From Secure Container to Secure Service

Xu Wang & Fupan Li, Ant Financial
We summarized the progress.

We talked about the overhead and other issues.

And we predicted the secure containers is going to production in 2019.
Additional Background

Long tutorial in KubeCon NA 2018 by Lei & Me

Hands-on: K8s + containerd + Kata Containers

Deck and Video:
"The only real solution to security is to admit that bugs happen, and then mitigate them by having multiple layers."

---Linus Torvalds (LinuxCon NA 2015, Seattle)
Container Runtimes on Linux

Linux Containers
By Process Isolation

- Process A
  - Filter: Seccomp, MAC, CAPS
  - Namespaces

- Process B
  - Filter: Seccomp, MAC, CAPS
  - Namespaces

- Process C
  - Filter: Seccomp, MAC, CAPS
  - Namespaces

Kata Containers
(Secure Container)

- Virtual Machine
  - Process A
    - Namespaces
    - Linux Kernel A
  - VMM (Qemu, Firecracker...)

- Virtual Machine
  - Process B
    - Namespaces
    - Linux Kernel B
  - VMM (Qemu, Firecracker...)

- Virtual Machine
  - Process C
    - Namespaces
    - Linux Kernel C
  - VMM (Qemu, Firecracker...)

- cgroups
  - Process A
    - Sentry (Kernel)
    - Platform (KVM or ptrace)

- cgroups
  - Process B
    - Sentry (Kernel)
    - Platform (KVM or ptrace)

- cgroups
  - Process C
    - Sentry (Kernel)
    - Platform (KVM or ptrace)

Kata Containers
(Secure Container)

- gVisor
  - Independent kernel for each POD sandbox
  - Resource Isolation + Security Isolation

Linux Kernel

Platform (KVM or ptrace)
A Brief History of Kata Containers

May 2015:
Clear Containers and runV open sourced

Dec 2017:
Clear Containers and runV merged into Kata Containers and hosted in OpenStack Foundation as the first Pilot Project

Apr 2019:
Kata Containers was confirmed by foundation board as the second project of OpenStack Foundation

Updates since KubeCon NA 2018

- May 2018 1.0.0
- Jul 2018 1.1.0
- Aug 2018 1.2.0
- Sep 2018 1.3.0
- Nov 2018 1.4.0
- Jan 2019 1.5.0
- Mar 2019 1.6.0
- May 2019 1.7.0
Shim v2 Support in Kata 1.5

Eliminated 2N+1 helper processes
FireCracker Support in Kata 1.5

Firecracker

- Open sourced by AWS - Nov 2018
- From their GitHub page:
  "Firecracker has a minimalist design. It excludes unnecessary devices and guest-facing functionality to reduce the memory footprint and attack surface area of each microVM. This improves security, decreases the startup time, and increases hardware utilization."

Kata + Firecracker integration status

- With minimal design of the VMM, there are limitations when using Kata+Firecracker:
  - No filesystem sharing with host
  - No hardware device support
  - No dynamic resizing of the guest (vCPU/memory hotplug)
On each node, you can run workloads which will utilize runc, kata-qemu and kata-firecracker.

You can select your method of isolation on a per-workload (per-pod) basis.
Virtio-fs Support in Kata 1.7

- Origin from RedHat
- Based on fuse, better POSIX compatibility
- VirtIO based, native design for virtualization (not another network FS)
- With DAX, better performance and lower memory overhead in guests
- Userspace virtiofs daemon, more flexible
Summary of the Progress

• Better integration with Kubernetes
• Less memory overhead
• Improvements on filesystem sharing
Well, Security is an End-to-End Issue

We need not only secure container runtime, but secure services in Financial Scenarios.
ServiceMesh: Evolution of the Financial Grade Infrastructure

Service

- Business Logic
- Protocol Encoder
- Service Discovery
- Rating Limit
- Load Balance
- Traffic Routing

Pod

User Container

User Container

Or Part of Infra?

Sidecar

- Service Discovery
- Rating Limit
- Load Balance
- Traffic Routing

In Secure Container
Service Mesh + Kata Containers

Service Mesh (Istio)

- Secure by default
- Defense in depth
- Zero-trust network

Control Plane

Application Level Defense By Service Mesh

Data Plane

Virtual Machine

- App A
  - Linux Kernel B
  - VMM (Qemu, Firecracker...)

Virtual Machine

- App B
  - Linux Kernel C
  - VMM (Qemu, Firecracker...)

Virtual Machine

- App C
  - Linux Kernel B
  - VMM (Qemu, Firecracker...)

Virtual Machine

- App D
  - Linux Kernel C
  - VMM (Qemu, Firecracker...)

Data Plane

Node

- sidecar
- mTLS

Node

- sidecar
- RBAC

System Level Defense By Secure Container

- Identity
- Policy
- AAA
- Encryption

- Secure by default
- Defense in depth
- Zero-trust network

- Secure Container
- System
- Level
- Defense
- By

- Service Mesh

- RBAC
- mTLS

- Citadel
- Pilot
- Mixer

- Secure by default
- Defense in depth
- Zero-trust network

- System
- Defense
- By

- Secure Container

- RBAC
- mTLS

- Citadel
- Pilot
- Mixer
• ServiceMesh Security Mechanisms and Kata Containers
  • Enforce mTLS Data Plane for Kata + Istio (video)
    https://istio.io/docs/tasks/security/authn-policy/#namespace-wide-policy
  • Enable RBAC for ingress traffic for Kata + Istio (video)
    https://istio.io/docs/tasks/security/authz-http/#enforcing-namespace-level-access-control
SOFAMesh: Service Mesh Practice in Ant Financial

SOFAMesh

- Large-scale Service Mesh Practice
- Based on Istio, with improvements and extensions
  - SOFAMosn (in golang) as sidecar to replace envoy
  - Migrate mixer to data plane for performance
  - Improve Pilot for more flexible service discovery
  - Performance improvement of Pilot
- Support RPC: SOFARPC/Dubbo/HSF
- Verified in Ant Financial, and feed back to community
- Open Source: [https://github.com/sofastack/sofa-mesh](https://github.com/sofastack/sofa-mesh)

SOFAMosn

- Not only Service Mesh Sidecar in SOFAMesh
- But also: API Gateway, Ingress Gateway
- Support envoy xDS v2 API
- Open Source: [https://github.com/sofastack/sofa-mosn](https://github.com/sofastack/sofa-mosn)
The Next Step

• Current:
  • Kata works with Istio / SOFAMesh

• In the Future:
  • Mesh sidecar optimization in Kata Context (w/ eBPF etc.)
    • And Interoperability with non-kata containers
  • Resource isolation between mesh sidecar and user containers
谢谢！Thank You