Considerations for Benchmarking Network Performance in Containerized Infrastructure

draft-ietf-bmwg-containerized-infra-03

Minh-Ngoc Tran (Soongsil University), Sridhar Rao (The Linux Foundation),

Jangwon Lee, Younghan Kim (Soongsil University)

Scope Re-introduction

- Previous **NFV benchmarking** related RFCs
  - RFC 8172: Considerations for Benchmarking Virtual Network Functions and Their Infrastructure
  - RFC 8204: Benchmarking Virtual Switches in the Open Platform for NFV (OPNFV)

- The primary scope of this document is to fill in **the gaps** of these works

when applying to containerized NFV infrastructure.

- **The consideration gaps are:**
  - Different **network models/topologies configured by Container Network Interfaces** (including the extended Berkeley Packet Filter model which was not mentioned in previous documents)

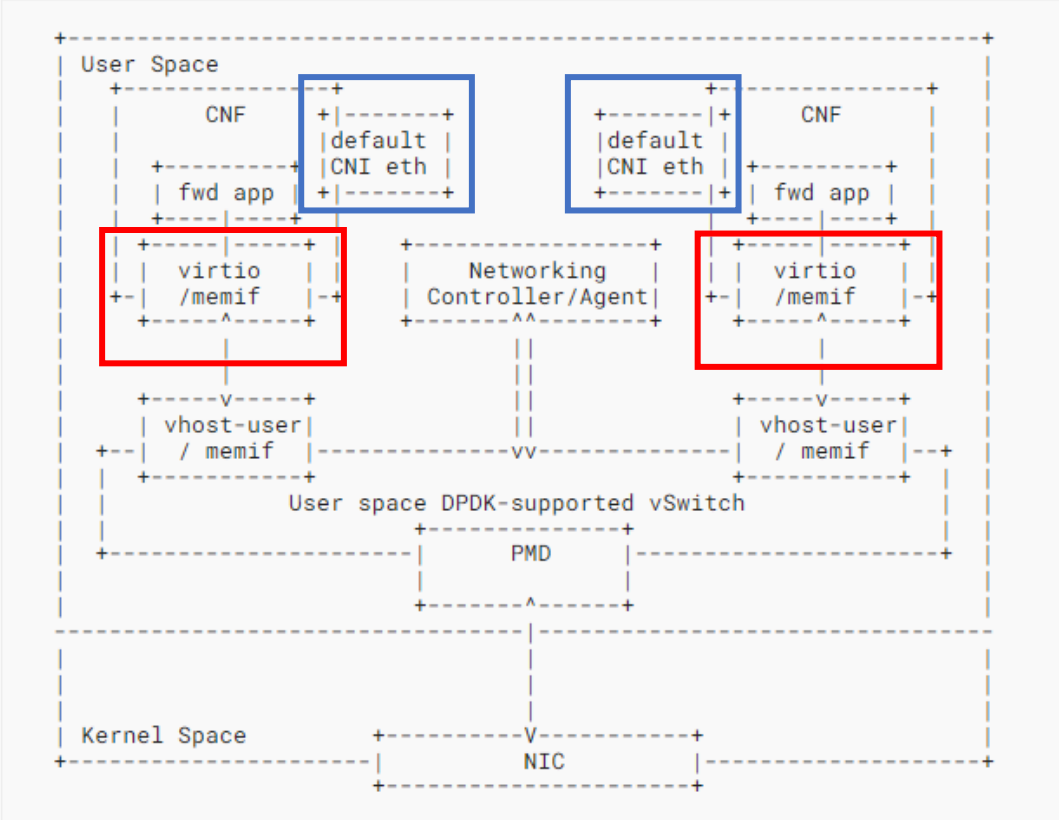
  - **Resources configuration for containers.**

Updates (1)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Modify figures to show hybrid container networking stack**
  - **Enable by Multus CNI**
  - **One CNF network interface is created by the default CNI – is Used for pod management, control plane traffic**
  - **One CNF network interface is created by the accelerated CNI – is Used for Accelerated User Application Traffic (Benchmark Target)**

5.1.2.1. User-space Acceleration Model



Updates (1)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Modify figures to show hybrid container networking stack**
  - This change is applied for all Accelerated Networking Model
  - Add sentences to describe this difference between non-Acceleration Model and Acceleration Model

Non Acceleration

- **Single CNI** for IPAM and user application traffic

Acceleration

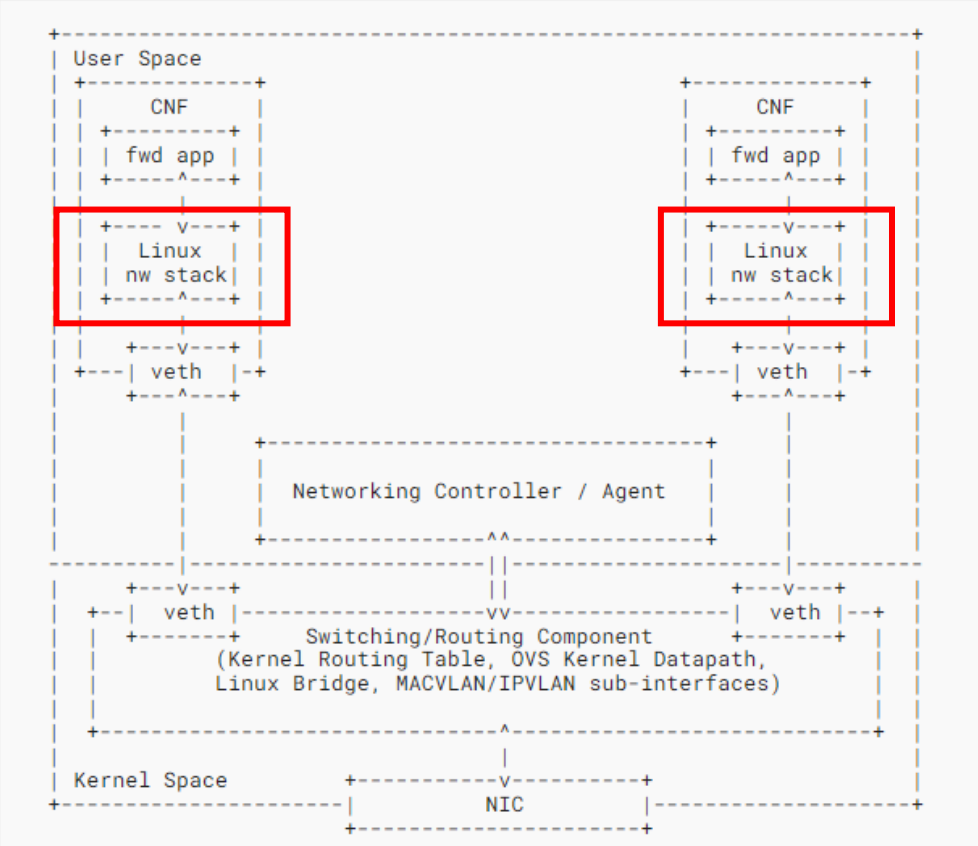
- **One default CNI** for IPAM and management
- **At least one Accelerated CNI** for user application traffic
- Benchmarking target can be default/accelerated CNI

Updates (2)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Modify figure to show the CNF separated networking stack in Normal non-Acceleration networking model**
  - CNF has a separate network namespace from the host and has a separated networking stack
  - Is only applied for Normal Non-Acceleration networking model
  - In Accelerated Networking model, datapath is optimally configured to reach the user application via accelerated technology library

5.1.1. Normal non-Acceleration Networking Model



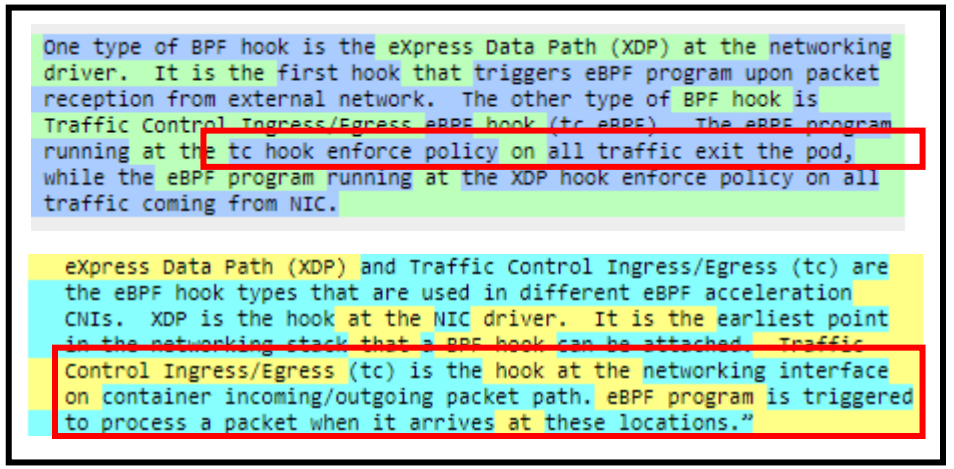
Updates (3)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Changing word choices to correctly describe eBPF, AF_XDP technology**

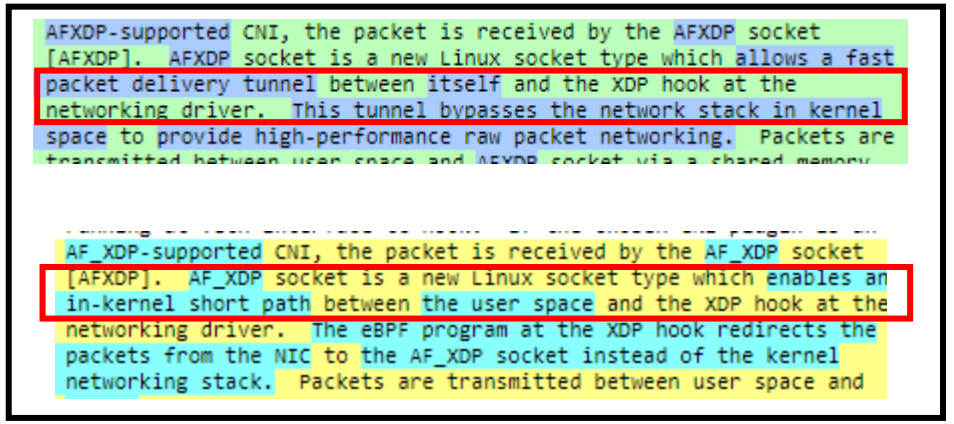
  - Instead of eBPF at the traffic control hook “enforce policy”
    -> eBPF is triggered to process packet when it arrives at the traffic control hook

  - Instead of AF_XDP create a tunnel bypass networking stack
    -> AF_XDP enables an in_kernel short path



One type of BPF hook is the eXpress Data Path (XDP) at the networking driver. It is the first hook that triggers eBPF program upon packet reception from external network. The other type of BPF hook is Traffic Control Ingress/Egress eBPF hook (tc eBPF). The eBPF program running at the tc hook enforce policy on all traffic exit the pod, while the eBPF program running at the XDP hook enforce policy on all traffic coming from NIC.

eXpress Data Path (XDP) and Traffic Control Ingress/Egress (tc) are the eBPF hook types that are used in different eBPF acceleration CNIs. XDP is the hook at the NIC driver. It is the earliest point in the networking stack that a BPF hook can be attached. Traffic Control Ingress/Egress (tc) is the hook at the networking interface on container incoming/outgoing packet path. eBPF program is triggered to process a packet when it arrives at these locations.”



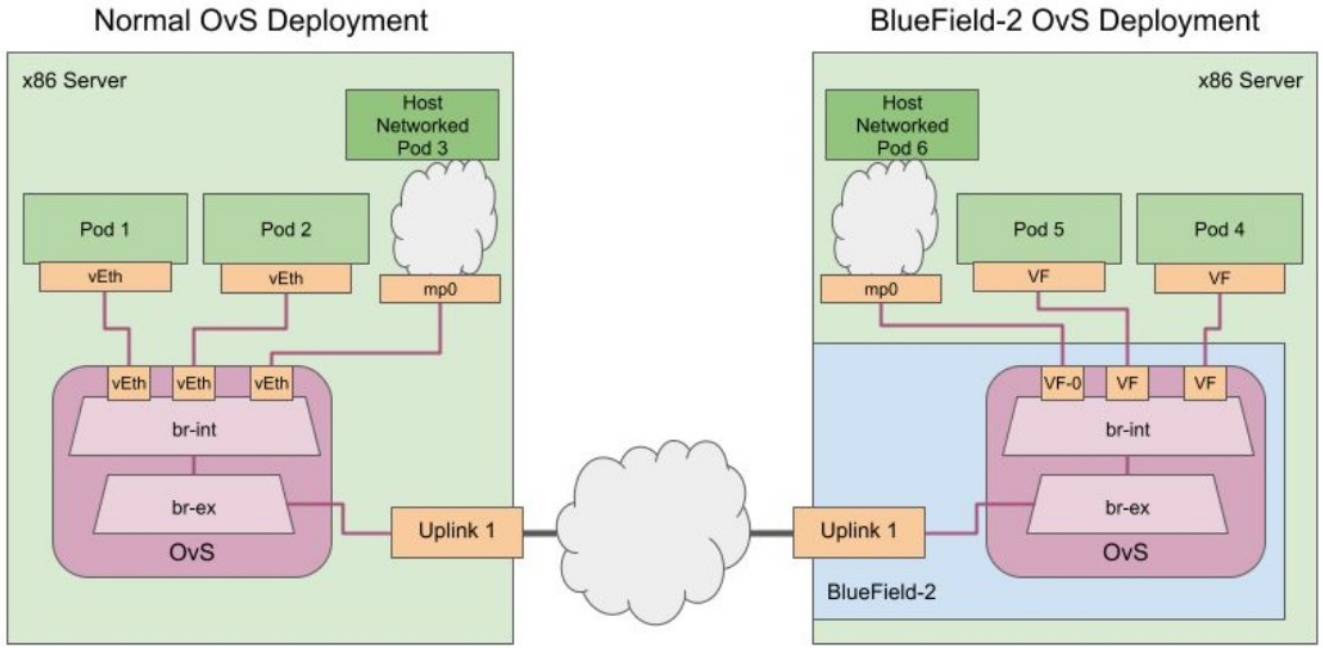
AFXDP-supported CNI, the packet is received by the AFXDP socket [AFXDP]. AFXDP socket is a new Linux socket type which allows a fast packet delivery tunnel between itself and the XDP hook at the networking driver. This tunnel bypasses the network stack in kernel space to provide high-performance raw packet networking. Packets are transmitted between user space and AFXDP socket via a shared memory

AF_XDP-supported CNI, the packet is received by the AF_XDP socket [AFXDP]. AF_XDP socket is a new Linux socket type which enables an in-kernel short path between the user space and the XDP hook at the networking driver. The eBPF program at the XDP hook redirects the packets from the NIC to the AF_XDP socket instead of the kernel networking stack. Packets are transmitted between user space and

Updates (4)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Add sentences to mention about container network acceleration via xPU networking devices in "Smart NIC Acceleration Model"**
  - xPUs are "any" Processing Unit networking devices that can contains Smart NIC, their own CPU cores and acceleration engines
    - E.g Data Processing Unit (DPU), Infrastructure Processing Unit (IPU)
  - xPUs are new acceleration technology option and has not been widely implemented or documented.
  - Mention in this document as useful additional notice about future technology



Example illustration of container network using an NVIDIA Bluefield DPU

From presentation material *"Accelerating Kubernetes Hybrid Clouds with BlueField DPUs and OpenShift for Ultimate Security and Efficiency"* NVIDIA, Red Hat March 2022

Updates (5)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **Add AF_XDP Configuration as a new Resource Configuration benchmarking consideration**
  - AF_XDP can operate in busy/non-busy polling mode
  - In busy mode, the same CPU core is used for both application and packet processing
  - In non-busy mode, different CPU cores can be allocated for these two tasks
- AF_XDP polling mode, and CPU core allocation in non-busy mode are additional AF_XDP configuration parameters for benchmarking consideration

Updates (6)

We received reviews from the related RFC 8204 author – Maryam Tahhan

- **The reviewer agreed with our explanation to not change about**

- In eBPF AF_XDP model, Adding a datapath from XDP hook to the networking stack.
- eBPF at XDP hook of NIC driver can not only forward packet to userspace via AF_XDP socket, but also different options such as to normal networking stack / drop / back to the original interface.
  - We do not add it because this document focuses and highlights the accelerated datapath only

- Receive Side Scaling configuration at NIC can assign multiple packet processing queues to a network interface and affect benchmarking performance
  - This configuration is not unique in containter environment

Document Review History

Before Adoption

- **Version 10 – March 2023**
  - **Review from Linux Foundation VinePerf Project (Sridhar and Al Morton)**
  - Agreed on Container Networking Models and Resource Configuration categorization
- **Version 11 – July 2023**
  - **Review from BMWG members (Gábor and Vratko)**
  - Added Benchmarking parameter for each Resource Configuration consideration
- **Version 13 – November 2023**
  - **Comments at IETF 117 meeting**
  - Provided clear Scope section and remove duplicate introduction information

After Adoption

- **Version 0 – March 2024**
  - **Reviews from BMWG members in WG adoption call comments**
  - Added Environment Setup repeatability guidance for each Containter Networking Models
- **Version 3 – October 2024**
  - **Review from Related RFC 8204 document - Maryam**
  - Revise Hybrid networking stack information for Accelerated networking models, added xPU acceleration, added AF_XDP resources consideration

Summary

- We addressed all comments from the related RFC 8204 author Maryam Tahhan about this document
- All changes was updated to the new version of the document

- We would like to request Last Call.