Towards Trusted Container Networking: Physical Network Segmentation by Hardware-assisted Secure Bridge

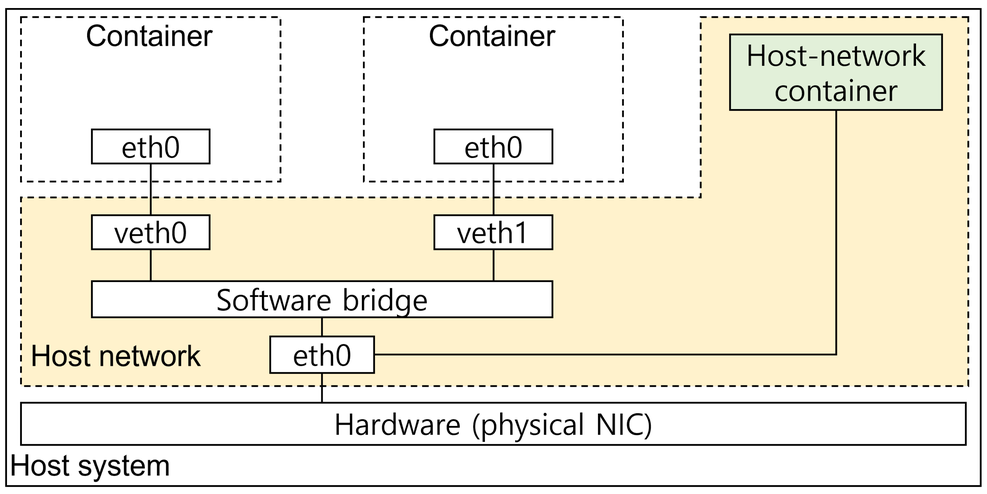
Myoungsung You, Seungwon Shin
Korea Advanced Institute of Science and Technology

Introduction

- In the cloud environment, most services are operated in the form of a microservice.
- A microservice consists of **several containers connected through a network.**
- Containers communicate through **a network interface implemented with virtual network devices of a host system** (e.g., Linux bridge, a software switch).
- This approach effectively enables inter-container networking, but it denotes that **the container network is still tightly coupled with the host system.**
- The non-isolated container **network inevitably exposes the inter-container traffic to the host system.**

Background and Motivation

- Container networking architecture
  - Containers have their own network namespace (i.e., networking stacks) and are separated from the host network.
  - Some containers called **host-network containers share a network namespace with the host kernel**, **allowing them to access the host system's network resources.**



- Container network security solutions
  - Traffic inspection
    - Restrict network flow between containers according to security policies
    - Cilium, Calico, Bastion (USENIX ATC 19)
  - Traffic encryption
    - Encrypt the container traffic by using mTLS
    - Istio, Linkerd
    - **Low performance (traffic encryption)**

- Limitations of existing solutions
  - Even if well-defined security policies exist, **existing solutions cannot mitigate network attacks from compromised host-network containers**.
    - **Container Traffic Exposure**
      - Host-network containers, which reside in the host network namespace, naturally have full visibility of inter-container traffic
    - **Lack of visibility into spoofed packets**
      - Containers can inject spoofed packets directly into the network interfaces of other containers (e.g., veth0, veth1), bypassing the inspection of traditional solutions.

Challenge and Approach

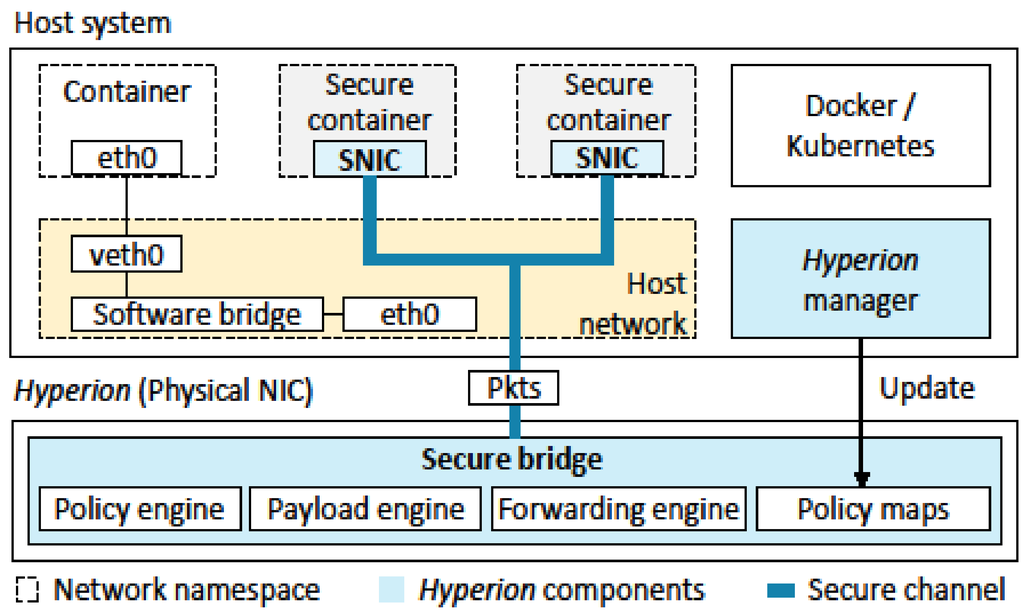
- How to provide an isolated inter-container communication channel from the host network namespace without compromising networking performance?
  - **Offload the existing software bridge to hardware**
  - Enable containers to **communicate directly through a hardware bridge**

- How to ensure the reliability of inter-container communication in the isolated channel?
  - Inspect the traffic of all containers on the hardware bridge
  - Restrict the container from sending spoofed packets on the container-side

Design

- Hyperion: A novel hardware-assisted security extension for container networks.
  - **The Secure bridge**: A hardware-offloaded networking bridge with embedded security engines.
  - **SNICs**: Secure network interfaces directly connected to the secure bridge through SR-IOV.

- All communications between secure containers (containers with SNICs) are supervised through Hyperion's security inspectors(Blue Box) rather than the existing bridge network.



Evaluation

- Security evaluations
  - The host network namespace has no network interfaces that are directly connected to secure containers.

Before the SNIC installation

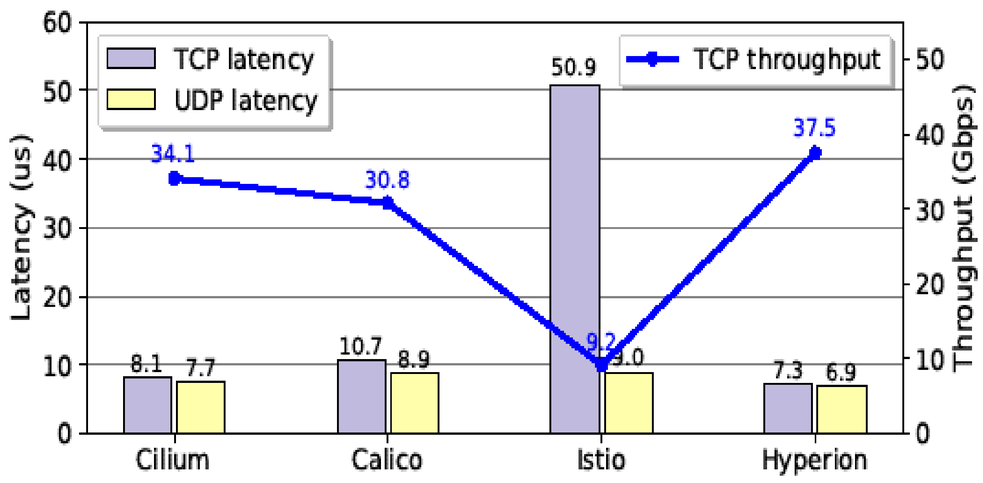
```
root@flask-b:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
37: eth0@if38: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue s
```

After the SNIC installation

```
root@flask-b:/# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
42: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mo

32: lxc356dc3fb9894@if31: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
    link/ether 7a:58:60:aa:cb:81 brd ff:ff:ff:ff:ff:ff link-netnsid 6
34: lxc7fb05ea132a4@if33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
    link/ether f6:2c:02:ca:b3:ad brd ff:ff:ff:ff:ff:ff link-netnsid 7
36: lxccf4a2d50a1fa@if35: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
    link/ether 9a:67:6c:7f:2b:b4 brd ff:ff:ff:ff:ff:ff link-netnsid 8
46: lxc7fc6596f6aab@if45: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
```

- Performance evaluations
  - Hyperion outperforms state of the arts solutions by up to five times.



Myoungsung You
KAIST
Email: famous77@kaist.ac.kr
Website: nss.kaist.ac.kr