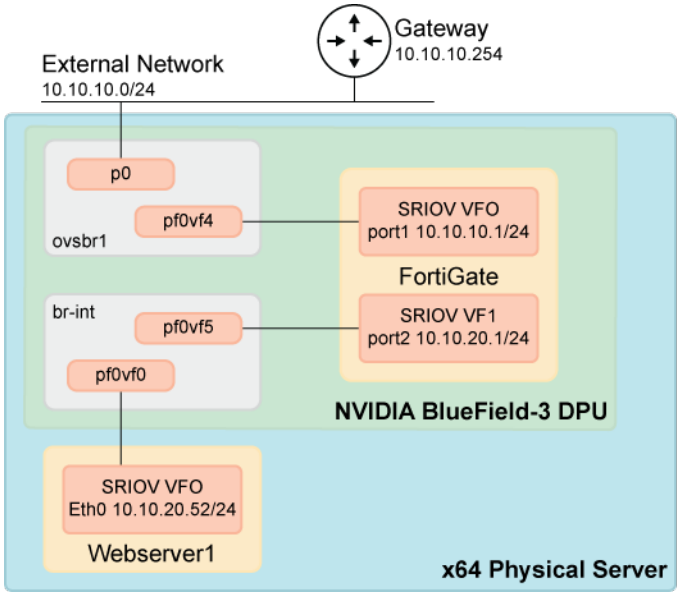
FortiGate-VM on NVIDIA BlueField-3

This guide provides the steps to deploy a FortiGate-VM on an NVIDIA BlueField-3 (BF3) DPU in an x64 host server (Host). The FortiGate runs directly on the BF3 using an ARM64 image and secure outbound Internet access and inbound access for workloads using a VIP.

Following this guide, you will:

- Deploy a FortiGate with external (WAN) and internal (LAN) connectivity.
- Provide Internet access to virtualized workloads running on the Host.
- Enable secure inbound access and inspection for workloads through the FortiGate.

Architecture overview



The deployment uses two primary Open vSwitch (OVS) bridges on the BF3:

1. `ovsbr1` - External Network Bridge (WAN):
   - Pre-created by BF3 firmware.
   - Connects FortiGate `port1` VF representor to the physical network via the uplink representor `p0`.
2. `br-int` - Internal Network Bridge (LAN):
   - Created manually by the user.
   - Provides internal connectivity by bridging FortiGate `port2` VF representor with the Host workload VF representors.

Refer to the DPU Kernel Representors Model documentation for more information.

Prerequisites

On the BF3

- OVS is pre-installed and pre-configured.
- `ovsbr1` and `ovsbr2` bridges are pre-created by the BF3 firmware.
- The required virtualization tools are installed:

```
apt-get update
apt-get install -y qemu-kvm libvirt-daemon-system libvirt-clients virtinst
```

On the Host

- SRIOV is enabled on the Host.
- The required virtualization and OVS tools are installed:

```
apt-get update
apt-get install -y qemu-kvm libvirt-daemon-system libvirt-clients virtinst
```

Install DOCA-Host

On the Host, install the DOCA-Host package. This software enables the management and configuration of features of the BF3.

See the NVIDIA DOCA-Host Installation Guide for detailed platform-specific steps.

Enable SR-IOV on BF3 and Host

Enable SR-IOV and set the number of Virtual Functions (VFs). In this example, we configure four VFs per port.

BF3 Configuration:

1. Determine the PCI device addresses:

```
# lspci | grep Mellanox
00:00.0 PCI bridge: Mellanox Technologies MT43244 BlueField-3 SoC Crypto enabled (rev 01)
03:00.0 Ethernet controller: Mellanox Technologies MT43244 BlueField-3 integrated ConnectX-7
network controller (rev 01)
03:00.1 Ethernet controller: Mellanox Technologies MT43244 BlueField-3 integrated ConnectX-7
network controller (rev 01)
```

2. Enable SR-IOV and set the PF_NUM_OF_VF value to 4:

```
# mst start
# mlxconfig -d 03:00.0 -y s PF_NUM_OF_VF_VALID=1
# mlxconfig -d 03:00.0 -y s PF_NUM_OF_VF=4
# mlxconfig -d 03:00.1 -y s PF_NUM_OF_VF=4
```

Host Configuration:

Enable SR-IOV according to the host OS installation guide and set the number of VFs. In this example, we create four VFs per port. Host based workloads will use these VFs for connectivity.

1. Determine the PCI device addresses of the Host ports:

```
# lspci | grep Mellanox
b5:00.0 Ethernet controller: Mellanox Technologies MT43244 BlueField-3 integrated ConnectX-7
network controller (rev 01)
b5:00.1 Ethernet controller: Mellanox Technologies MT43244 BlueField-3 integrated ConnectX-7
network controller (rev 01)
b5:00.2 DMA controller: Mellanox Technologies MT43244 BlueField-3 SoC Management Interface
(rev 01)
```

2. Set the PF_NUM_OF_VF value to 4:

```
# mlxconfig -d b5:00.0 -y s PF_NUM_OF_VF=4
# mlxconfig -d b5:00.1 -y s PF_NUM_OF_VF=4
```

Perform a BF3 system reboot to apply changes by following the NVIDIA BlueField Reset and Reboot Procedures guide.

Instantiate Virtual Functions on BF3 and Host

After the reboot, instantiate the VFs on both the BF3 and the Host. In this example, we instantiate two VFs per port on BF3 and four VFs per port on the Host.

BF3 Ports

```
# echo 2 > /sys/class/net/p0/device/sriov_numvfs
# echo 2 > /sys/class/net/p1/device/sriov_numvfs
```

```
# lspci | grep "Virtual Function"
03:00.2 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
03:00.3 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
03:00.6 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
03:00.7 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
```

The first two VFs associated with port p0 (PCI addresses 03:00.2 and 03:00.3) will be used for port1 and port2 of the FortiGate in this setup.

HOST Ports

```
# echo 4 > /sys/bus/pci/devices/0000:b5:00.0/sriov_numvfs
# echo 4 > /sys/bus/pci/devices/0000:b5:00.1/sriov_numvfs
```

```
# lspci | grep "Virtual Function"
b5:00.3 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:00.4 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:00.5 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:00.6 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:00.7 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:01.0 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:01.1 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
b5:01.2 Ethernet controller: Mellanox Technologies ConnectX Family mlx5Gen Virtual Function (rev
01)
```

These VFs are dedicated for host-side workloads. In this deployment example, Webserver1 uses one of the four VFs associated to p0. The PCI address is b5:00.3 and it maps to the VF Representor pf0vf0 on the BF3 in this setup.

Deploy the FortiGate

Use FortiGate ARM64 images, as BF3 uses ARM processors. For example: FGT_ARM64_KVM-v7.6.3.F-build3510-FORTINET.out.kvm.zip

Prepare FortiGate images:

1. Create a deployment folder:

   ```
   # mkdir /home/ubuntu/fg1
   # cd /home/ubuntu/fg1
   ```

2. Create a 30GB raw disk for logs:

   ```
   # qemu-img create -f raw /home/ubuntu/fg1/fg1.raw -o size=30G
   ```

3. Create a QCOW2 overlay to protect the base FortiOS image:

```
# qemu-img create -f qcow2 -F qcow2 -b /home/ubuntu/fg1/fortios.qcow2 /home/ubuntu/fg1/fg1-overlay.qcow2
```

Launch the FortiGate with two interfaces by assigning the VFs as network interfaces using PCI passthrough:

```
# virt-install \
  --name fg1 \
  --ram 16384 \
  --vcpus 8 \
  --virt-type kvm \
  --arch aarch64 \
  --machine virt \
  --import \
  --disk path=/home/ubuntu/fg1/fg1-overlay.qcow2,format=qcow2,bus=virtio \
  --os-variant generic \
  --boot loader=/usr/share/AAVMF/AAVMF_CODE.fd,loader.readonly=yes,loader.type=pflash \
  --graphics none \
  --console pty,target_type=serial \
  --network none \
  --hostdev pci_0000_03_00_2 \
  --hostdev pci_0000_03_00_3 \
  --noreboot
```

Attach the Logdisk to the FortiGate instance:

```
# virsh attach-disk fg1 --source /home/ubuntu/fg1/fg1.raw --target vdb --persistent
```

Power up the FortiGate:

```
# sudo virsh start fg1
```

Access the FortiGate console and change the default password:

```
# sudo virsh console fg1
```

```
Default username: admin
Default password: <No password>
```

Network Setup

External Connectivity (WAN)

Connect FortiGate port1 VF representor to the ovsbr1 bridge.

```
# ovs-vsctl add-port ovsbr1 pf0vf4
```

Verify ovsbr1 settings on BF3:

```
# ovs-vsctl show
```

Expected output:

```
Bridge ovsbr1
    Port p0
        Interface p0
    Port pf0vf4
        Interface pf0vf4
    Port ovsbr1
        Interface ovsbr1
            type: internal
```

Internal Connectivity (LAN)

Create a bridge (br-int) on the BF3 to facilitate internal communication between the Host workloads and the FortiGate, then attach the corresponding VF representors to this bridge:

```
# ovs-vsctl add-br br-int
```

Attach FGT port2 VF representor:

```
# ovs-vsctl add-port br-int pf0vf5
```

Attach Webserver1 VF representor:

```
# ovs-vsctl add-port br-int pf0vf0
```

Verify br-int settings on BF3:

```
# ovs-vsctl show
```

Expected output:

```
Bridge br-int
    Port pf0vf0
        Interface pf0vf0
    Port pf0vf5
        Interface pf0vf5
    Port br-int
        Interface br-int
            type: internal
```

FortiGate Configuration

Configure Port1 (WAN)

```
config system interface
    edit port1
        set mode static
        set ip 10.10.10.1 255.255.255.0
        set allowaccess https ssh ping
    next
end
```

Set the default route:

In this example, 10.10.10.254 is the default gateway.

```
config router static
    edit 1
        set gateway 10.10.10.254
        set device port1
    next
end
```

Verify connectivity:

```
# execute ping fortinet.com
```

Configure FortiGate port2 (LAN)

```
config system interface
    edit port2
        set mode static
```

```
        set ip 10.10.20.1 255.255.255.0
        set allowaccess https ssh ping
    next
end
```

Test connectivity to the workload:

In this example, the Webserver1 IP is 10.10.20.52:

```
# ping 10.10.20.52
```

License the FortiGate

Access the FortiGate GUI at `https://10.10.10.1` and upload the FortiGate license.

Create firewall policy for workload Internet access

Configure workload to use 10.10.20.1 (FortiGate port2 IP address) as its default gateway.

```
config firewall policy
    edit 1
        set name "Protect-Internet"
        set srcintf "port2"
        set dstintf "port1"
        set action accept
        set srcaddr "all"
        set dstaddr "all"
        set schedule "always"
        set service "ALL"
        set utm-status enable
        set ssl-ssh-profile "certificate-inspection"
        set av-profile "default"
        set ips-sensor "default"
        set nat enable
        set logtraffic all
    next
end
```

Configure a VIP for inbound access

In this example, the external clients access 10.10.10.2, which is translated to the Webserver1 IP address 10.10.20.52.

Create the VIP:

```
config firewall vip
    edit "Webserver1"
        set extip 10.10.10.2
        set mappedip "10.10.20.52"
        set extintf "port1"
    next
end
```

Create a policy for the VIP traffic:

```
config firewall policy
    edit 2
        set name "Webserver-Access"
        set srcintf "port1"
        set dstintf "port2"
        set srcaddr "all"
        set dstaddr "Webserver1"
        set action accept
        set service "ALL"
        set utm-status enable
        set ssl-ssh-profile "certificate-inspection"
        set av-profile "default"
        set ips-sensor "default"
        set nat enable
    next
end
```

Verification

Internet access from workload

From a workload VM:

```
# ping fortinet.com
```

On FortiGate:

```
# diagnose sniffer packet any 'host <fortinet.com resolved IP>' 4 0 l
```

External access to Webserver1

From the external network:

```
# ping 10.10.10.2
```

Access http://10.10.10.2 and you should be able to see the Webserver1's landing page.