Leveraging virtio-vsock in the cloud and containers

FOSDEM 2021 Virtual Conference
6th of February 2021

Andra Paraschiv, Amazon Web Services
Stefano Garzarella, Red Hat

aws

Red Hat

Agenda

Introduction

Use Cases

- AWS Nitro Enclaves
- Kata Containers
- libkrun

Developing with AF_VSOCK

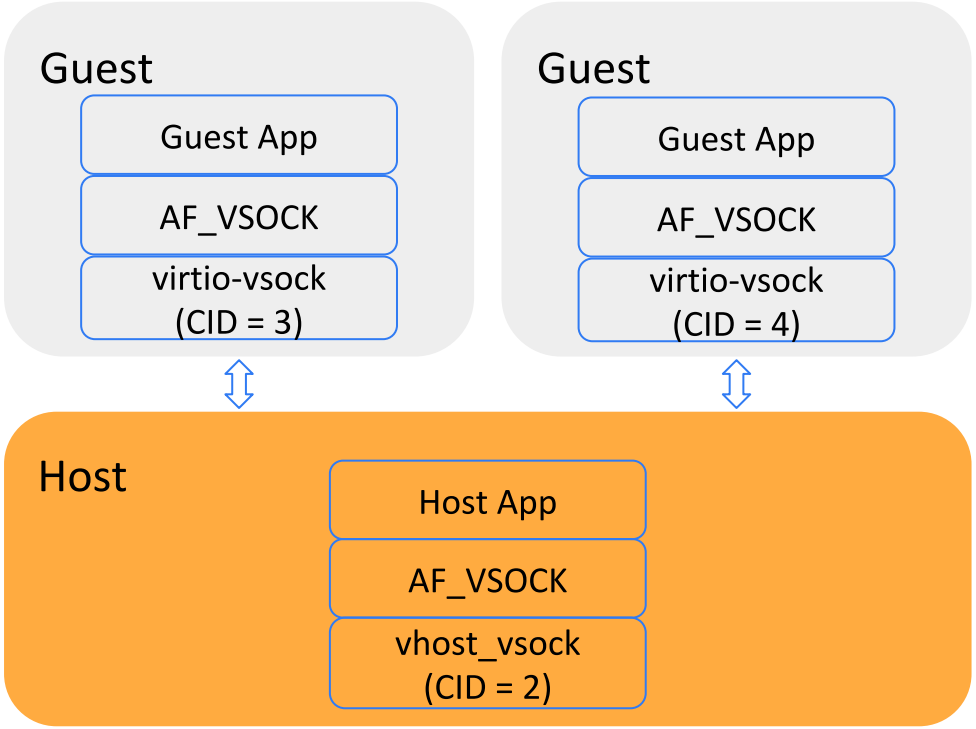
- Languages Bindings
- Communication Debugging
- Performance Evaluation

Next Steps

- SOCK_SEQPACKET
- Namespaces
- Multi-queue
- Shared Memory

Vsock - Overview

- POSIX Socket API (e.g. socket, bind, listen, connect)
- SOCK_STREAM / SOCK_DGRAM
- CID (Context Identifier) and port
  - Well-known CIDs
    - VMADDR_CID_ANY (0xFFFFFFFF)
    - VMADDR_CID_HYPERVISOR (0)
    - VMADDR_CID_LOCAL (1)
    - VMADDR_CID_HOST (2)
  - Privileged ports (port < 1024)
- Multi transports - support for nested VMs



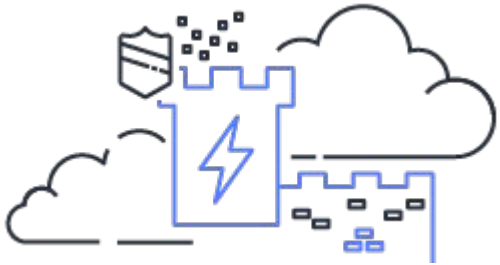
Vsock - Use Cases

Network Applications
(e.g. SOCK_STREAM apps)

Guest Agents
(e.g. QEMU guest agent, Kata containers agent)

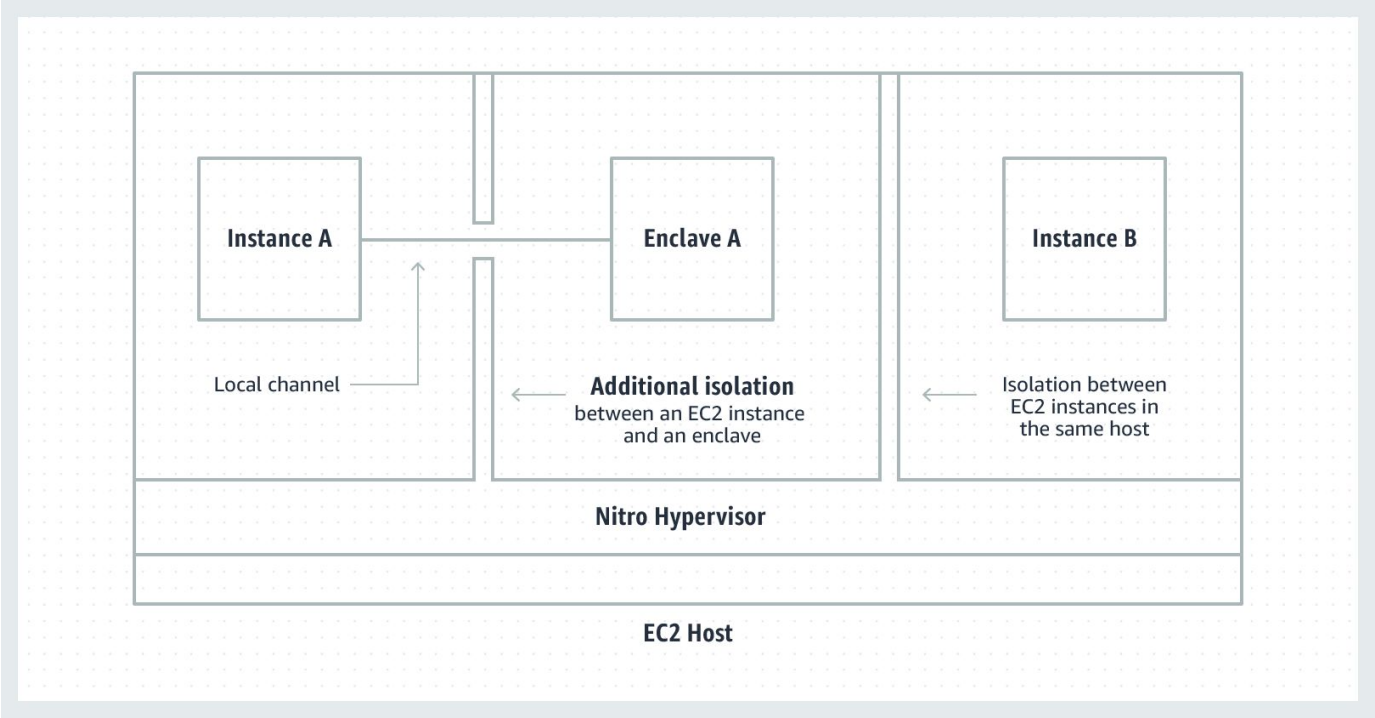
Hypervisor Services
(e.g. file sharing)

AWS Nitro Enclaves



- Isolated and trusted environment
- Cryptographic attestation
- Open source
- Portable across operating systems and architectures
- Integrated with AWS Key Management Service (KMS) and AWS Certificate Manager (ACM)

AWS Nitro Enclaves (2)



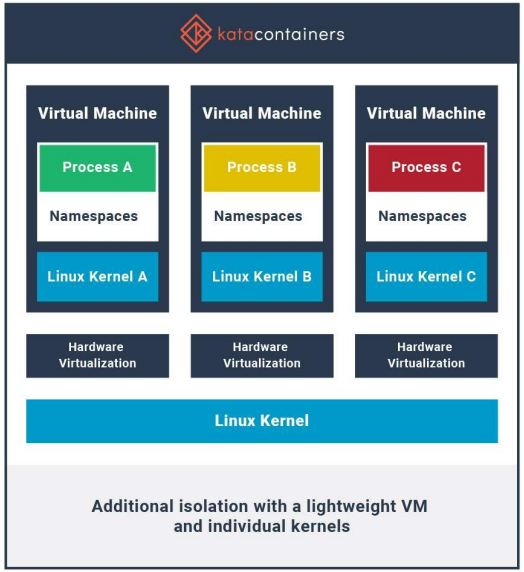
AWS Nitro Enclaves - Serial Console Access (Demo)

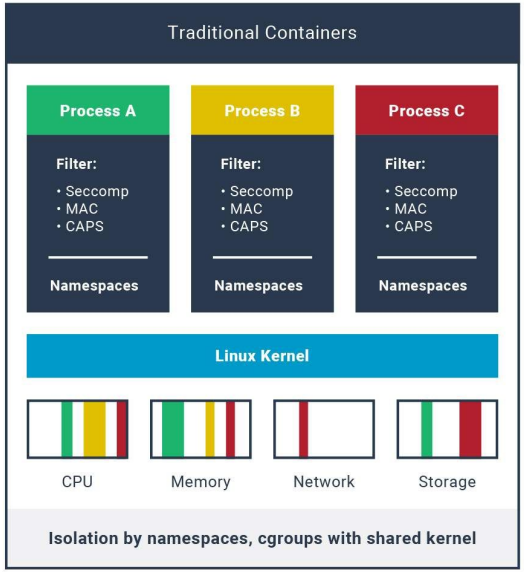
```
[ec2-user@vsock-demo ~]$ lsmod | grep nitro_enclaves
nitro_enclaves         32768  0
[ec2-user@vsock-demo ~]$ sudo lspci -v -s 00:02.0
00:02.0 Communication controller: Amazon.com, Inc. Device e4c1 (rev 01)
        Subsystem: Amazon.com, Inc. Device e4c1
        Physical Slot: 2
        Flags: fast devsel
        Memory at febf0000 (32-bit, non-prefetchable) [size=4K]
        Memory at febf1000 (32-bit, non-prefetchable) [size=4K]
        Capabilities: [40] MSI-X: Enable+ Count=3 Masked-
        Kernel driver in use: nitro_enclaves
        Kernel modules: nitro_enclaves

[ec2-user@vsock-demo ~]$ sudo lspci -v -s 00:02.1
00:02.1 Communication controller: Red Hat, Inc. Device 1053 (rev 01)
        Subsystem: Red Hat, Inc. Device 1053
        Physical Slot: 2
        Flags: fast devsel
        [virtual] Memory at febf2000 (32-bit, non-prefetchable) [size=4K]
        [virtual] Memory at febe0000 (32-bit, non-prefetchable) [size=32K]
        Capabilities: [90] Vendor Specific Information: VirtIO: DeviceCfg
        Capabilities: [80] Vendor Specific Information: VirtIO: ISR
        Capabilities: [70] Vendor Specific Information: VirtIO: <unknown>
        Capabilities: [5c] Vendor Specific Information: VirtIO: Notify
        Capabilities: [4c] Vendor Specific Information: VirtIO: CommonCfg
        Capabilities: [40] MSI-X: Enable+ Count=4 Masked-
        Kernel driver in use: virtio-pci
        Kernel modules: virtio_pci

[ec2-user@vsock-demo ~]$ cat /sys/devices/system/node/node0/hugepages/hugepages-
```

AWS Nitro Enclaves - Vsock Sample (Demo)

Kata Containers

- **secure** container runtime
- **lightweight** virtual machines
  - feel and perform like containers
- stronger workload **isolation**
  - hardware virtualization technology as a second layer of defense



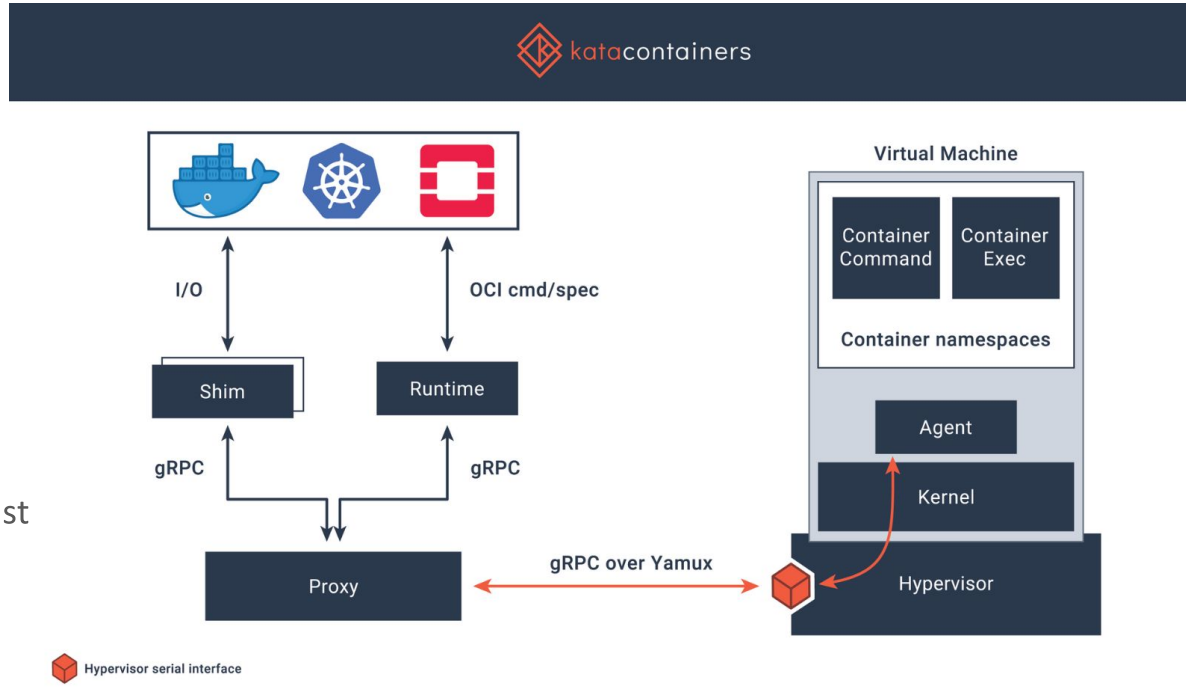


https://katacontainers.io

Red Hat

Kata Containers using serial ports

- guest processes
  - serial port device
- host processes
  - Unix socket
- Portable solution
- Cons
  - one process at a time can access serial link
  - serial port and Unix socket must be multiplexed by using kata-proxy and Yamux

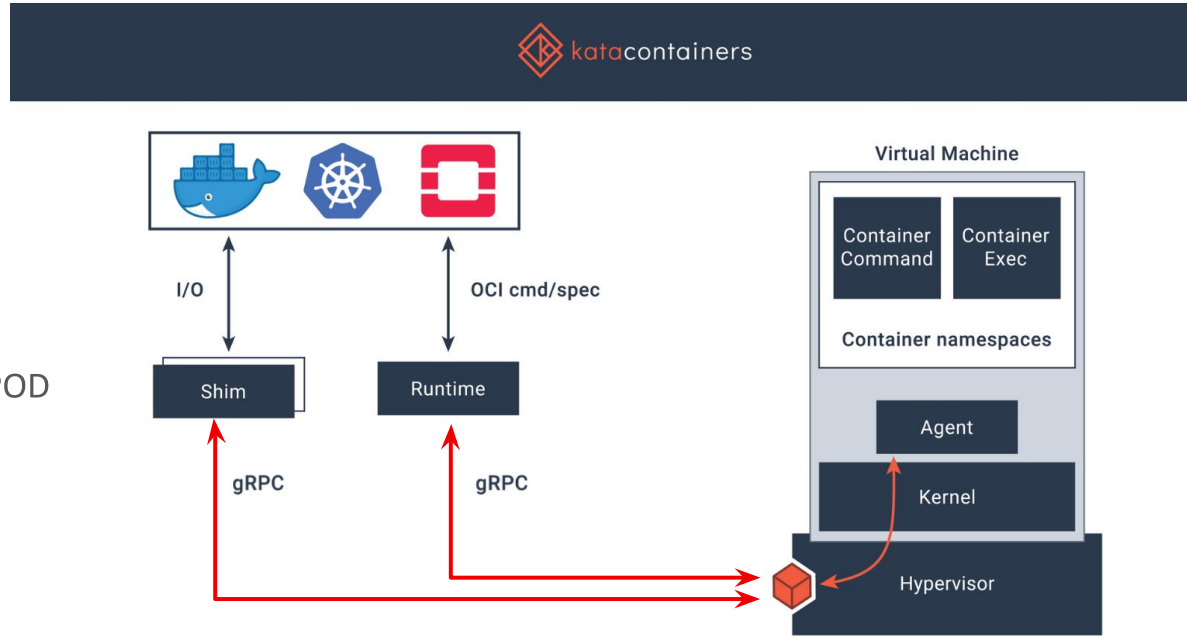


https://katacontainers.io/learn

Red Hat

Kata Containers using VSOCK

- Listen sockets can accept connections from multiple clients
  - does not require multiplexers (kata-proxy and Yamux)
- High density
  - kata- proxy uses ~4.5MB per POD
- Reliability
  - if kata-proxy dies all POD connections get broken



virtio-vsock device

https://github.com/kata-containers/documentation/blob/master/design/VSocks.md

Red Hat

Kata Containers demo

```
1: @ada0d224b72c:/ 63x15
00:1f.6 Ethernet controller: Intel Corporation Ethernet Connect
ion (4) I219-LM (rev 21)
04:00.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
05:00.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
05:01.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
05:02.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
06:00.0 System peripheral: Intel Corporation JHL6240 Thunderbol
t 3 NHI (Low Power) [Alpine Ridge LP 2016] (rev 01)
07:00.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
08:00.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
```

```
2: ./demo-host2.sh 63x15
05:00.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
05:01.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
05:02.0 PCI bridge: Intel Corporation JHL6240 Thunderbolt 3 Bri
dge (Low Power) [Alpine Ridge LP 2016] (rev 01)
06:00.0 System peripheral: Intel Corporation JHL6240 Thunderbol
t 3 NHI (Low Power) [Alpine Ridge LP 2016] (rev 01)
07:00.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
08:00.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
08:01.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
08:02.0 PCI bridge: Intel Corporation JHL6540 Thunderbolt 3 Bri
dge (C step) [Alpine Ridge 4C 2016] (rev 02)
```

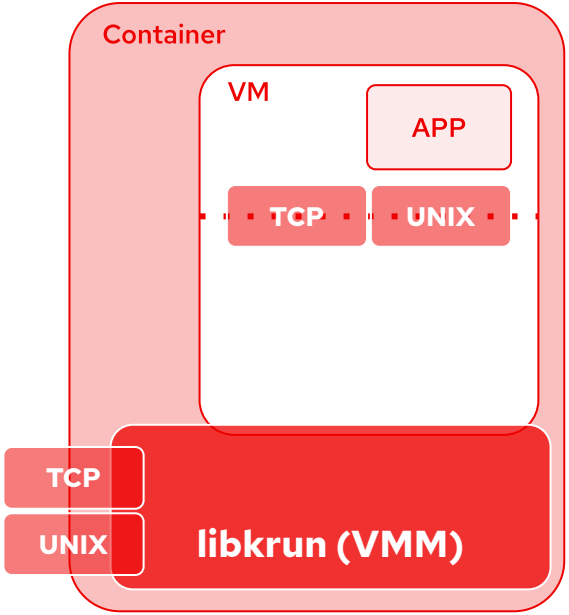
```
3: @781ca166b285:/ 63x15
ls[root@781ca166b285 /]# lspci
00:00.0 Host bridge: Intel Corporation 82G33/G31/P35/P31 Expres
s DRAM Controller
00:01.0 Communication controller: Red Hat, Inc. Virtio console
00:02.0 PCI bridge: Red Hat, Inc. QEMU PCI-PCI bridge
00:03.0 SCSI storage controller: Red Hat, Inc. Virtio SCSI
00:04.0 Unclassified device [00ff]: Red Hat, Inc. Virtio RNG
00:05.0 Communication controller: Red Hat, Inc. Virtio socket
00:06.0 Mass storage controller: Red Hat, Inc. Device 105a (rev
 01)
00:07.0 Ethernet controller: Red Hat, Inc. Virtio network devic
e
00:1f.0 ISA bridge: Intel Corporation 82801IB (ICH9) LPC Interf
ace Controller (rev 02)
00:1f.2 SATA controller: Intel Corporation 82801IR/IO/IH (ICH9R
/DO/DH) 6 port SATA Controller [AHCI mode] (rev 02)
```

```
4: ./demo-host4.sh 63x15
har-e84208feddfb6791,tag=kataShared,romfile= -netdev tap,id=net
work-0,vhost=on,vhostfds=4,fds=5 -device driver=virtio-net-pci,
netdev=network-0,mac=5e:92:72:f5:45:3a,disable-modern=false,mq=
on,vectors=4,romfile= -global kvm-pit.lost_tick_policy=discard
-vga none -no-user-config -nodefaults -nographic -daemonize -ob
ject memory-backend-file,id=dimm1,size=2048M,mem-path=/dev/shm,
share=on -numa node,memdev=dimm1 -kernel /usr/lib/modules/5.9.1
6-100.fc32.x86_64/vmlinuz -initrd /var/cache/kata-containers/os
builder-images/5.9.16-100.fc32.x86_64/fedora-kata-5.9.16-100.fc
32.x86_64.initrd -append tsc=reliable no_timer_check rcupdate.r
cu_expedited=1 i8042.direct=1 i8042.dumbkbd=1 i8042.nopnp=1 i80
42.noaux=1 noreplace-smp reboot=k console=hvc0 console=hvc1 iom
mu=off cryptomgr.notests net.ifnames=0 pci=lastbus=0 quiet pani
c=1 nr_cpus=8 agent.use_vsock=true scsi_mod.scan=none -pidfile
/run/vc/vm/781ca166b2859f08190347aaa87be1a2159c077f5ce45834922d
c8732482214e/pid -smp 1,cores=1,threads=1,sockets=8,maxcpus=8
```


Red Hat

libkrun

- **Dynamic library** to run processes in a partially isolated environment
  - HW Virtualization
    - Linux KVM
    - macOS Hypervisor.Framework
  - VMM with a minimum amount of emulated devices
    - virtio-console
    - virtio-fs
    - virtio-vsock
    - virtio-balloon
  - Transparent Socket Impersonation (TSI)

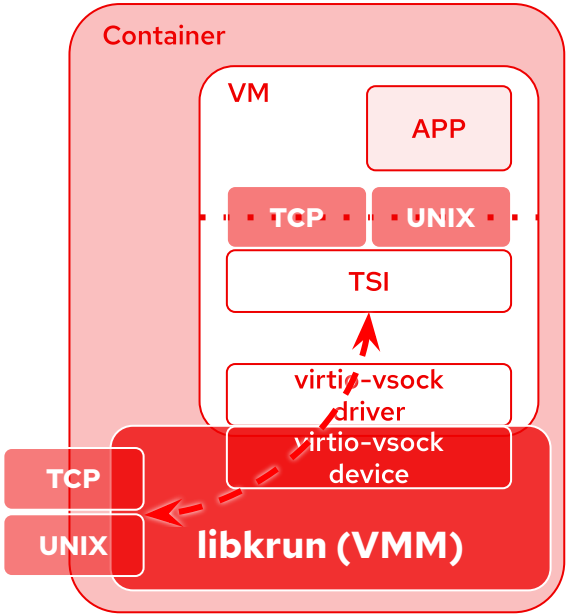


https://github.com/containers/libkrun

Red Hat

libkrun: Transparent Socket Impersonation (TSI)

- VMs have network connectivity without a virtual NIC
- Network connections encapsulated using VSOCK sockets
  - Guest kernel
    - Intercepts sockets syscall
    - Forward requests to the host using virtio-vsock
  - libkrun (VMM in the host)
    - Receives requests through virtio-vsock
    - behaves like the application in the guest
      - making the same sockets syscall in the host
- Fully transparent for application running in the VMs
- Proof of Concept
  - limited to TCP/IPv4 and UNIX domain sockets



https://github.com/containers/libkrunfw

Red Hat

libkrun goals

- Simple API to easily allow other applications to run processes in a VM-based isolated environment
- Small footprint
  - RAM
  - CPU
  - Boot time
- Compatible with a reasonable amount of use cases
  - Stronger isolation for containers
  - Fully encrypted containers (using SEV/TDX)
  - Self-isolating microservices
  - Running Lightweight VMs based on OCI images on Linux and macOS https://twitter.com/slpnix/status/1348406770990002178?s=20

containers

Red Hat

libkrun demo

```
1: @70870d2e0079:/ 63x15
00001000-0009fbff : System RAM
000f0000-000fffff : System ROM
00100000-1fffffff : System RAM
  01000000-01a00bd0 : Kernel code
  01c00000-01d71fff : Kernel rodata
  01e00000-01e9d37f : Kernel data
  0203d000-025fffff : Kernel bss
d0000000-d0000fff : virtio-mmio.0
  d0000000-d0000fff : virtio-mmio.0
d0001000-d0001fff : virtio-mmio.1
  d0001000-d0001fff : virtio-mmio.1
d0002000-d0002fff : virtio-mmio.2
  d0002000-d0002fff : virtio-mmio.2
fee00000-fee00fff : Local APIC
[root@70870d2e0079 /]# ip li
```

```
2: ./demo-host2.sh 63x15
host $
```

```
3: @781ca166b285:/ 63x15
host $
```

```
4: ./demo-host4.sh 63x15
15:02:47.323610 VIRTIO 3319058279.1024 > 2.52193 PAYLOAD, lengt
h 115
15:02:47.324211 VIRTIO 2.52193 > 3319058279.1024 PAYLOAD, lengt
h 93
15:02:47.348178 VIRTIO 3319058279.1024 > 2.52193 PAYLOAD, lengt
h 146
15:02:47.348550 VIRTIO 2.52193 > 3319058279.1024 PAYLOAD, lengt
h 106
15:02:47.348842 VIRTIO 2.52193 > 3319058279.1024 DISCONNECT, le
ngth 76
15:02:47.348979 VIRTIO 3319058279.1024 > 2.52193 DISCONNECT, le
ngth 76
15:02:47.349724 VIRTIO 3319058279.1024 > 2.52193 DISCONNECT, le
ngth 76
```


Red Hat

Developing with AF_VSOCK

Red Hat

Languages providing AF_VSOCK bindings

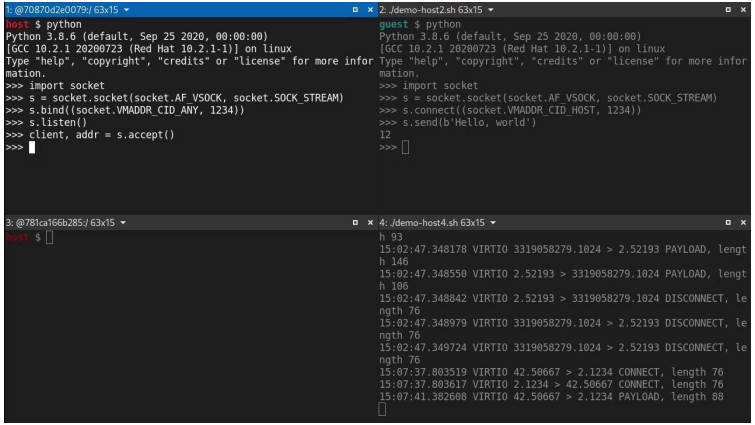
- C
  - glibc >= 2.18 [2013-08-10]
- Python
  - python >= 3.7 alpha 1 [2017-09-19]
- Golang
  - https://github.com/mdlayher/vsock
- Rust
  - libc crate >= 0.2.59 [2019-07-08]
    - struct sockaddr_vm
    - VMADDR_* macros
  - nix crate >= 0.15.0 [2019-08-10]
    - VSOCK supported in the socket API (nix::sys::socket)

Python example

GUEST

```
# Client running in the guest
import socket

s = socket.socket(socket.AF_VSOCK, socket.SOCK_STREAM)
s.connect((socket.VMADDR_CID_HOST, 1234))

s.send(b'Hello, world')
```

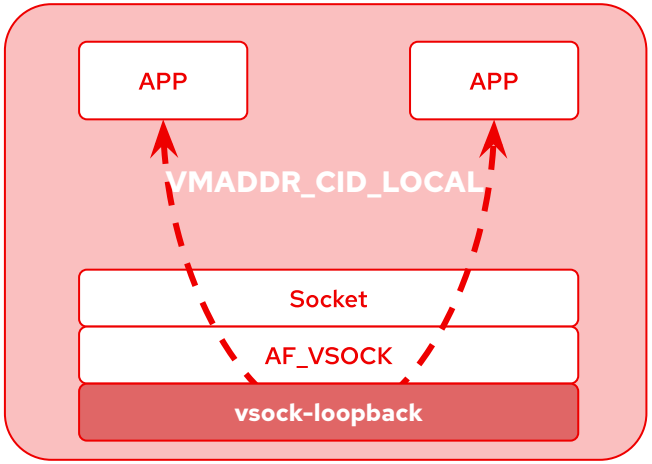
VSOCK



HOST

```
# Server running in the host
import socket

s = socket.socket(socket.AF_VSOCK, socket.SOCK_STREAM)
s.bind((socket.VMADDR_CID_ANY, 1234))
s.listen()
client, addr = s.accept()

data = client.recv(1024)
print("CID: {} port:{} data: {}".format(addr[0], addr[1], data))
```

Local communication with vsock-loopback



- Local communication without VMs
  - Tests
  - Debug
- vsock-loopback
  - New transport available from Linux v5.6
    - `modprobe vsock-loopback`
- CIDs
  - VMADDR_CID_LOCAL (1)
    - Well-known CID for loopback

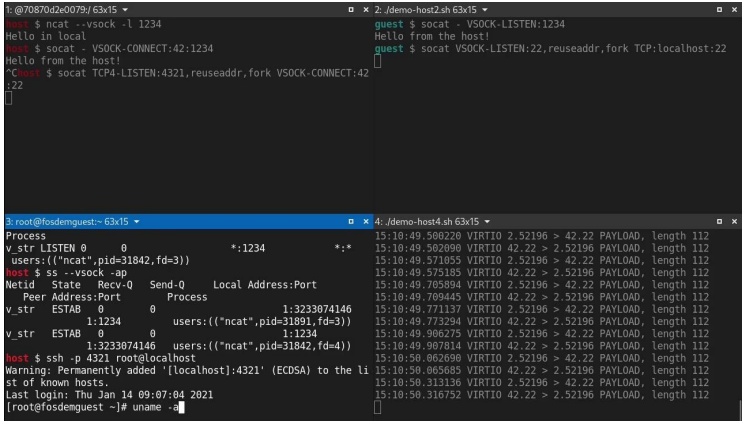
Useful tools with AF_VSOCK support

- **wireshark** >= 2.40 [2017-07-19]
- **iproute2** >= 4.15 [2018-01-28]
  - ss
- **tcpdump** >= 4.99 [2020-12-30]
  - libpcap >= 1.9 [2018-06-24]
  - Fedora backported vsock patches on tcpdump 4.9.3
- **nmap** >= 7.80 [2019-08-10]
  - ncat
- **nbd**
  - nbdkit >= 1.15.5 [2019-10-19]
  - libnbd >= 1.1.6 [2019-10-19]
- **socat** >= 1.7.4 [2021-01-04]
- **iperf-vsock**
  - iperf3 fork
  - https://github.com/stefano-garzarella/iperf-vsock

Concatenate and redirect sockets

- Ncat - https://nmap.org/ncat/
  - new parameter
    - --vsock
  - Examples
    - `ncat --vsock -l 4321`
    - `ncat --vsock 1 4321`

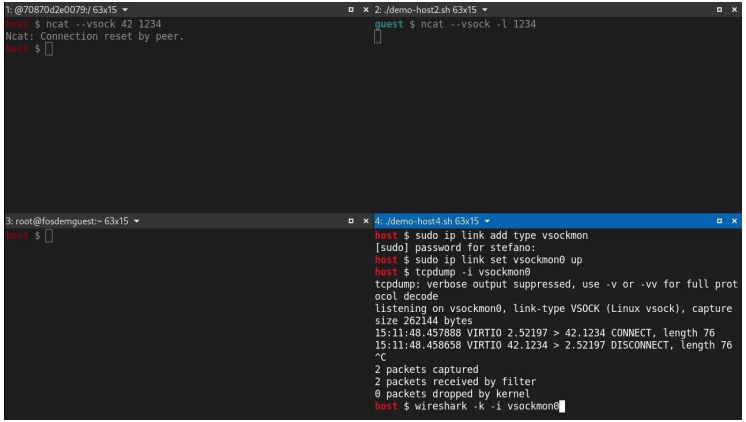
- socat - http://www.dest-unreach.org/socat/
  - New address types
    - VSOCK-LISTEN
    - VSOCK-CONNECT
  - Examples
    - `socat - VSOCK-LISTEN:4321`
    - `socat - VSOCK-CONNECT:1:4321`



Red Hat

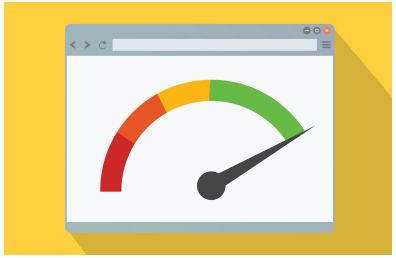
Dump and analyze AF_VSOCK traffic

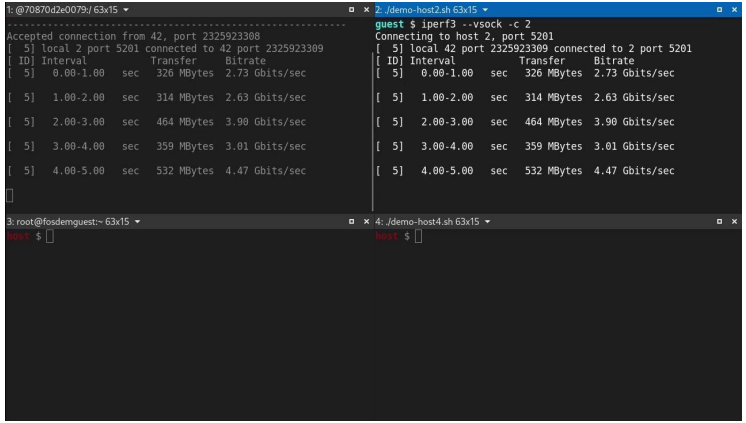
- Prerequisites
  - `vsockmon` kernel module available (CONFIG_VSOCKMON=m)
  - Create vsockmon virtual device to monitor AF_VSOCK sockets
    - `ip link add type vsockmon`
    - `ip link set vsockmon0 up`
- tcpdump
  - `tcpdump -i vsockmon0`
- Wireshark
  - `wireshark -k -i vsockmon0`



Performance evaluation

- iperf-vsock: iperf3 fork with AF_VSOCK support
  - https://github.com/stefano-garzarella/iperf-vsock
  - new parameter
    - --vsock
  - Examples
    - host$ `iperf3 --vsock -s`
    - guest$ `iperf3 --vsock -c 2`
  - Firecracker's hybrid VSOCK over AF_UNIX
    - Host runs iperf server
      - iperf3 --vsock -s -B /tmp/vm.vsock
    - Host runs iperf client
      - iperf3 --vsock -c /tmp/vm.vsock





Red Hat

Next Steps

- SOCK_SEQPACKET
  - Sequenced, reliable, two-way connection-based data transmission path for datagrams
  - [PATCH 0/5] virtio/vsock: introduce SOCK_SEQPACKET support
    - https://lore.kernel.org/netdev/20210103195454.1954169-1-arseny.krasnov@kaspersky.com
- Network namespaces
  - Useful for partitioning VMs or in a nested environment
  - [PATCH net-next 0/3] vsock: support network namespace
    - https://lore.kernel.org/lkml/20200116172428.311437-1-sgarzare@redhat.com/

Next Steps (2)

- Multi-queue
  - Currently one TX / RX queue per vsock device
  - I/O intensive workloads
  - Multiple endpoints communication
- Shared memory
  - Vsock communication using shared memory regions
  - Less buffer copies
  - Fast communication

Q&A

Andra Paraschiv <andraprs@amazon.com>

Stefano Garzarella <sgarzare@redhat.com>
Blog: https://stefano-garzarella.github.io/
IRC: sgarzare on #qemu irc.oftc.net

Thank you!

References

[1] https://man7.org/linux/man-pages/man7/vsock.7.html

[2] https://aws.amazon.com/ec2/nitro/nitro-enclaves/

[3] https://docs.aws.amazon.com/enclaves/latest/user/nitro-enclave.html

[4] https://www.kernel.org/doc/html/latest/virt/ne_overview.html

[5] https://github.com/aws/aws-nitro-enclaves-cli

[6] https://github.com/aws/aws-nitro-enclaves-samples

References (2)

[7] https://katacontainers.io

[8] https://github.com/kata-containers/documentation/blob/master/design/VSocks.md

[9] https://katacontainers.io/learn

[10] https://github.com/kata-containers/documentation/blob/master/design/architecture.md

[11] https://github.com/containers/libkrun

[12] https://github.com/containers/libkrunfw