Research: Firecracker Isolation

The Diagrams in this document are not mine. They were sourced from the web.

The Basics

Firecracker is a virtual machine manager (VMM) designed for running transient and short-lived processes that run on microVMs. It is optimized for running functions and serverless workloads that require faster cold start and higher density.

VMs vs. Containers

Containers are faster and lighter when compared to virtual machines, however containers are considered to be less secure because of relaxed isolation levels.

You can launch and run serverless workloads on containers, however the size of Docker images may adversely impact startup times of functions. Firecracker doesn't have this drawback as it runs on bare metal resources. The key takeaway here is not to deliver a serverless solution via a container.

Please note that container workloads can run on firecracker very efficiently but the opposite is not true. This is the implementation behind Amazon ECS Fargate. Amazon takes containers and deploys them on multi-tenant microVMs at scale.

Enter Firecracker

Firecracker is Amazon's answer to serverless. It's the low-level implementation of Amazon Lambda service that they have open sourced to the world. Think of firecracker as a tool to manage a bunch of short-lived micro virtual machines spun up on-demand to perform a specific function.

How Fast is Firecracker

From Amazon:

> *The microVMs launched by Firecracker are extremely transient and short-lived. You can only access them through UART/serial console because they don't even run SSH. Apart from the serial console, these microVMs may be connected to a virtual NIC, a block device and a one-button keyboard. That's pretty much what you can attach to the VM. This minimalistic design of the VMM makes Firecracker extremely fast. According to the official claims, Firecracker initiates user space or application code in less than 125ms and supports microVM creation rates of 150 microVMs per second per host.*
>
> *The Firecracker process exposes REST API via a UNIX socket, which can be used to manage the lifecycle of a microVM. The architecture is very similar to Docker Engine for exposing the control plane API. While there is no CLI yet, cURL can be used to send the payload to the Firecracker REST endpoint.*

Where can Firecracker Run

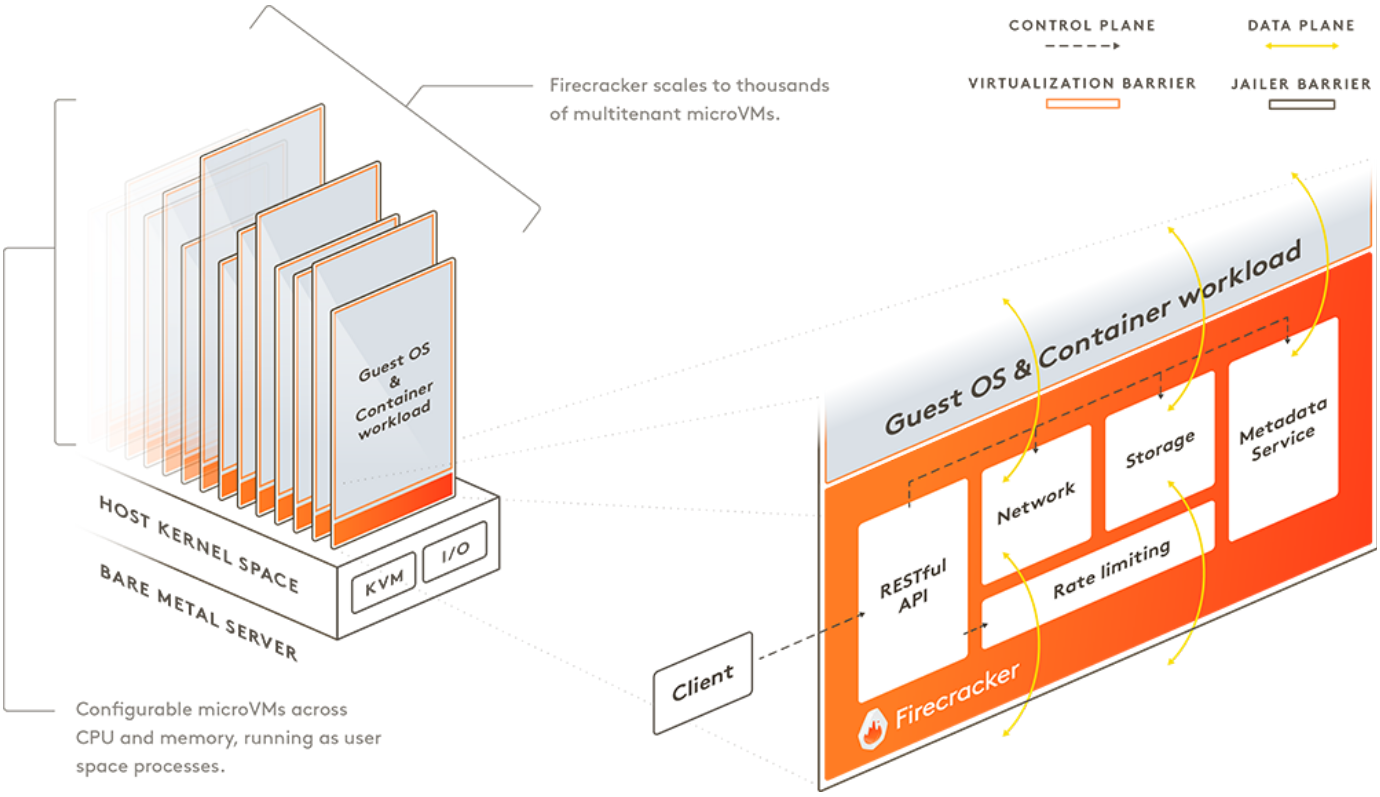
> *Customers can run Firecracker on AWS .metal instances as well as on any other bare-metal servers, including on-premises environments and developer laptops. Firecracker runs on Intel processors today, with support for AMD and ARM coming in 2019.*

Security & Isolation

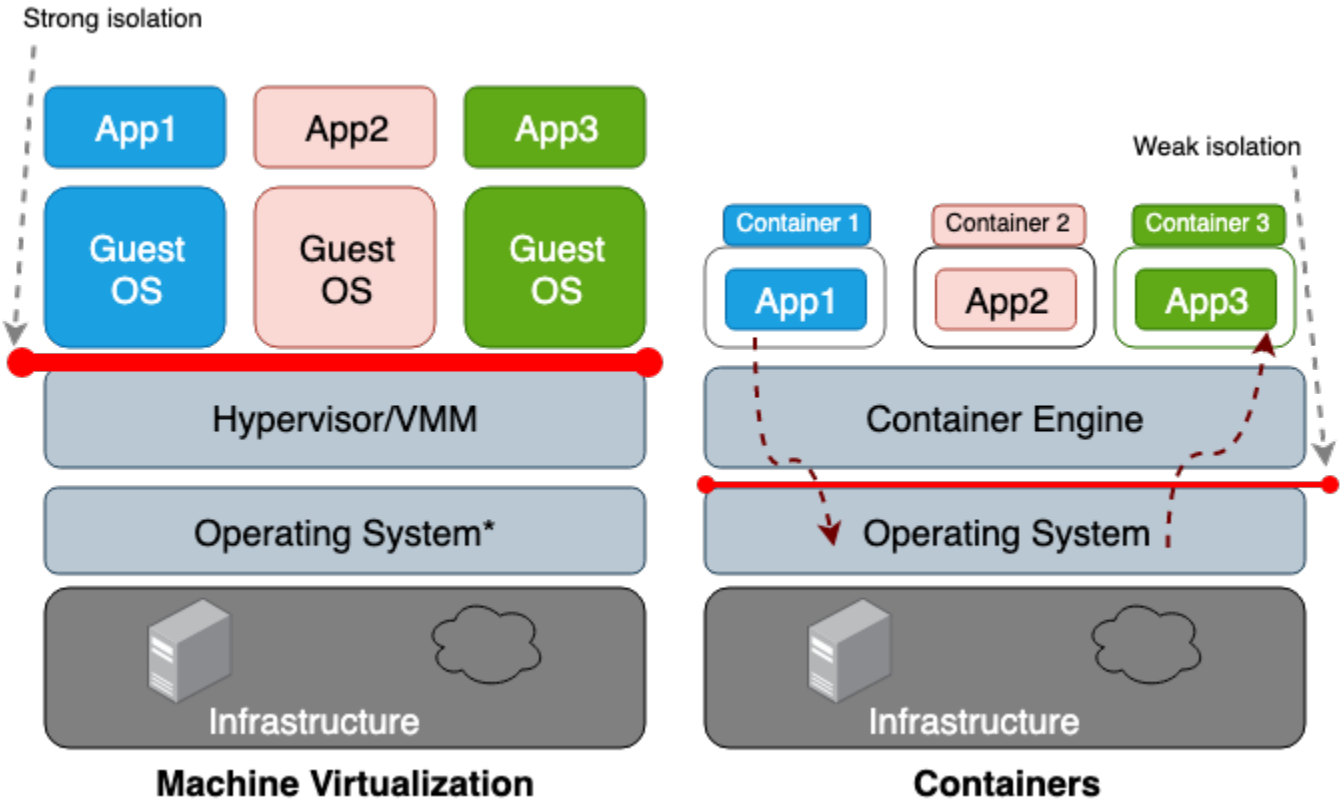
From Amazon:

> *Each microVM runs as a process within the host OS, which is associated with a dedicated socket and API endpoint. The VMs also support EC2-like metadata at well-known endpoints that can be used for service discovery and storing arbitrary data as key-value pairs.*
>
> *AWS has included a Jailer that secures microVMs by providing additional security boundaries through cgroup, namespace, and seccomp isolation.*

Also from Amazon:



Based on the diagram above the client communicates with firecracker through a restful API to manage microVMs. Each micro-vm is its own sandbox environment. In summary, microVMs afford us the same isolation traditional full-blown virtual machines do.



From the world wide web:

> *The main difference between a virtual machine (VM) and a container is that the VM is a hardware-level virtualization and a container is a OS-level virtualization. VM hypervisor emulates a hardware environment for each VM, where the container runtime emulates an operating system for each container. VMs share the host's physical hardware and containers share both the hardware and the host's OS kernel. Because containers share more resources from the host, their usages of storage, memory,*

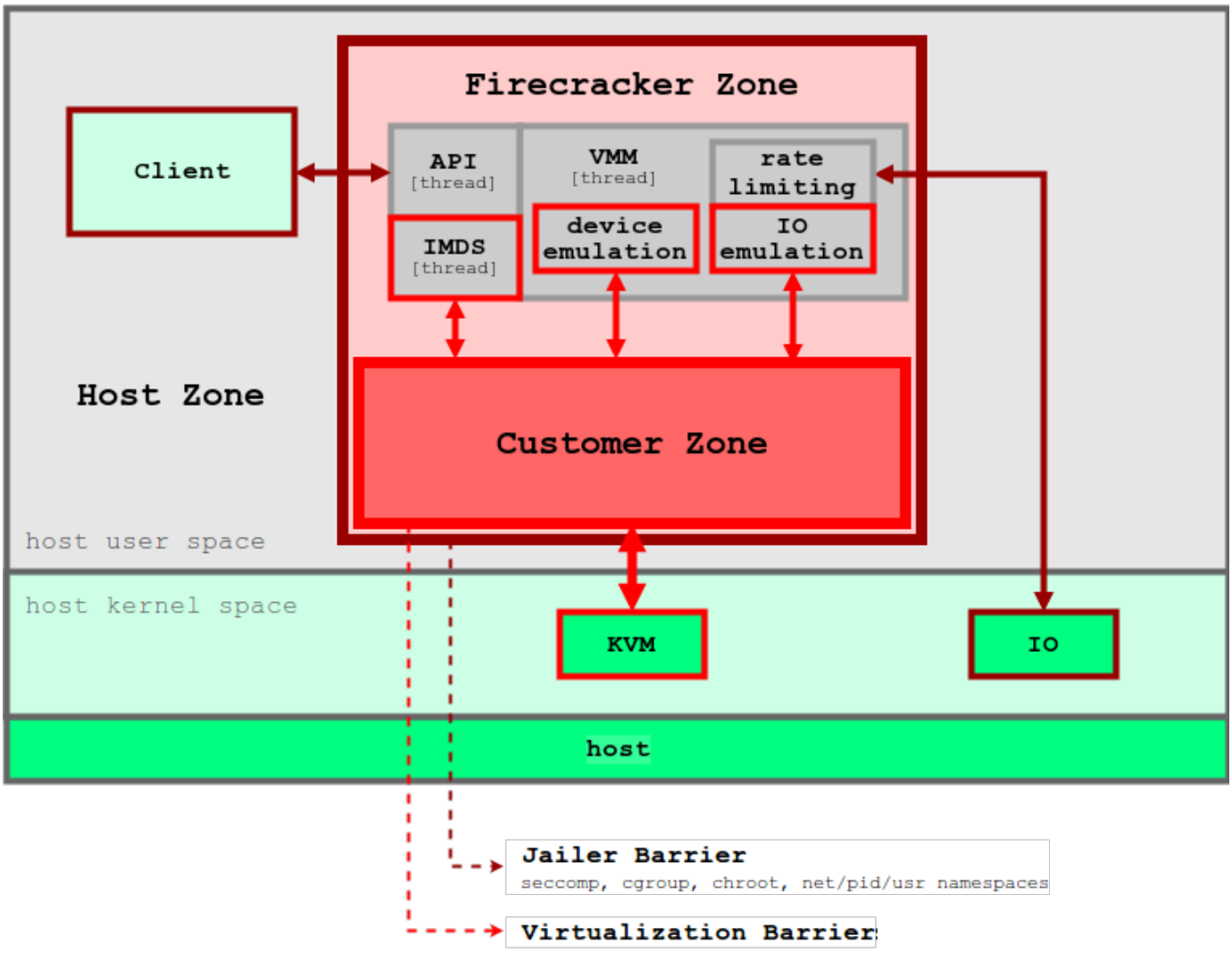
> *and CPU cycles are all much more efficient than a VM. However, the downside of more sharing is the weaker trust boundary between the containers and the host.*

Now you may be asking, how are microVMs different than a regular VM:

> *Firecracker is built with minimal device emulation that enables faster startup time, provides a reduced memory footprint for each microVM, and offers a trusted sandboxed environment for each container.*

More on Isolation:

> *Although VMs create strong isolation for containers in public cloud, using general-purpose VMMs and VMs for sandboxing applications is not very resource-efficient. Firecracker solves both the security and performance issues by creating a VMM specifically for cloud-native applications. Firecracker VMM provides each guest VM with the minimal OS functionalities and emulated devices to enhance both the security and performance.*



Understanding the Firecracker Jailer

> *Being written in Rust mitigates some risk to the Firecracker VMM process from malicious guests. But Firecracker also ships with a separate jailer used to reduce the blast radius of a compromised VMM process. The jailer isolates the VMM in a chroot, in its own namespaces, and imposes a tight seccomp filter. The filter whitelists system calls by number and optionally limits system-call arguments, such as limiting `ioctl()` commands to the necessary KVM calls. Control groups version 1 are used to prevent PID exhaustion and to prevent workloads sharing CPU cores and NUMA nodes to reduce the likelihood of exploitable side channels.*
>
> *The recommendations include a list of host security configurations. These are meant to mitigate side channels enabled by CPU features, host kernel features, and recent hardware vulnerabilities.*

Summary

Firecracker provides for robust isolation. It is designed to minimize the blast radius in the case of an attack and provides for better isolation and security than a traditional virtual machine by further restricting OS functionality and emulated devices. Security Enhanced Linux can be enabled on each microVM to meet security requirements and increase process and application isolation.

Further Reading

Refer to the following links for additional information:

https://firecracker-microvm.github.io

https://lwn.net/Articles/775736/