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
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)
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](https://katacontainers.io)
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
© 2021, Red Hat, Inc.
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

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

© 2021, Red Hat, Inc.
Kata Containers demo
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)

[GitHub Link](https://github.com/containers/libkrun)
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
© 2021, Red Hat, Inc.
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/slpnx/status/1348406770990002178?s=20
libkrun demo
Developing with AF_VSOCK
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)
# 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')

# 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

- CID(s)
  - 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](https://github.com/stefano-garzarella/iperf-vsock)
Concatenate and redirect sockets

- **Ncat** - [https://nmap.org/ncat/](https://nmap.org/ncat/)
  - new parameter
    - `--vsock`
  - Examples
    - `ncat --vsock -l 4321`
    - `ncat --vsock 1 4321`

- **socat** - [http://www.dest-unreach.org/socat/](http://www.dest-unreach.org/socat/)
  - New address types
    - VSOCK-LISTEN
    - VSOCK-CONNECT
  - Examples
    - `socat - VSOCK-LISTEN:4321`
    - `socat - VSOCK-CONNECT:1:4321`
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](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
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](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/](https://lore.kernel.org/lkml/20200116172428.311437-1-sgarzare@redhat.com/)
Next Steps (2)

- **Multi-queue**
  - Currently one TX / RX queue per vssock 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

References (2)


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

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


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