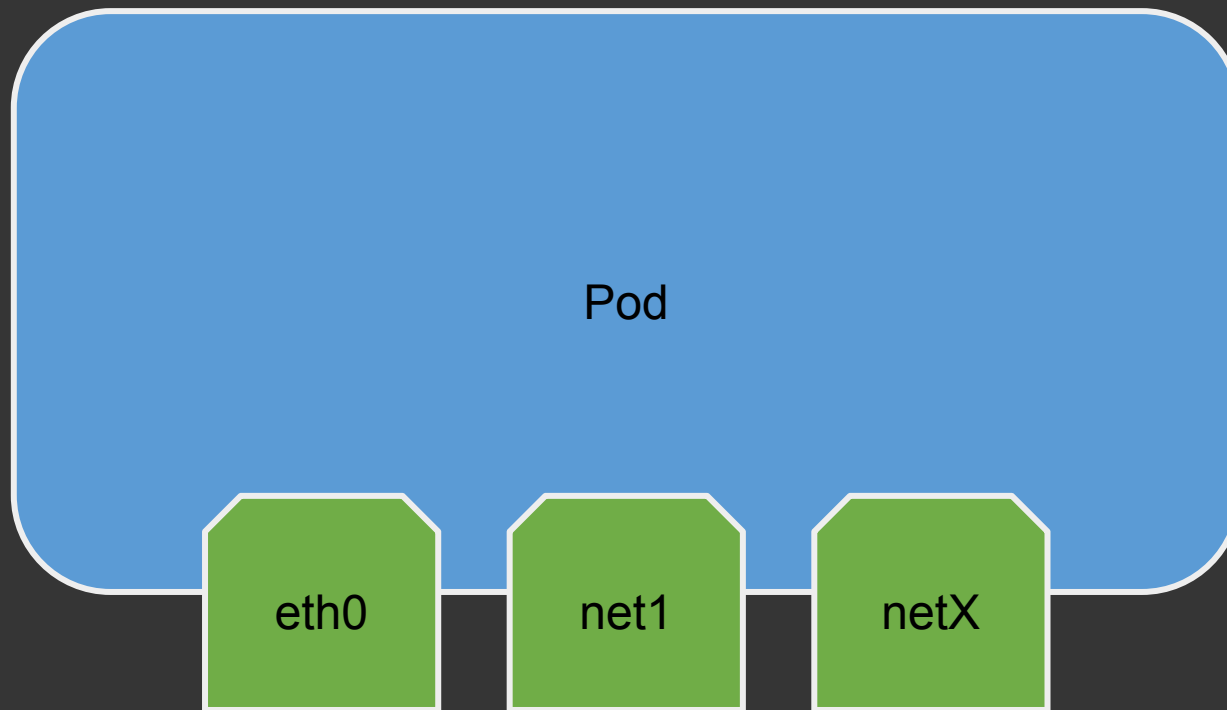
First off – Pods in Kubernetes get just ONE lone network interface...

Today, by default, we have just one network interface on each pod, that pod is promised network connectivity among pods in the network...



But what if you need more than one network?

Often in advanced network use cases, there's more than one network and more than one interface to it – and you need advanced connectivity to complex network topologies, in order to isolate traffic, connect to existing networks, and more.



There's one solution today, and there was one proposed solution...

Network Plumbing Working Group / Multus CNI *CNI JSON inside YAML and "out-of-tree"*

```
---
kind: Pod
metadata:
  name: pod1
  annotations:
    k8s.cni.cncf.io/networks: '{"name":
"media-net", "interface": "net1"}'
spec:
  [...]
```

```
---
kind: NetworkAttachmentDefinition
[...]
spec:
  conf: '{
    "type": "bridge",
    "name": "media-net",
    "ipam": {"type": "static"}
  }'
```

K8s Native Multi-networking / PodNetwork *All YAML and all "in tree"*


```
---
kind: Pod
metadata:
  name: pod1
spec:
  networks:
  - podNetworkName: media-net
    interfaceName: net1
  [...]
```

```
---
apiVersion: v1
kind: PodNetwork
metadata:
  name: media-net
spec:
  provider: "myprovider.io/application"
```

People problems are the hardest ones.

This was a
political
problem.

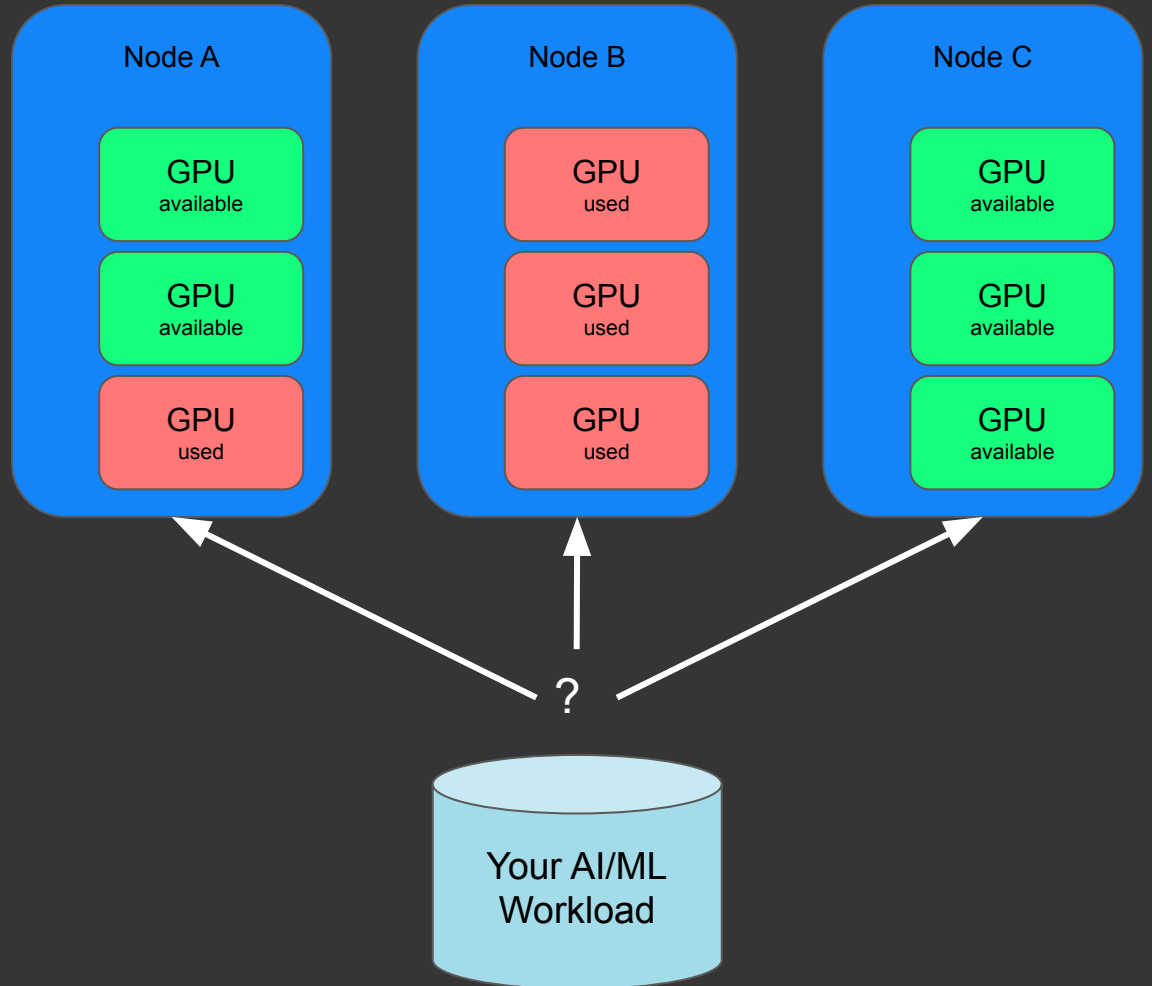
```
---  
kind: Pod  
metadata:  
  name: pod1  
spec:  
  networks:  
  - podNetworkName: media-net  
    interfaceName: net1  
  [...]
```



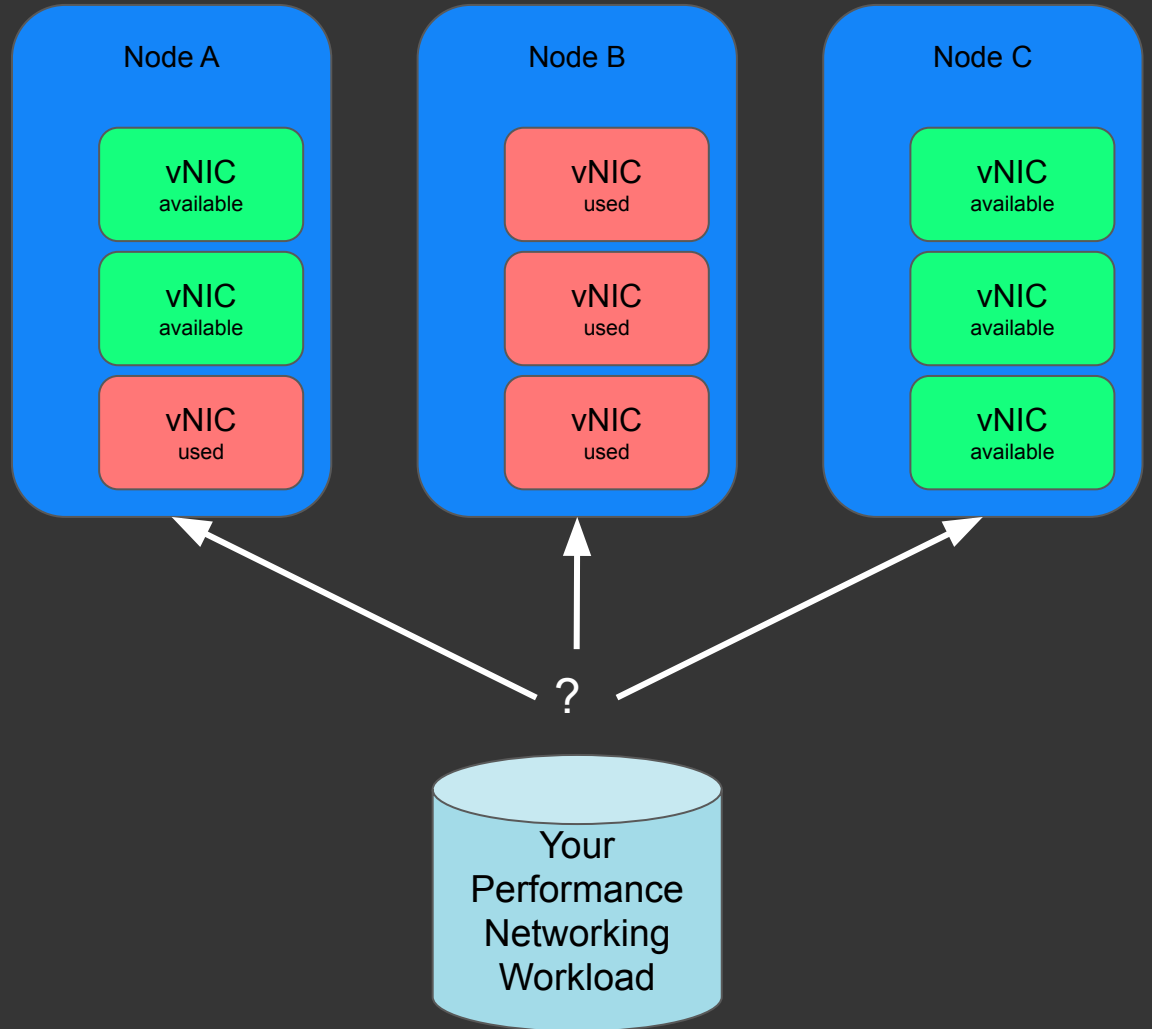
So, what the heck is DRA?

What is DRA?





- Dynamic Resource Allocation is “an API for requesting and sharing resources between pods and containers inside a pod” [-K8s.io](https://k8s.io)
- Currently beta in 1.32



Well, you could say the same thing for NICs!



Wait – don't we have Device Plugins?

	Device Plugins	DRA
Introduced in...	Beta in K8s 1.8	Beta in K8s 1.32
Resource Model	Extended Resources: <code>vendor.com/device=X</code>	<code>ResourceClass</code> & <code>ResourceClaim</code>
Sharing across pods?	 No (one pod per device)	 Yes (multi-pod resource sharing possible)
Per node daemon required?	 Yes (device plugin on each node)	 No (cluster-wide controller)

Why DRA for Networking?

WHY DRA?

- GPU + NIC allocation is pushing the issue
 - In training scenarios (especially) we need high bandwidth RDMA + infiniband networks for training across machines
 - Hopefully users can use a common method of allocation
- And we really still don't have a common Kubernetes native way to express networking resources.



Why not CNI?

- CNI and Kubernetes are two different APIs, like oil and water.
- CNI is “container orchestration engine agnostic”
- This looks like a lot of overheard for someone who’s building k8s controllers.



Are there downsides to DRA for networking? Yes.

- There's more moving parts other than "just CNI"
- It probably needs abstraction to be useful to admins, because otherwise it's tough to do by hand
 - ...But great for kubernetes controllers.



But CNI will still be there...

- Legacies are legacies, after all.
 - Everything today is built with CNI.
- There's a DRA-CNI driver being built as a reference.



A tour of the community.

Involved parties...

- Kubernetes Device Management WG
 - Working on “structured parameters for DRA”
- Kubernetes Multinetwork WG
 - Specification for a k8s native multinetwork expression
- Network Plumbing WG
 - Originators of a specification for multihomed pods in Kubernetes

Resources

- github.com/LionelJouin/network-dra
 - The DRA networking PoC by Lionel
- [kubernetes-sigs/cni-dra-driver](https://kubernetes-sigs.github.io/cni-dra-driver)
 - Brand spanking new!
- [johnbelamaric/wg-device-management](https://github.com/johnbelamaric/wg-device-management)
 - Especially see [the SR-IOV example](#)

In closing.

- What's the future?
 - We're seeing PoCs now!
- ...Get involved. The time is now!
 - You can seriously make an impact and the more community voices for your problems counts, big time!

If you “learn” one thing in this presentation, let this be it.

So, why DRA for networking?!

The answer to “the” political problem:

A Kubernetes native way to express
consuming ~~resources~~ network
attachments

Check out the blog article!



...or go to doughbtv.com