black hat USA 2020

# Escaping Virtualized Containers

Yuval Avrahami
Palo Alto Networks

paloalto NETWORKS

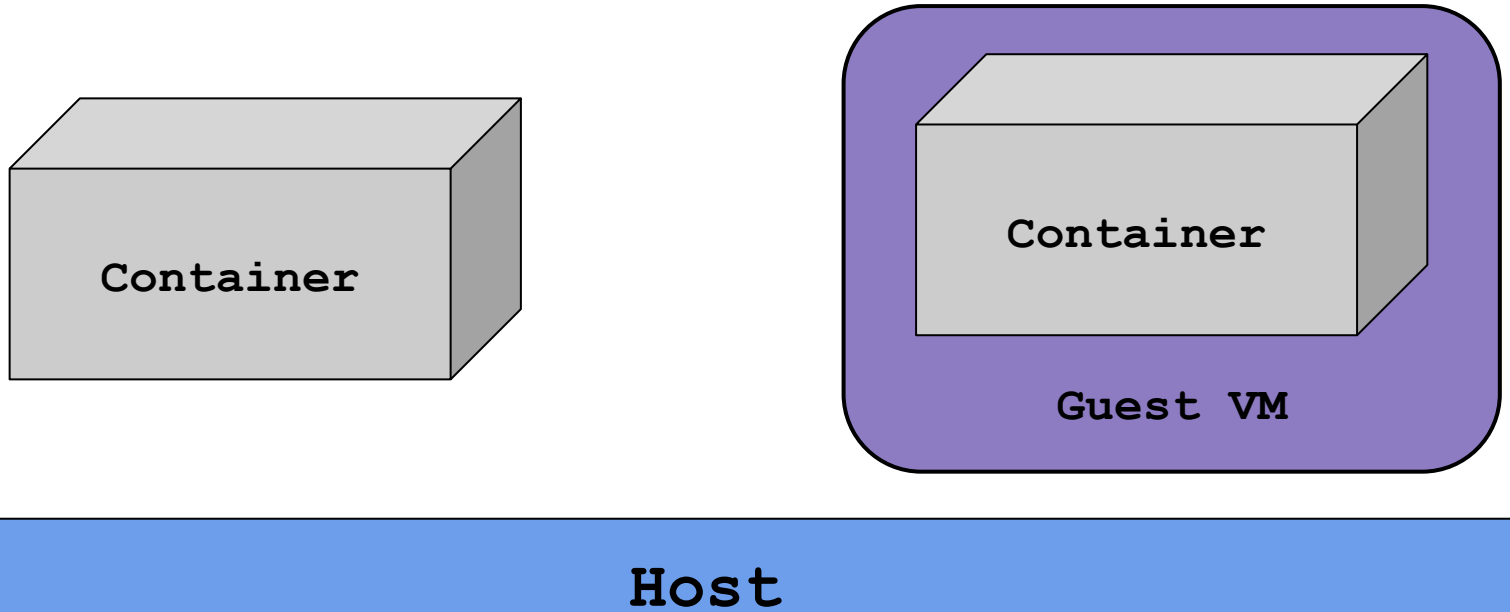
# Agenda

- Containers 101
- Kata Containers
  - Virtualized Containers Runtime
- Break out of the Container
  - Container Escopology
- Escape the VM
- Takeaways

## Classic Containers

VS

## Kata Containers



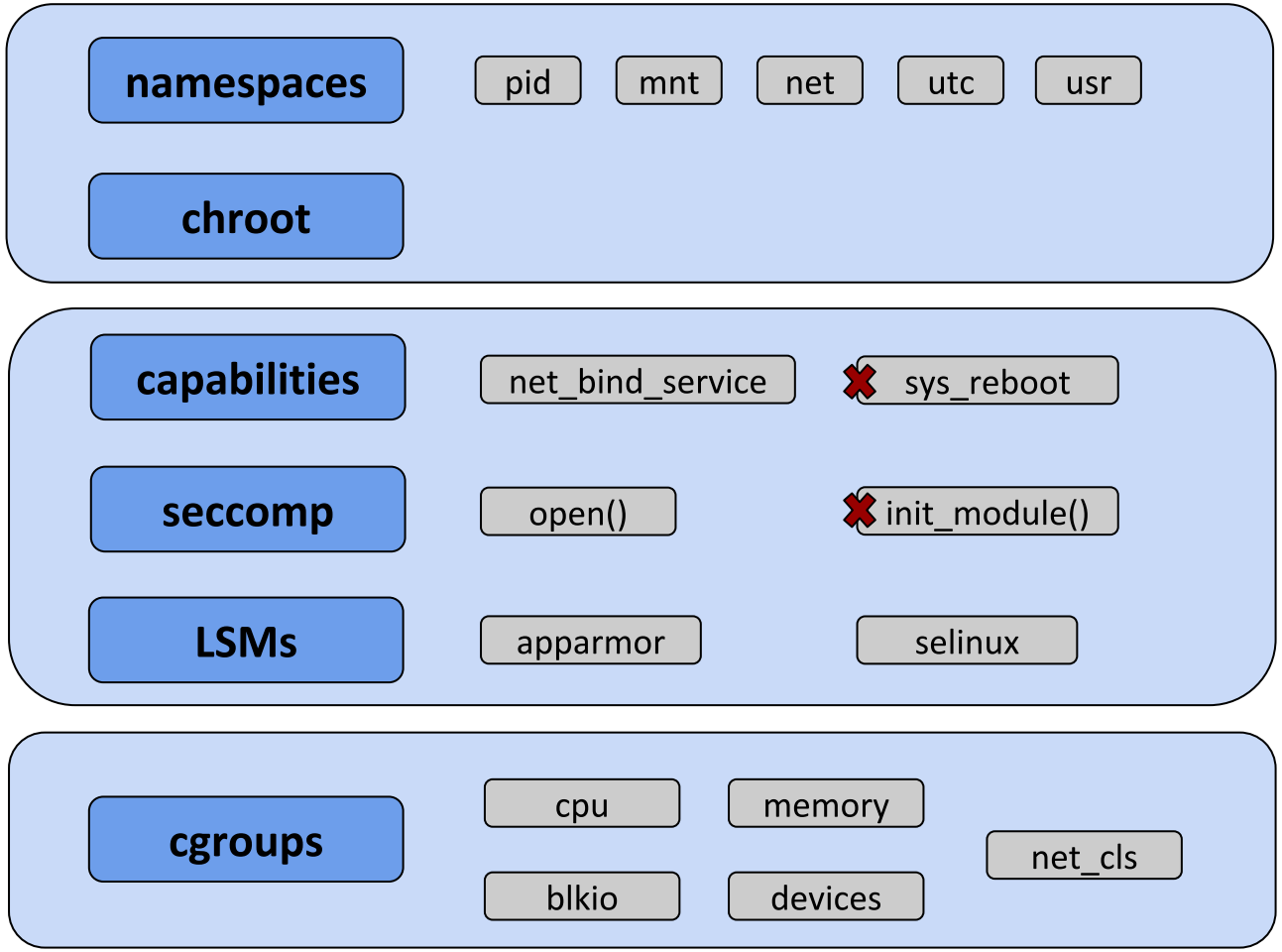
Containers

Restricted **processes**
running in a
separate filesystem

What Can I See?

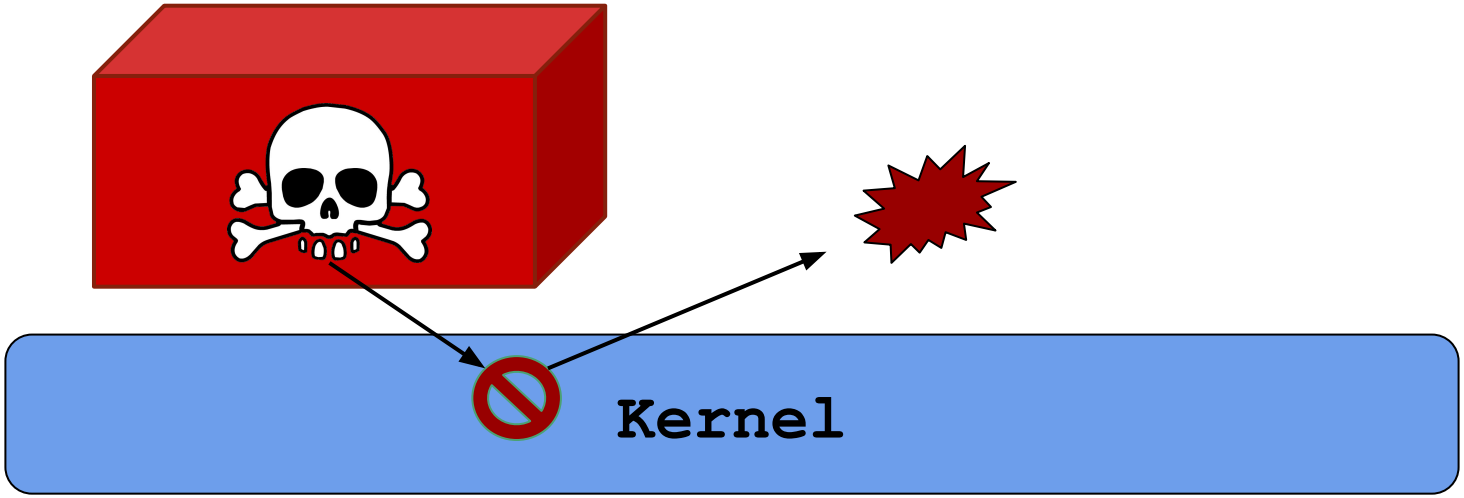
What Can I Do?

Resource Isolation



# Shared Kernel - a Disturbing Attack Surface

- Unlike VMs, containers share the host's Kernel
- Kernel vulnerabilities may lead to breakouts

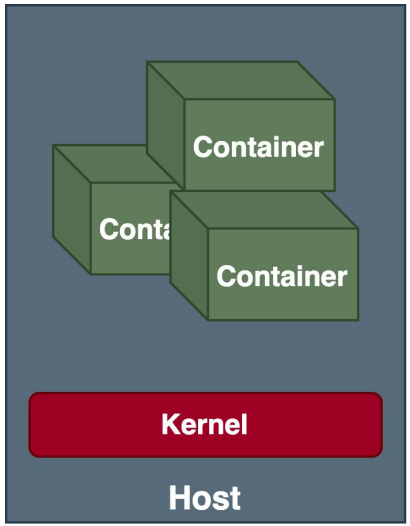


# Kata Containers

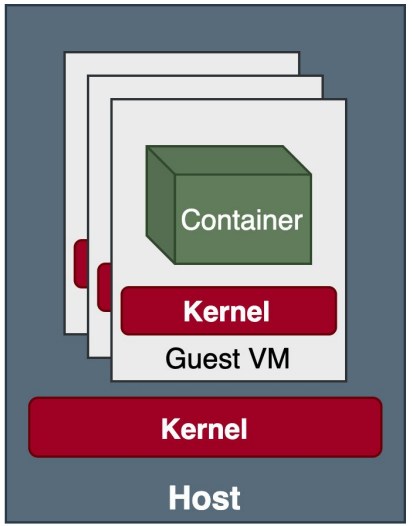
Sandboxing Containers

# Kata Containers

- Virtualized Containers
- **Encapsulates** each container inside a lightweight VM
- **Simple way to sandbox containers**
  - Compatible runtime for Docker & Kubernetes



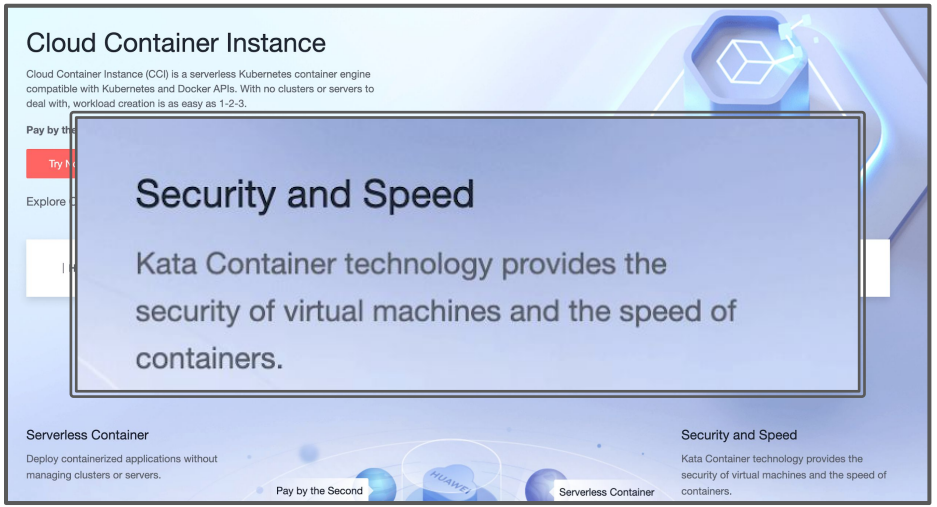

Traditional Containers



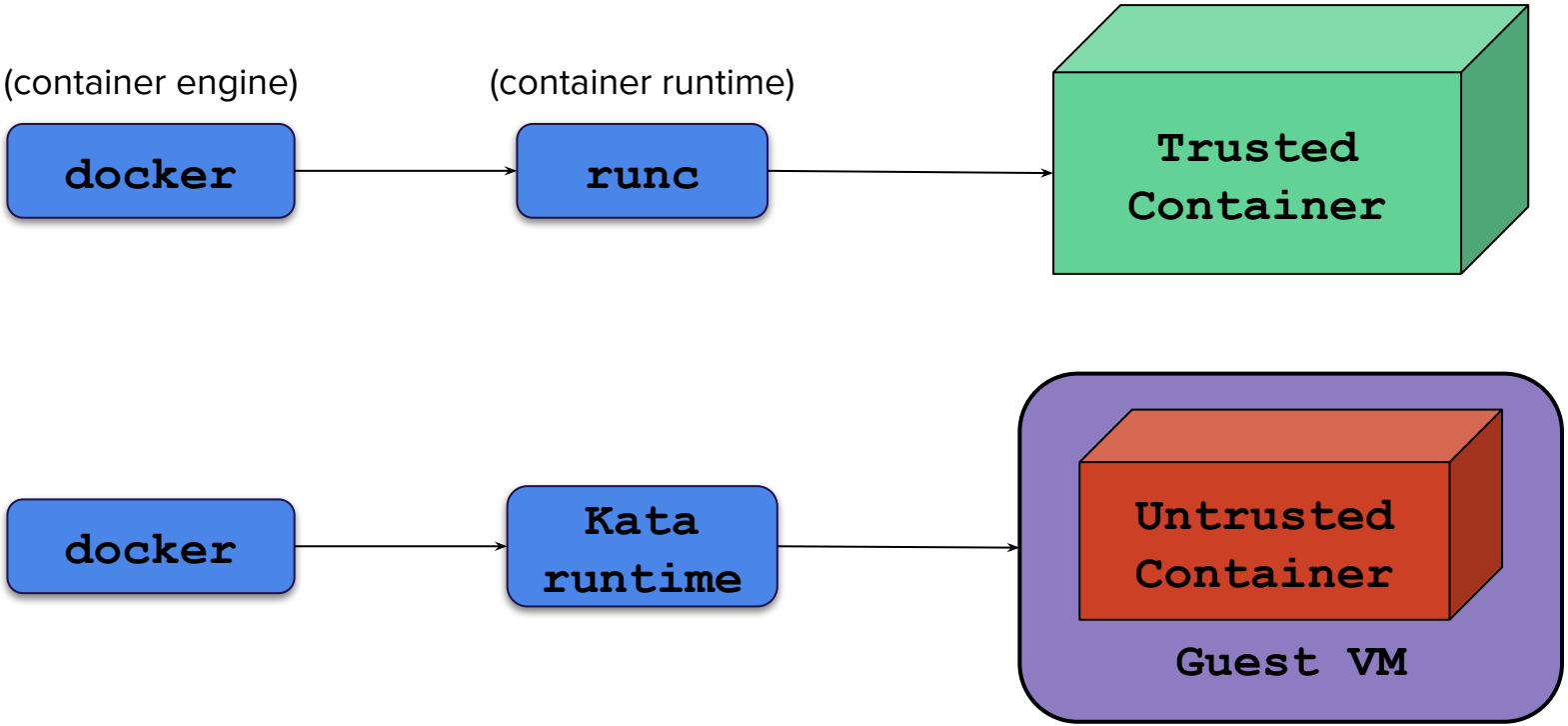

Kata Containers

# Use Cases

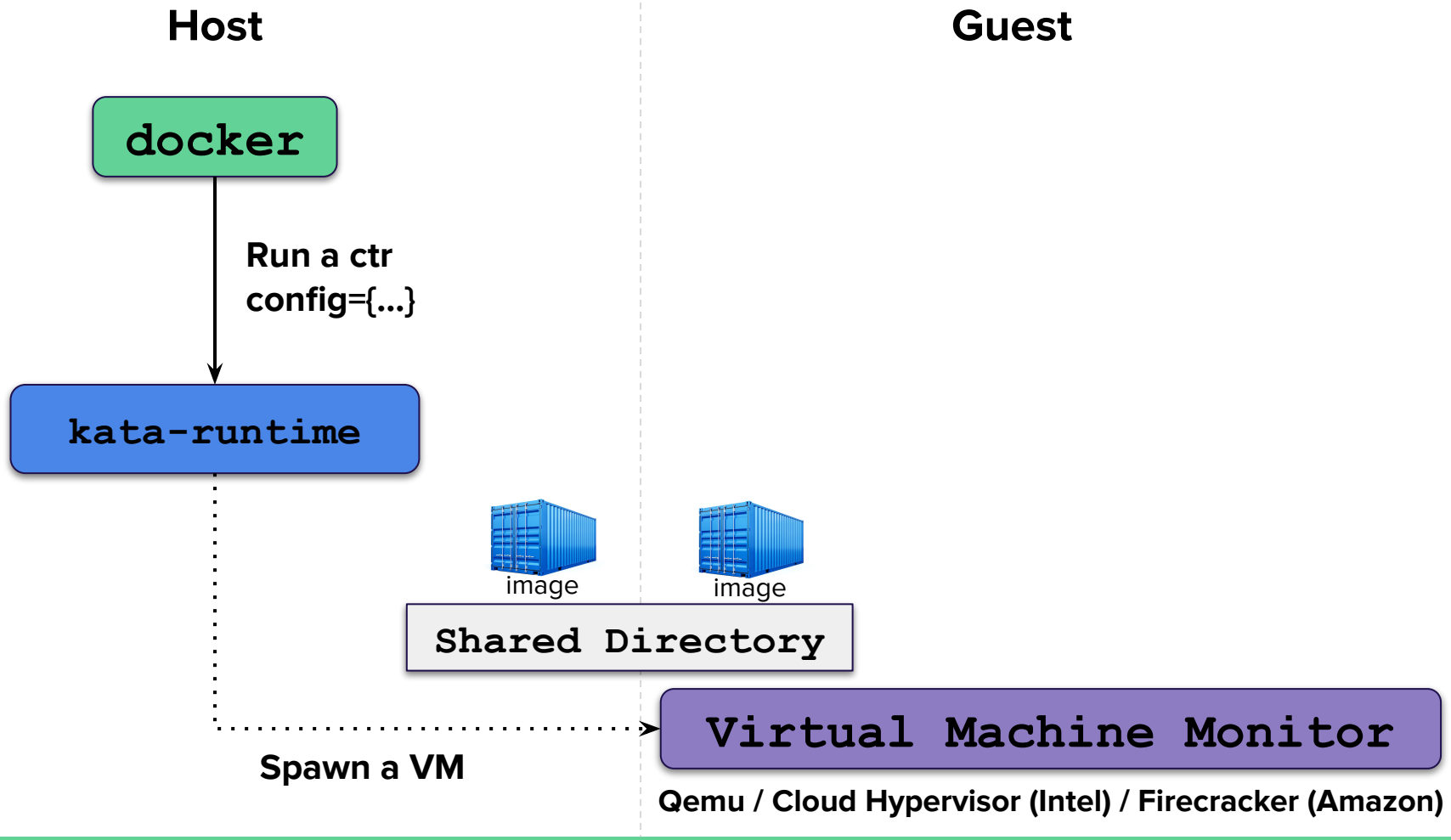
- Untrusted or targeted workloads
- Multi-tenant environments
- Cloud Service Providers

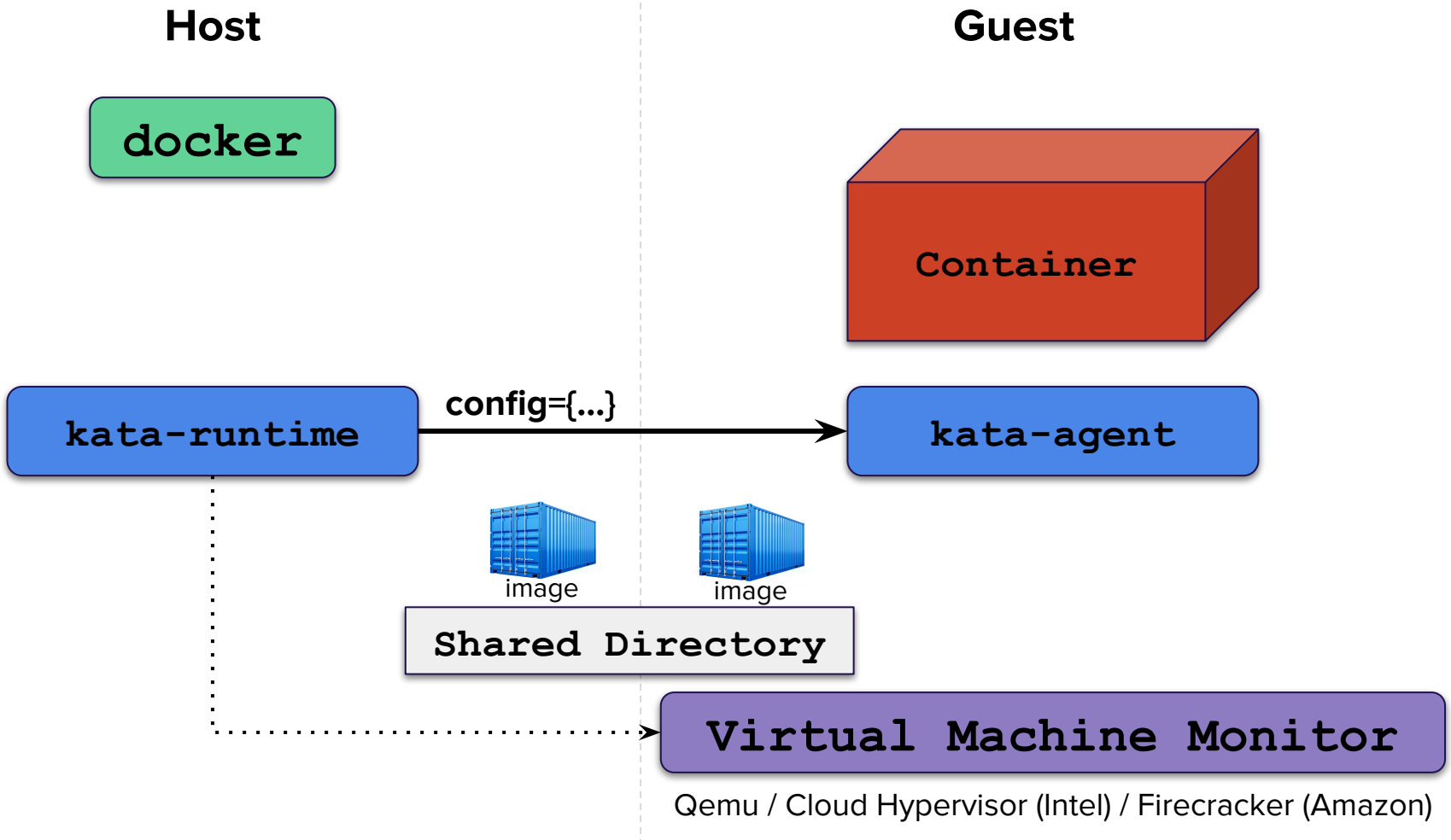


# Using Kata


(container engine)
(container runtime)
docker
runc
Trusted Container
docker
Kata runtime
Untrusted Container
Guest VM

```
$ docker run --runtime=kata ubuntu bash
```


Host
Guest
docker
Run a ctr
config={...}
kata-runtime
image
image
Shared Directory
Virtual Machine Monitor
Spawn a VM
Qemu / Cloud Hypervisor (Intel) / Firecracker (Amazon)


Host
Guest
docker
Container
kata-runtime
config={...}
kata-agent
image
image
Shared Directory
Virtual Machine Monitor
Qemu / Cloud Hypervisor (Intel) / Firecracker (Amazon)

# Let’s Escape!

# Why?

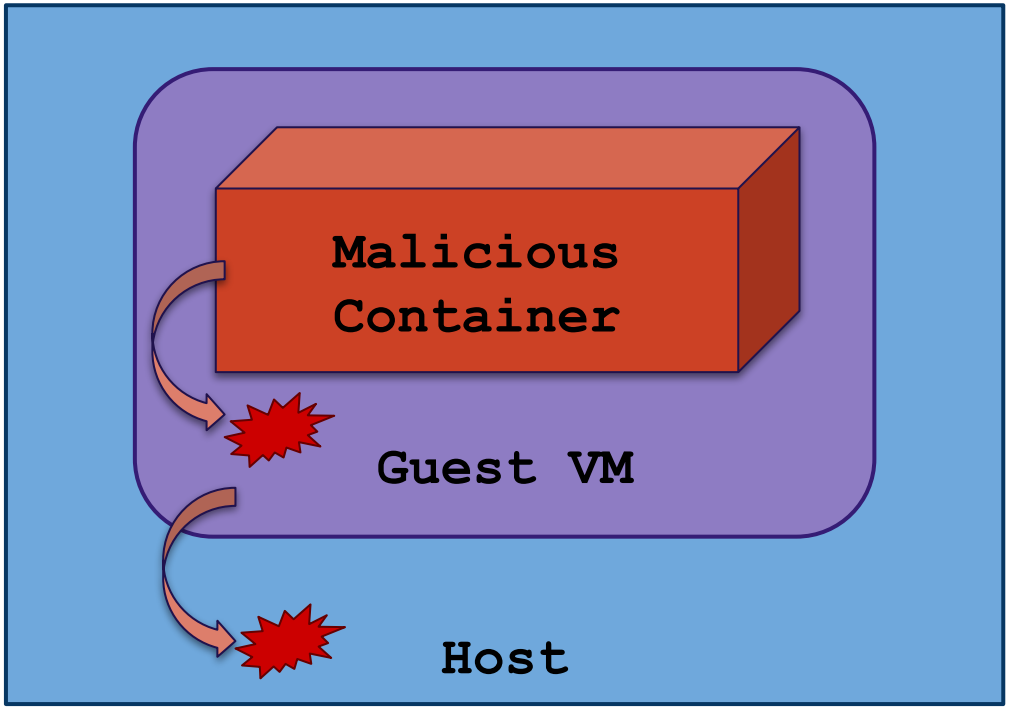
- Fun and challenging
  - Two isolation layers to break
- **Learn about container security**

# Attack Scenarios

- Enterprises use Kata to contain untrusted / targeted containers
  - We're that untrusted container 
- Cloud Service Providers use Kata to segregate containers from different customers
  - We're the evil customer running a malicious container 

# The Plan

- Escape the container
- Break out of the VM



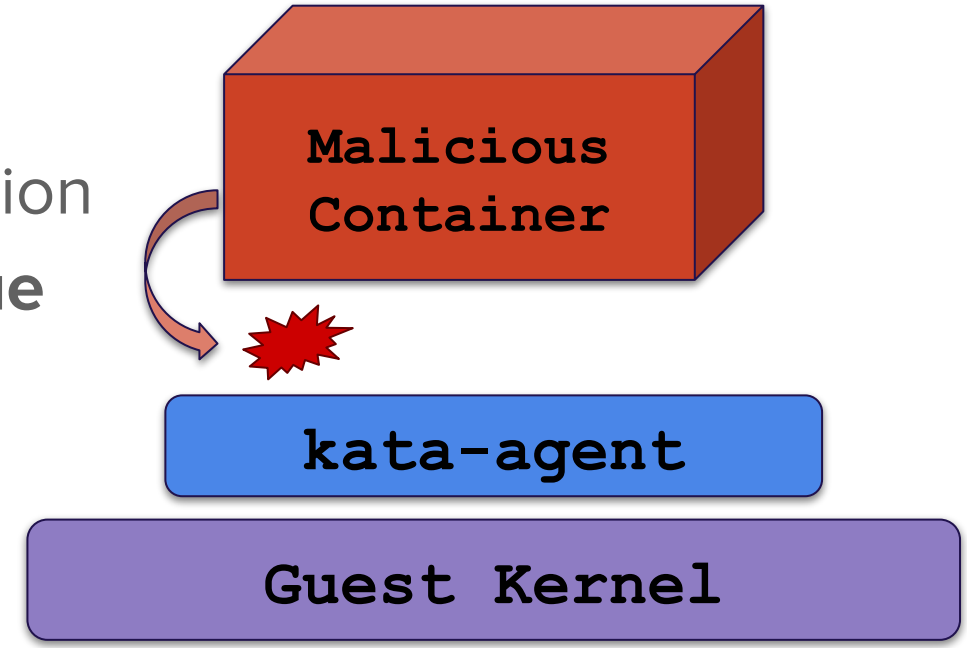
# Scope

- Kata Containers is highly configurable
  - Vulnerabilities won’t work in every config, targeting standard
- Focus on simple single-container guests under Docker
  - K8s+kata vulnerable to issues, exploitation gets complex
- Not an indictment against Kata

# Escape the Container

# Escape the Container

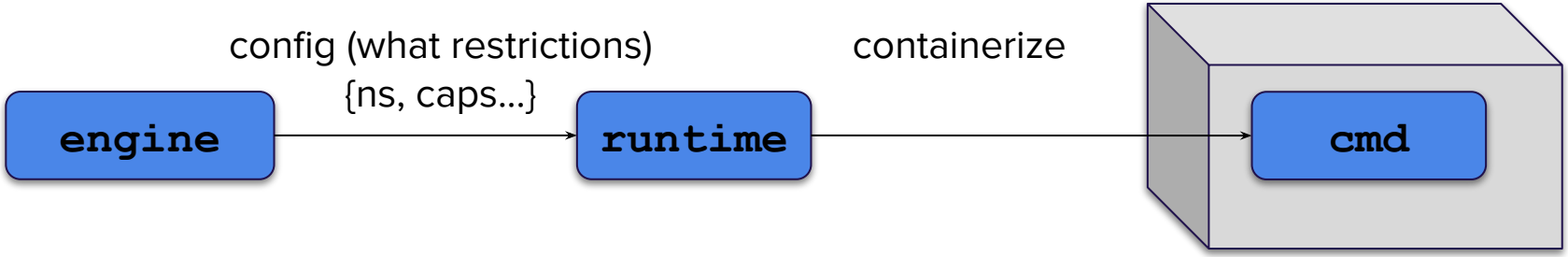
- Don’t rely on a guest kernel privilege escalation
- **Find a Kata-native issue**



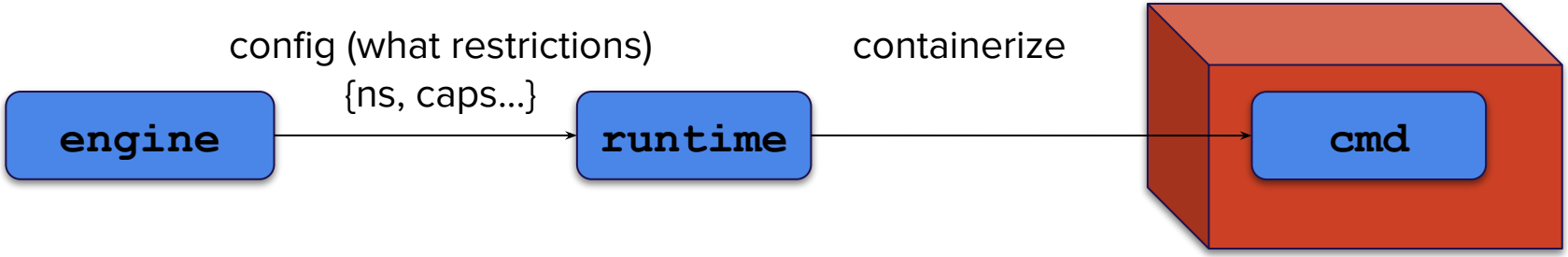
# Container Escapology

## In a nutshell

# The Usual Suspects



# Container Escapology (in a nutshell)




**1) Setup of the containerized process (runtime issue)**

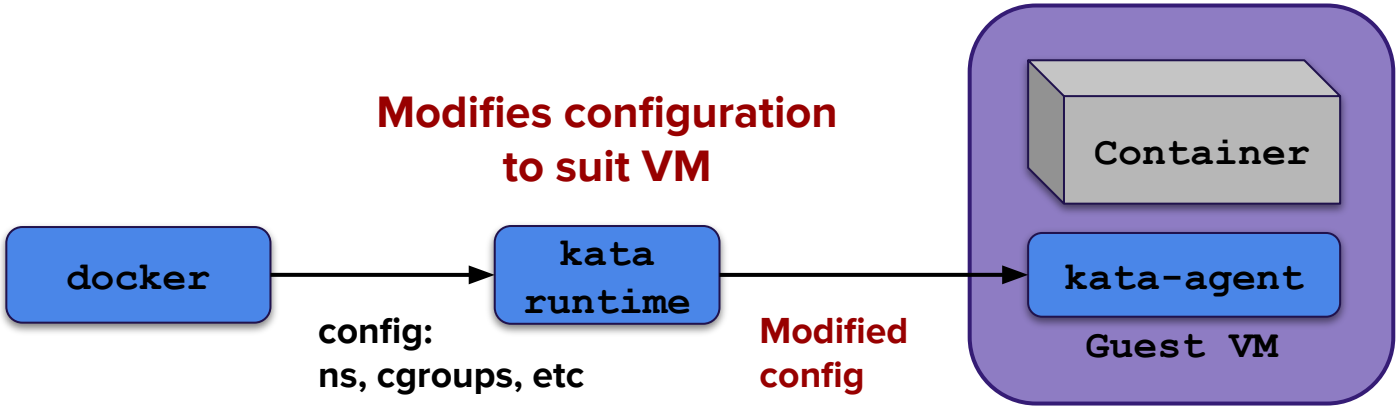
- Untrusted variables: image, cmd, existing ctr

**2) Running container isn't restricted enough (engine issue)**

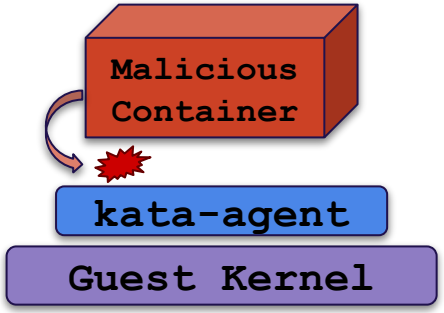
- Permissive engine defaults, new breakout technique

# Kata Modifies the Container's Config

- Kata changes the config received from engine
  - Config generated on host needs to adjusted for VM
- That's dangerous!



# Kata Modifies the Container's Config



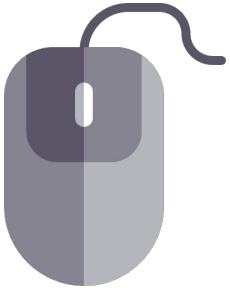

- Kata discards several cgroups
  - Host and guest have different hardware resources
  - Some cgroups don't make sense it the the guest
    - blkio, device
- Cgroups are **mainly** about denying DoS
  - Container DoSing the guest isn't an issue

# Device cgroup

Not only DoS

# Device cgroup

- Restricts container’s access to system’s devices
- Kata doesn’t enforce
- What guest device can interest us?
  - The hard disk!



# Accessing Hard Disk / Block Devices?

```
root@test:~$ mkdir -p test/fs
root@test:~$ mknod test/dev-sda1 b 8 1  # sda1 = 8:1
root@test:~$ mount test/dev-sda1 test/fs/
root@test:~$ ls test/fs
bin         boot  etc   initrd.img      lib    lost+found
bin_copy    dev   home  initrd.img.old  lib64  media
```

- Container has **CAP_MKNOD** but no **CAP_SYS_ADMIN** - can't mount

```
yuval@bh:~$ docker run -it --rm --runtime=kata-qemu yuvalavra/util
root@426c0751a9cf:/# mknod /dev/guest_hd b 259 1
root@426c0751a9cf:/# mkdir guest_fs
root@426c0751a9cf:/# mount /dev/guest_hd guest_fs
mount: /guest_fs: permission denied.
```

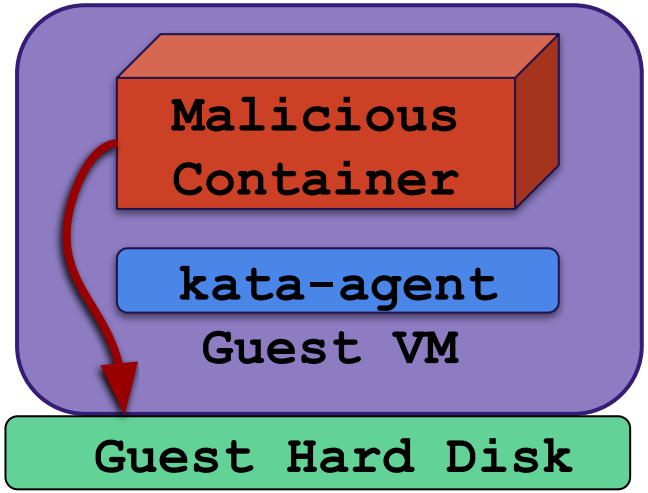
# Direct Device Access

- Directly reading / writing to device file
  - Normally used to debug and fix corrupted hard disk
  - `debugfs` - ext2/3/4 filesystem debugger

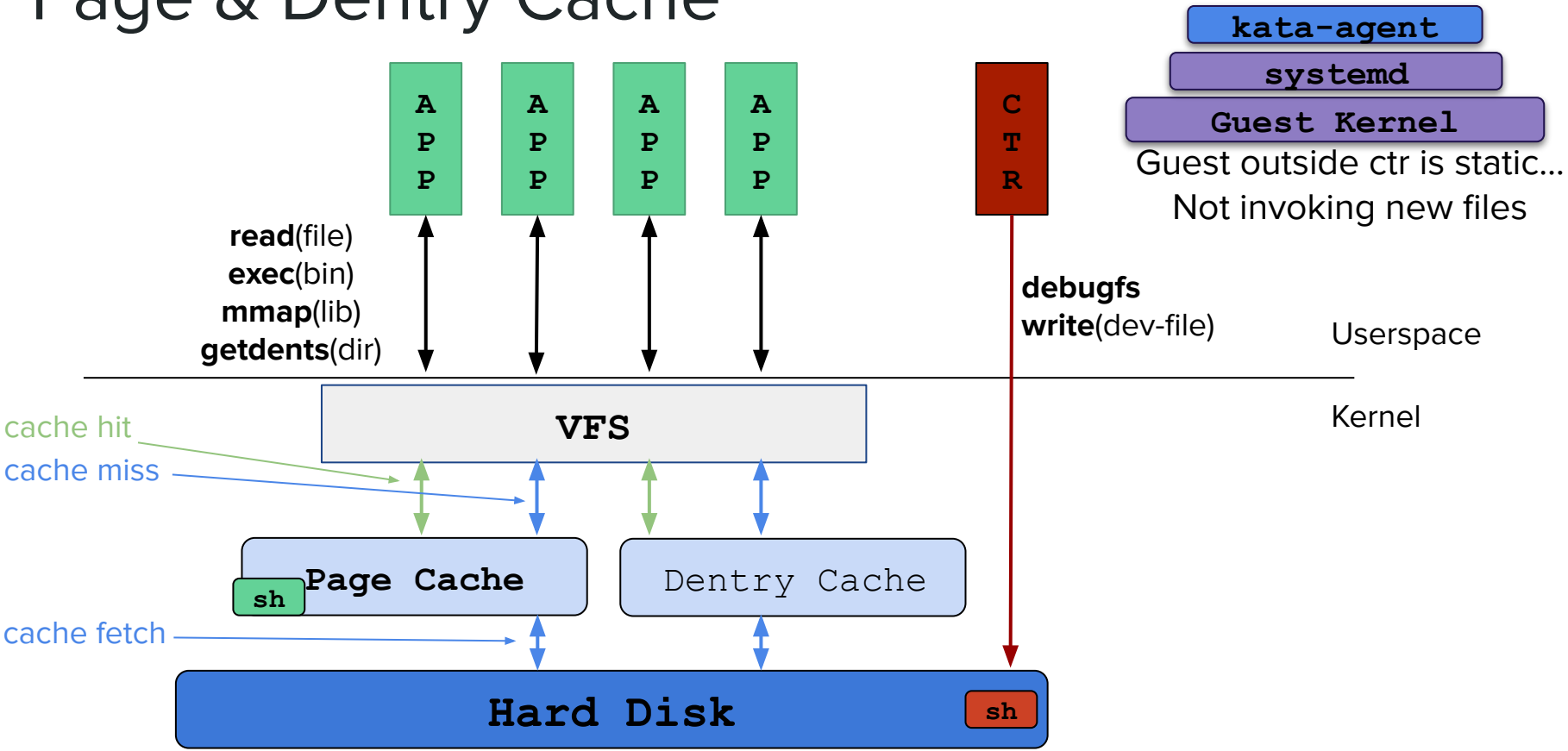
```
yuval@bh:~$ docker run -it --rm --runtime=kata-qemu yuvalavra/util
root@63f271fdf934:/# mknod --mode 0600 /dev/guest_hd b 259 1
root@63f271fdf934:/# debugfs /dev/guest_hd -R "ls /usr/bin" | grep kata-agent
 435  (12) .    33  (12) ..    436  (20) kata-agent    437  (16) ldconfig
root@63f271fdf934:/# debugfs -w /dev/guest_hd -R "write /bin/bash test"
Allocated inode: 169
root@63f271fdf934:/# debugfs /dev/guest_hd -R "ls /" | grep test
 33  (12) usr    2061  (12) var    169  (3812) test
```

# Container can Modify Guest Hard Disk

- **Did we breakout?**
- Not so fast
  - Page cache and dentry cache
- Device-level changes may not be seen by guest processes!



# Page & Dentry Cache

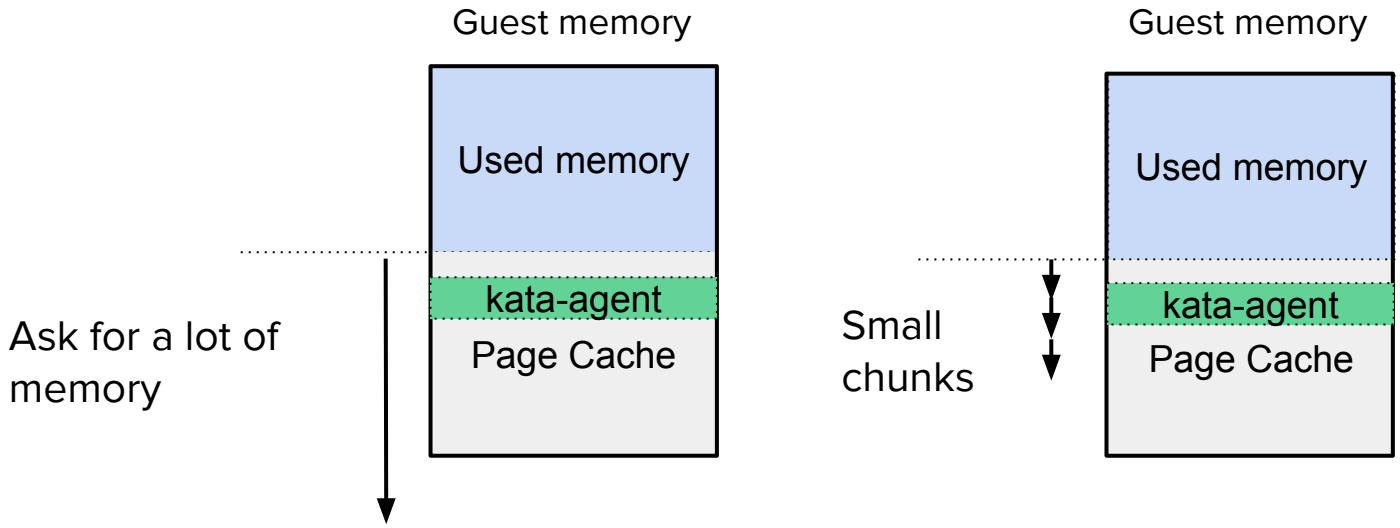


# Gaining Execution on Guest

- Guest is static - need to replace a running executable
  - kata-agent, systemd
  - But those are already loaded to the page cache
- **Force the guest kernel to free the page cache**

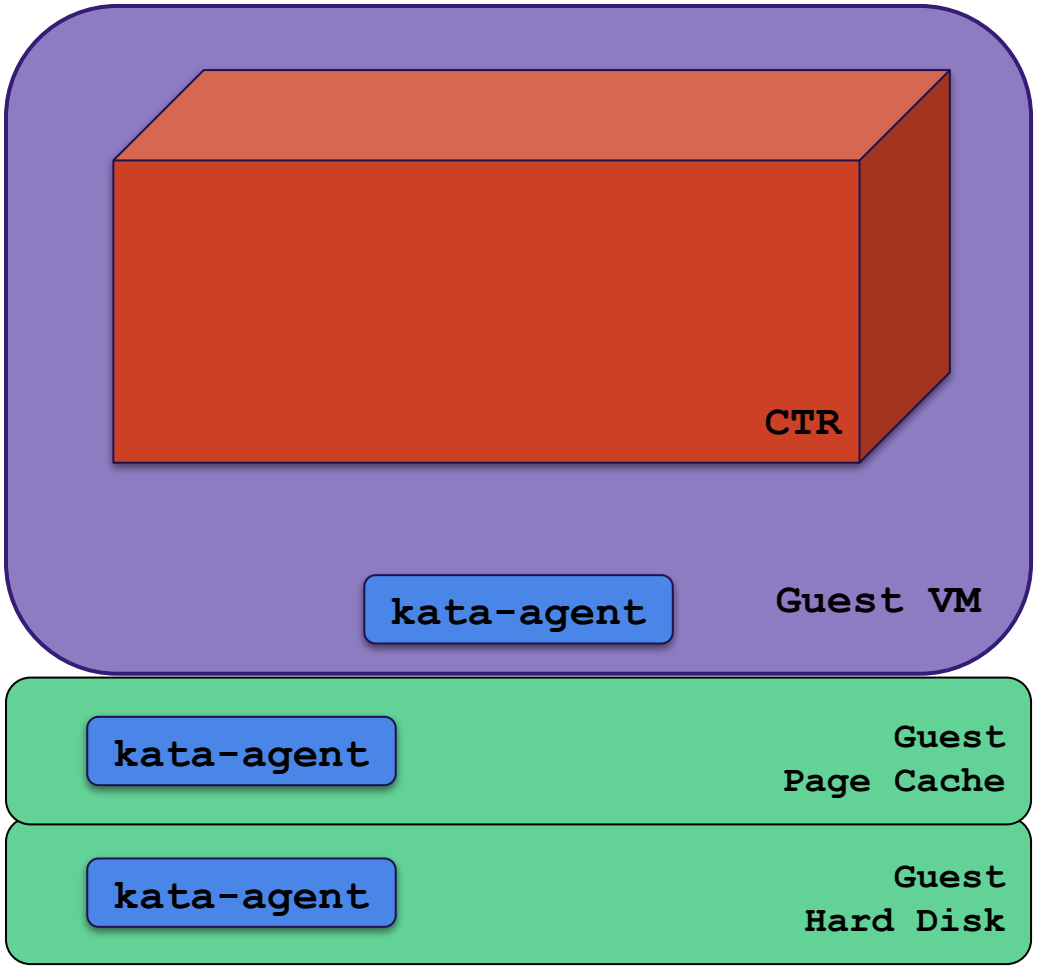
# Gaining Execution on Guest - Freeing Cache

- Container allocates small chunks of memory

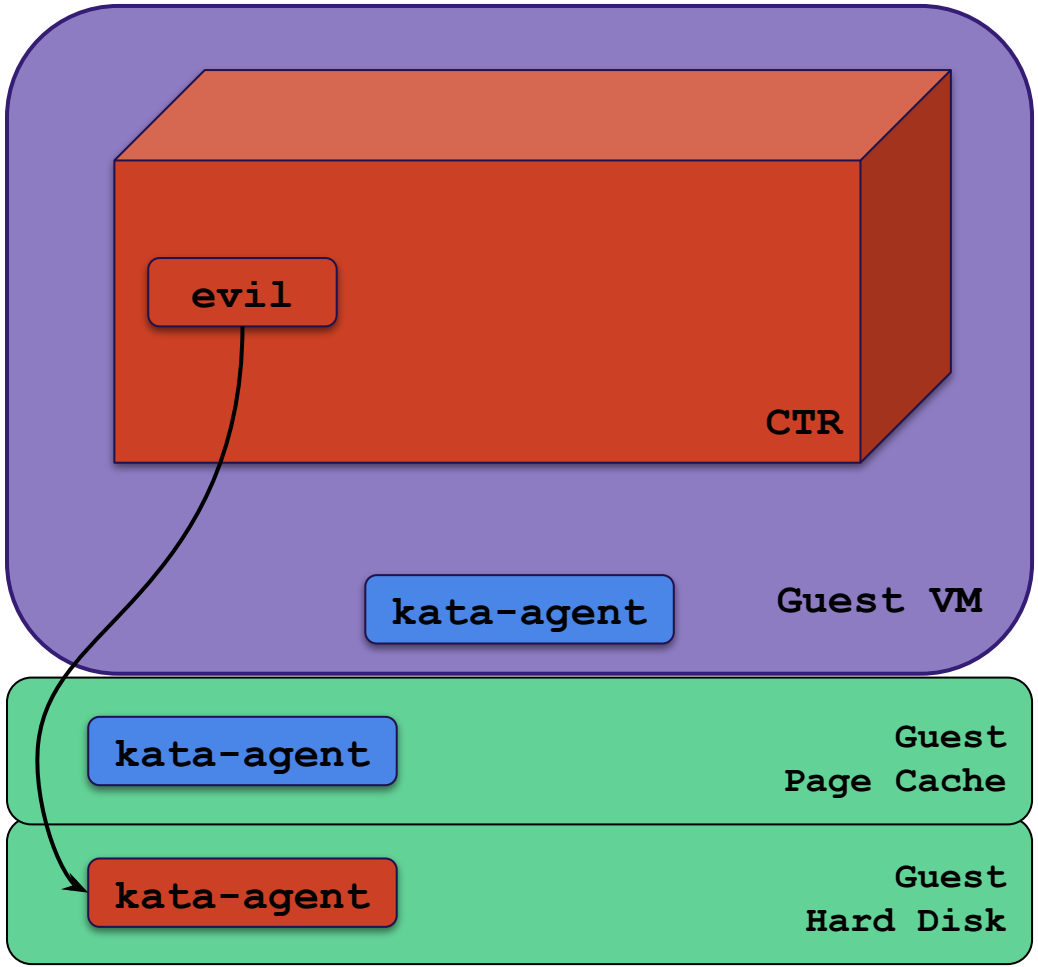


# Container-to-Guest Attack

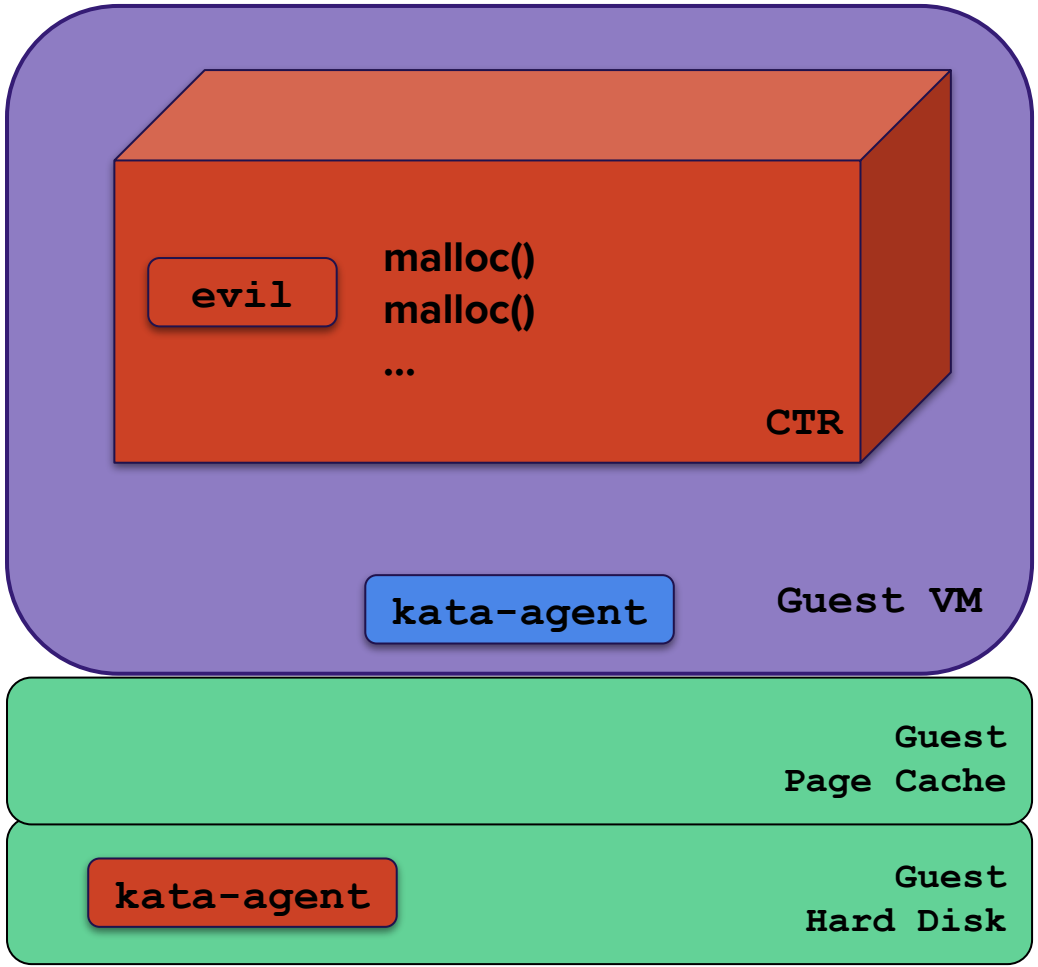
Our malicious container runs under Kata



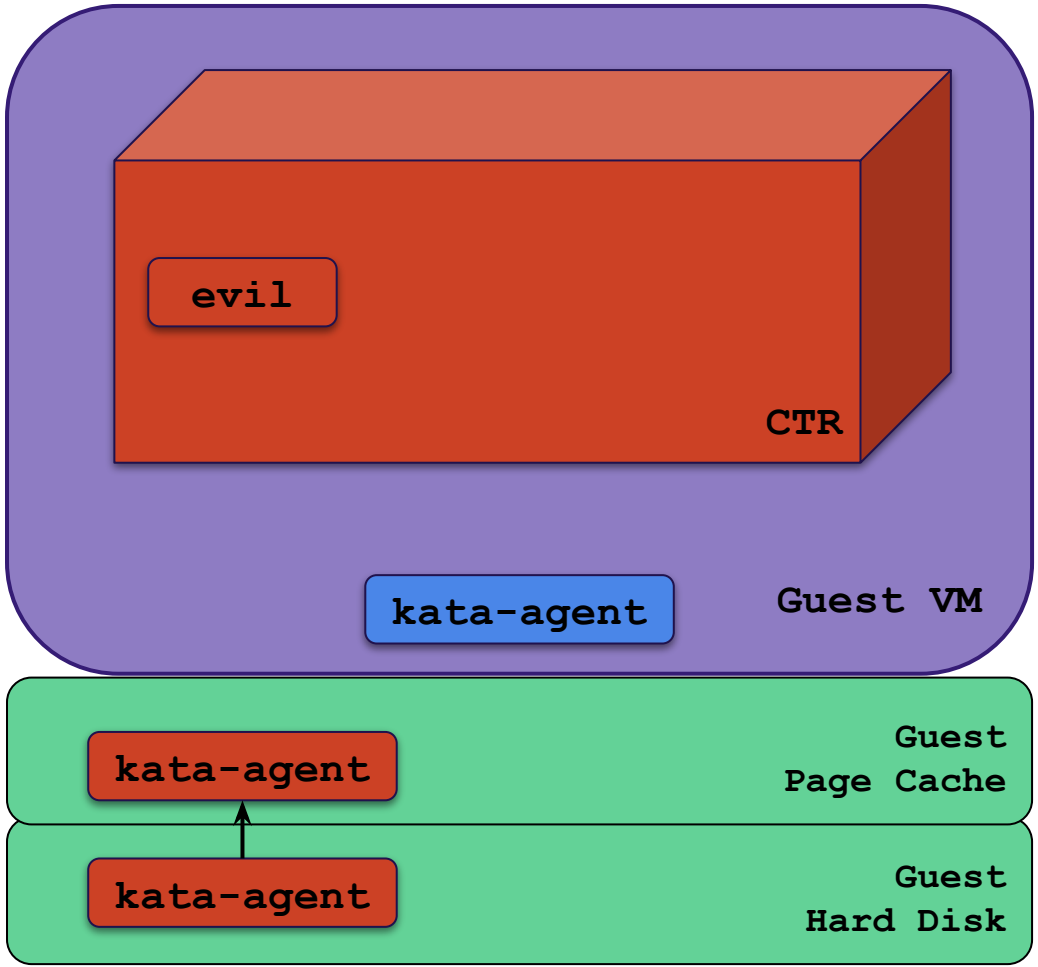
1. Container overwrites kata-agent on hard disk with malicious binary



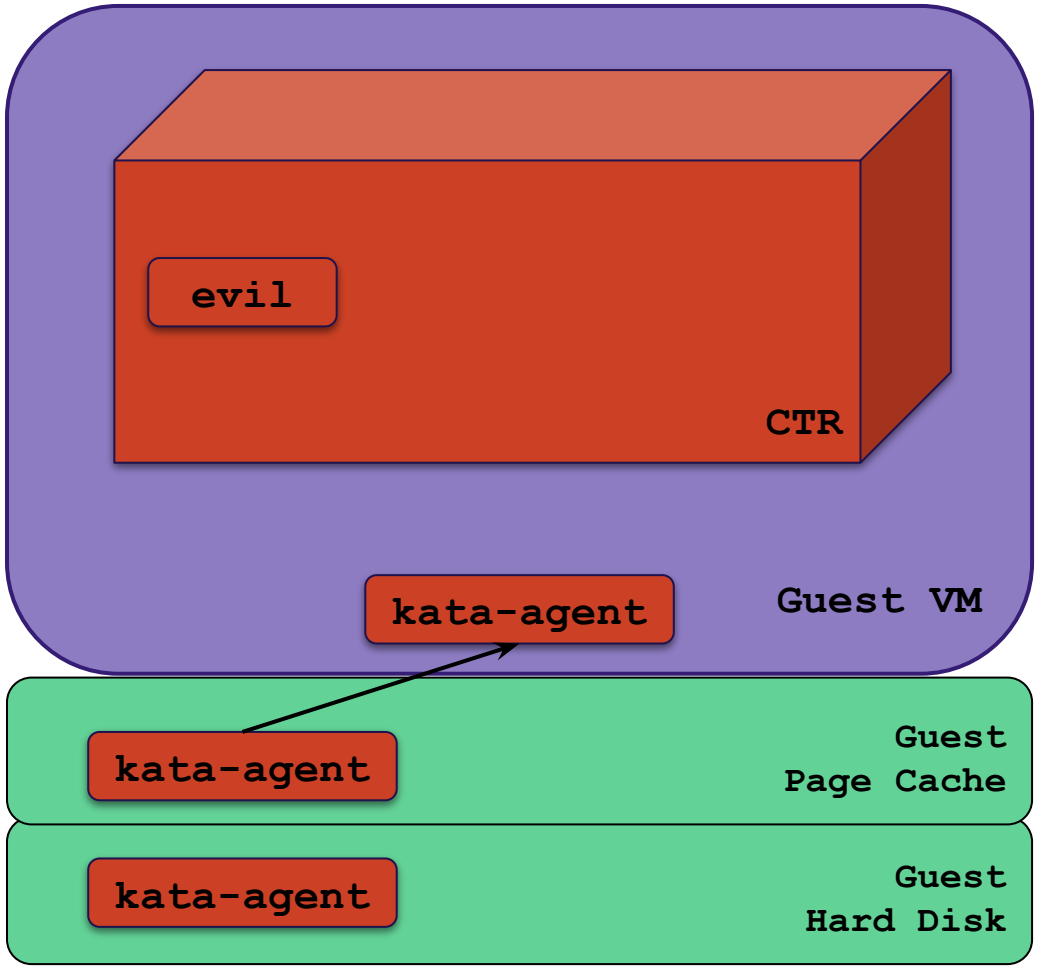
2. Container allocates small chunks to **clear kata-agent from page cache**



3. Execution passes back to kata-agent, kernel must read kata-agent binary from disk

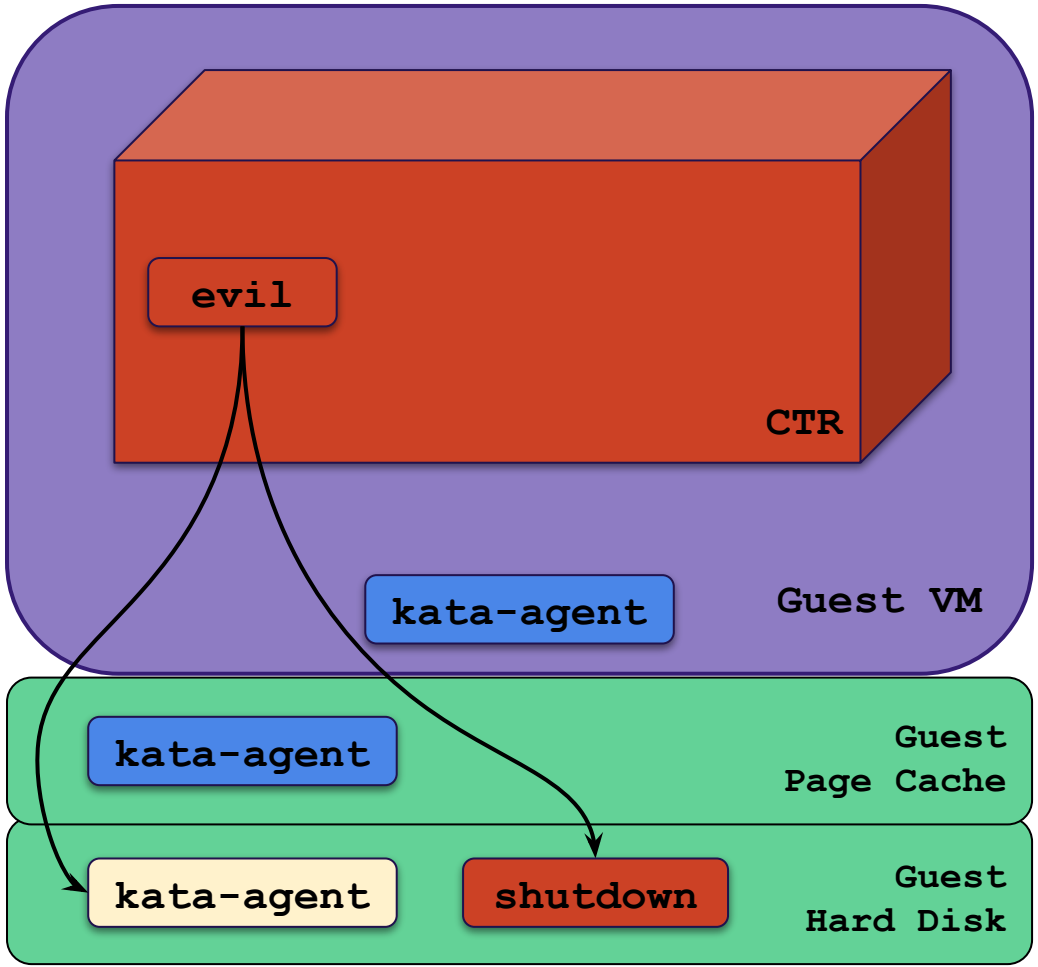


4. Kata-agent process now maps to our malicious binary

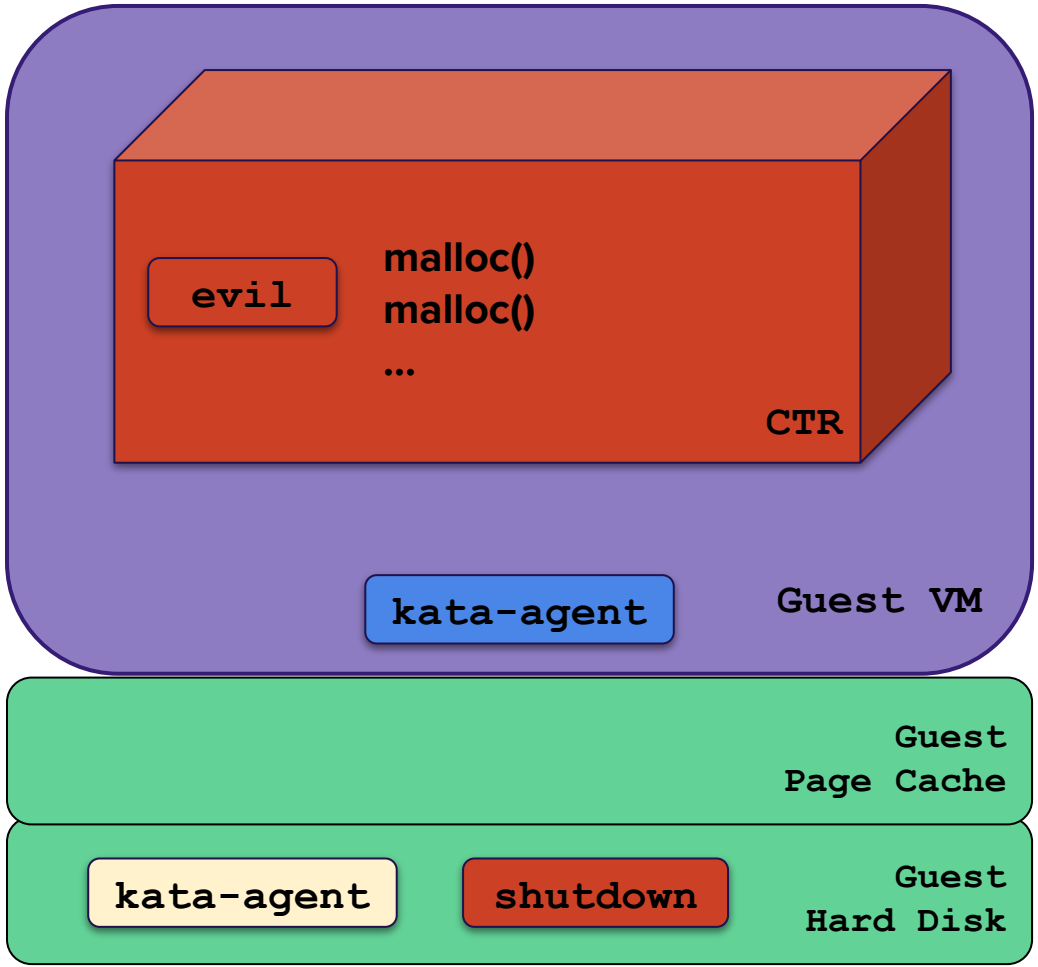


# Replacing a process binary mid-execution is tricky!

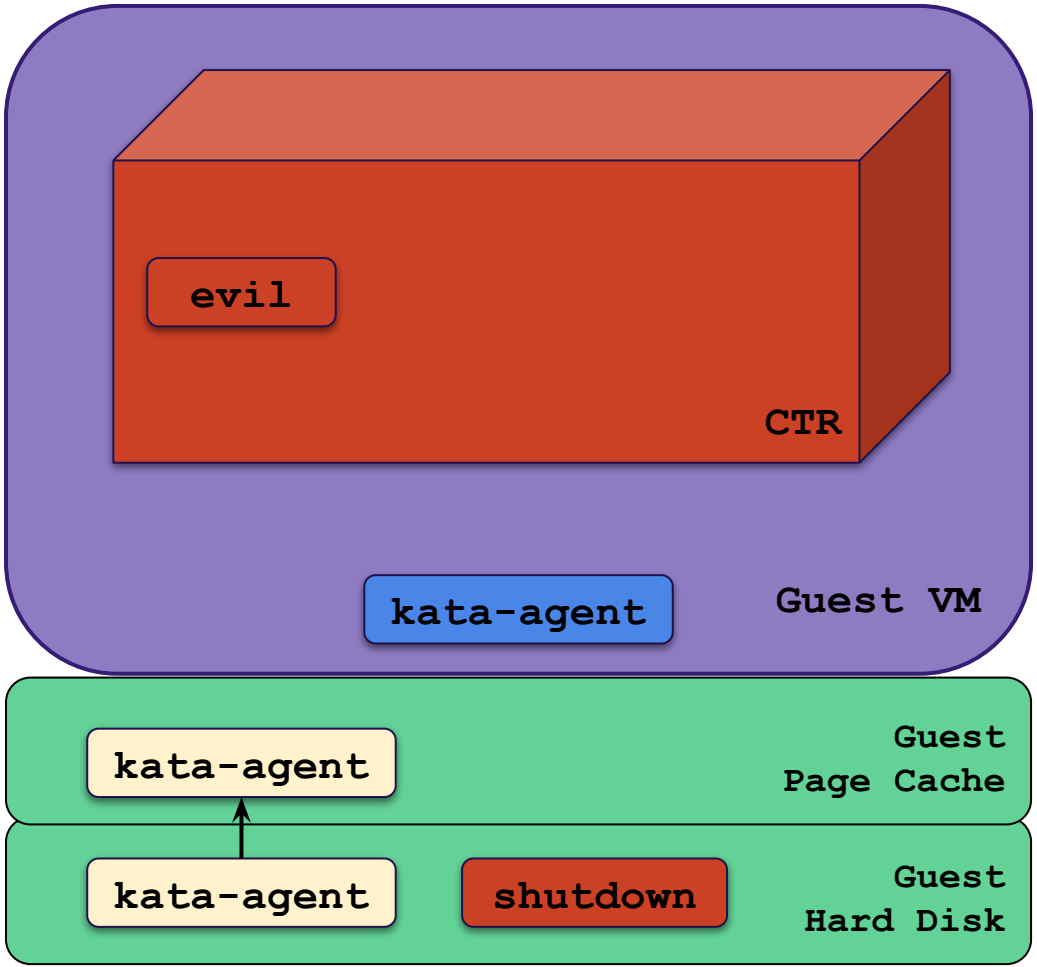
1. Container overwrites kata-agent on hard disk with **garbage data**
2. Container overwrites a non-cached binary -e.g. **systemd-shutdown**



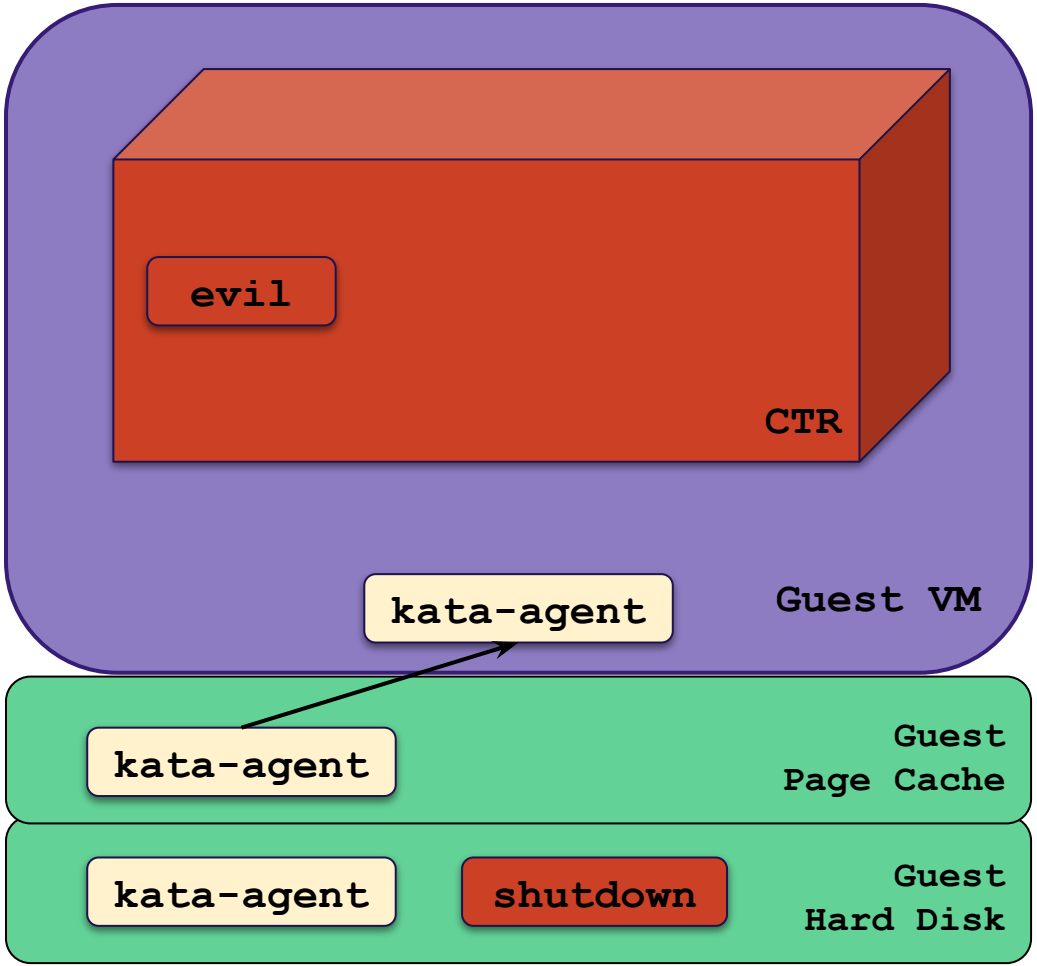
3. Container allocates small chunks to **clear kata-agent from page cache**



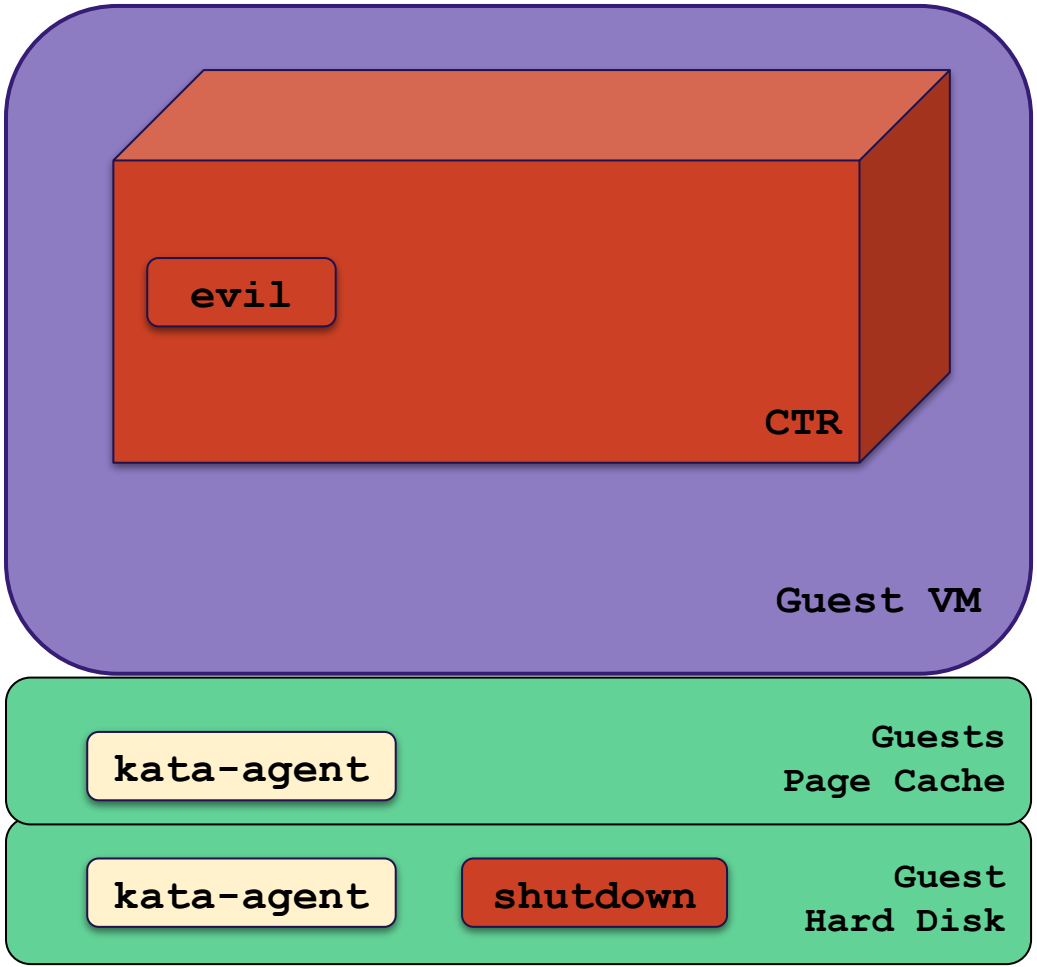
4. Execution passes back to kata-agent, kernel must read kata-agent binary from disk



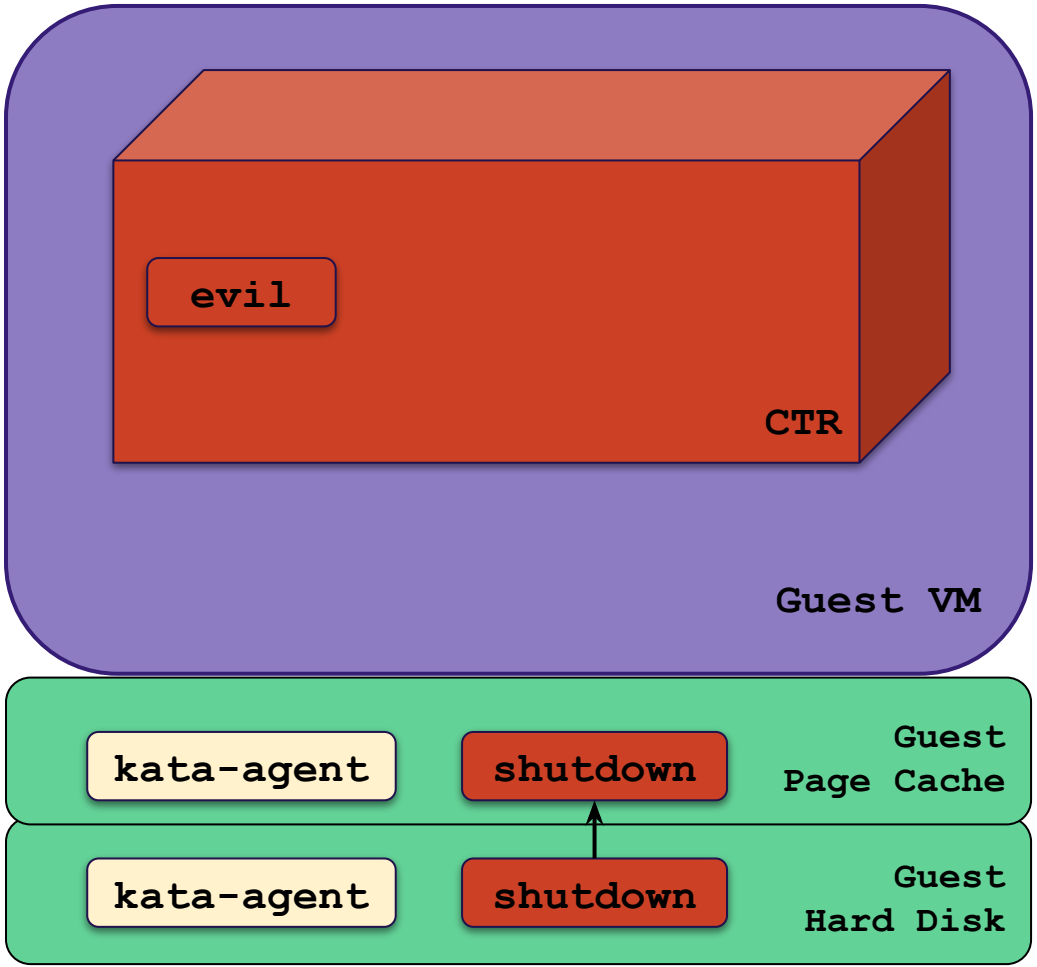
5. kata-agent process now maps to garbage data, and crashes



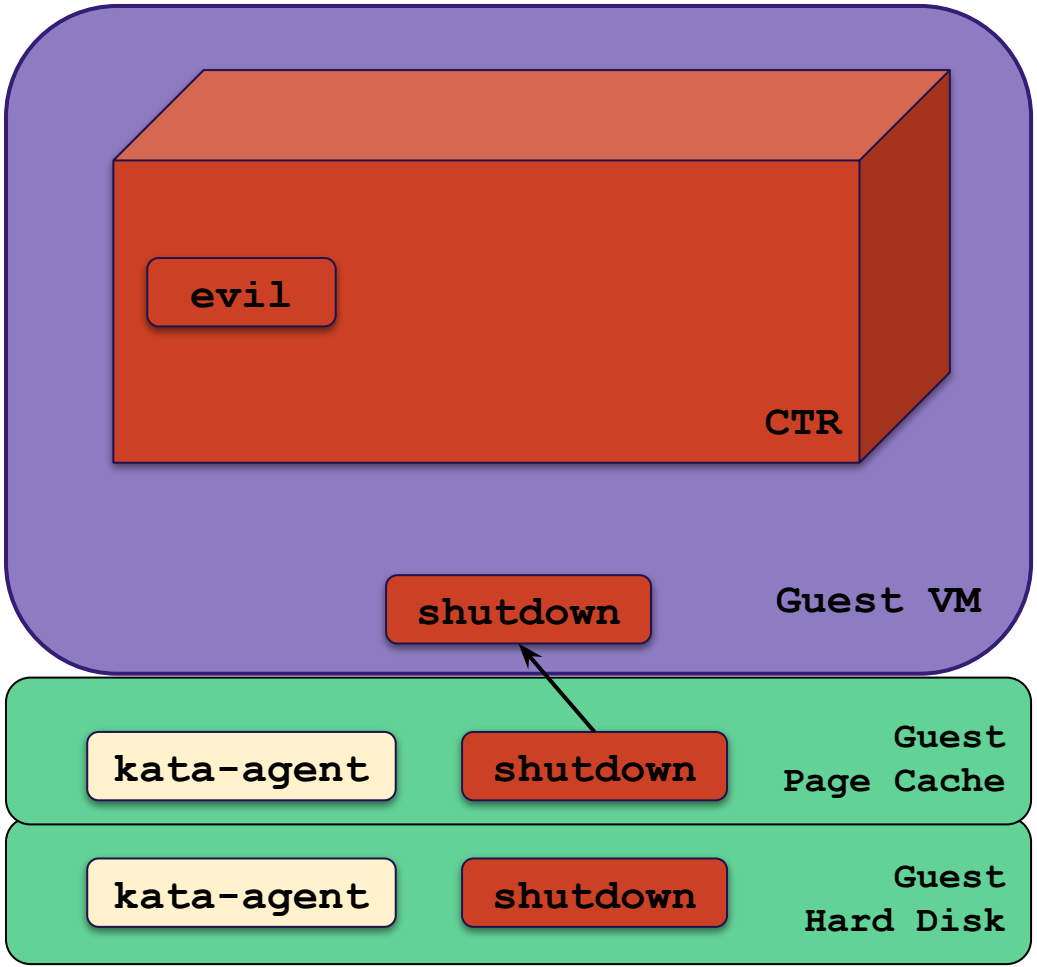
6. A shutdown sequence is started, calling systemd-shutdown



6. A shutdown sequence is started, calling systemd-shutdown



7. Our malicious systemd-shutdown runs on the guest as root!



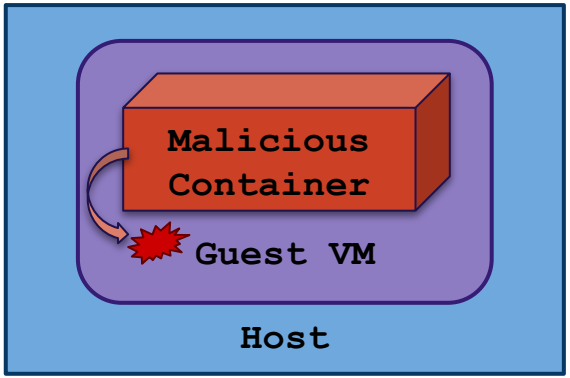
# PoC

- Malicious `systemd-shutdown` will create a `guest-is-now-malicious` file in shared dir

**Demo:**
container-to-guest
escape

# Container-to-Guest Breakout

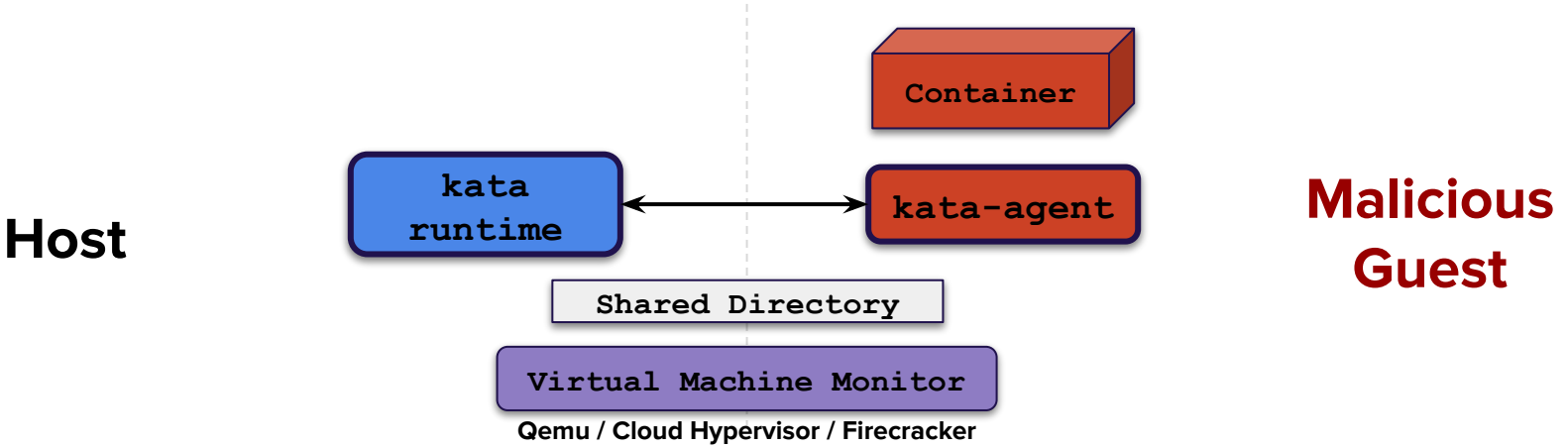
- Breakout technique exploiting direct device access
  - If you modify a container's config, you better be adding restrictions
- Container needs `CAP_MKNOD`
  - Default in docker & k8s+containerd, not in k8s+crio
- CVE-2020-2023



# Escaping the VM

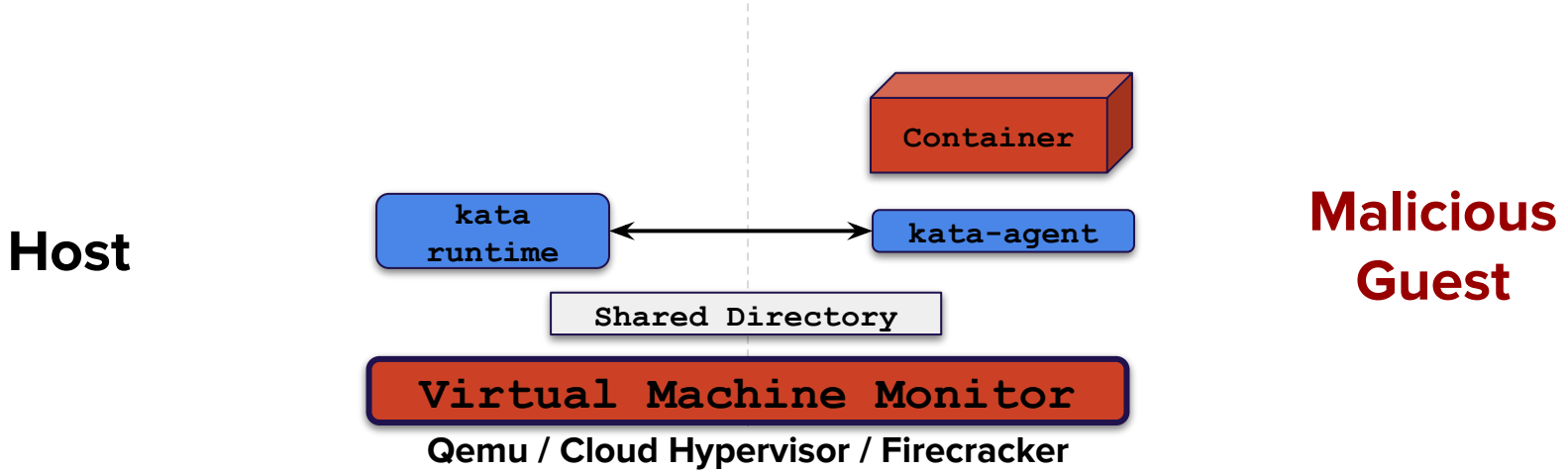
# VM Attack Surface

- kata-runtime parsing of kata-agent msgs



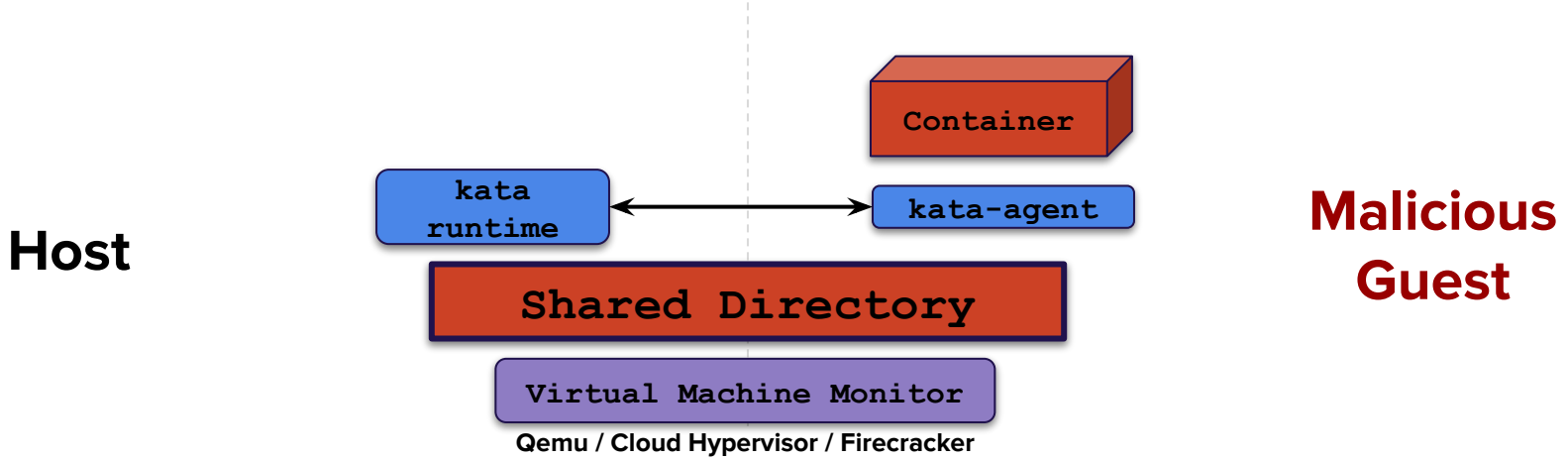
# VM Attack Surface

- kata-runtime parsing of kata-agent msgs ✖
- **Issue with a VMM**



# VM Attack Surface

- kata-runtime parsing of kata-agent msgs
- Issue with a VMM (to be continued...)
- **Shared directory between the host & guest**



# Shared Dir Attack Surface

- kata-runtime (host) operates on files in shared dir
- **Guest can control as much as host**
- Used to deliver the image to the guest 

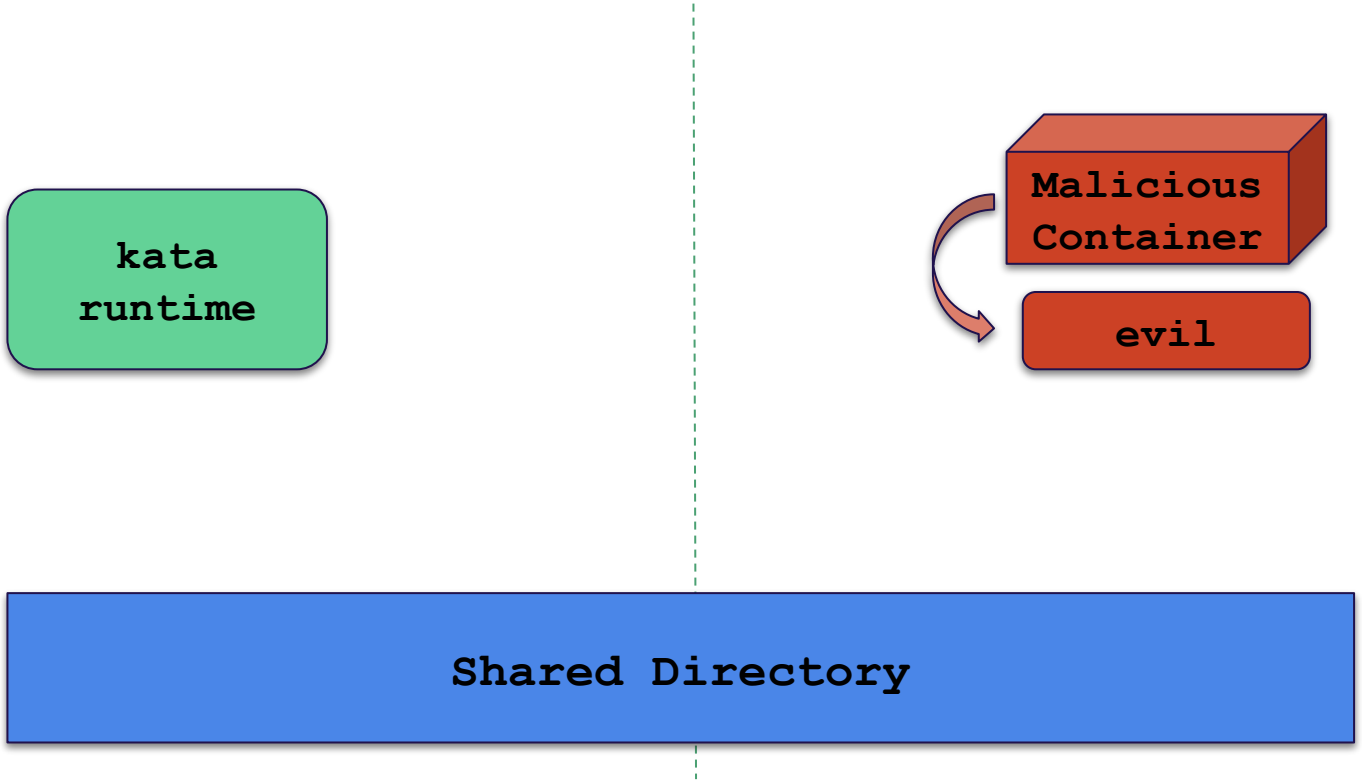
# Shared Dir Attack Surface

1. kata-runtime **bind-mounts ctr image** to shared dir
    2. Container starts
    3. Container terminates
4. kata-runtime **unmounts ctr image** from shared dir

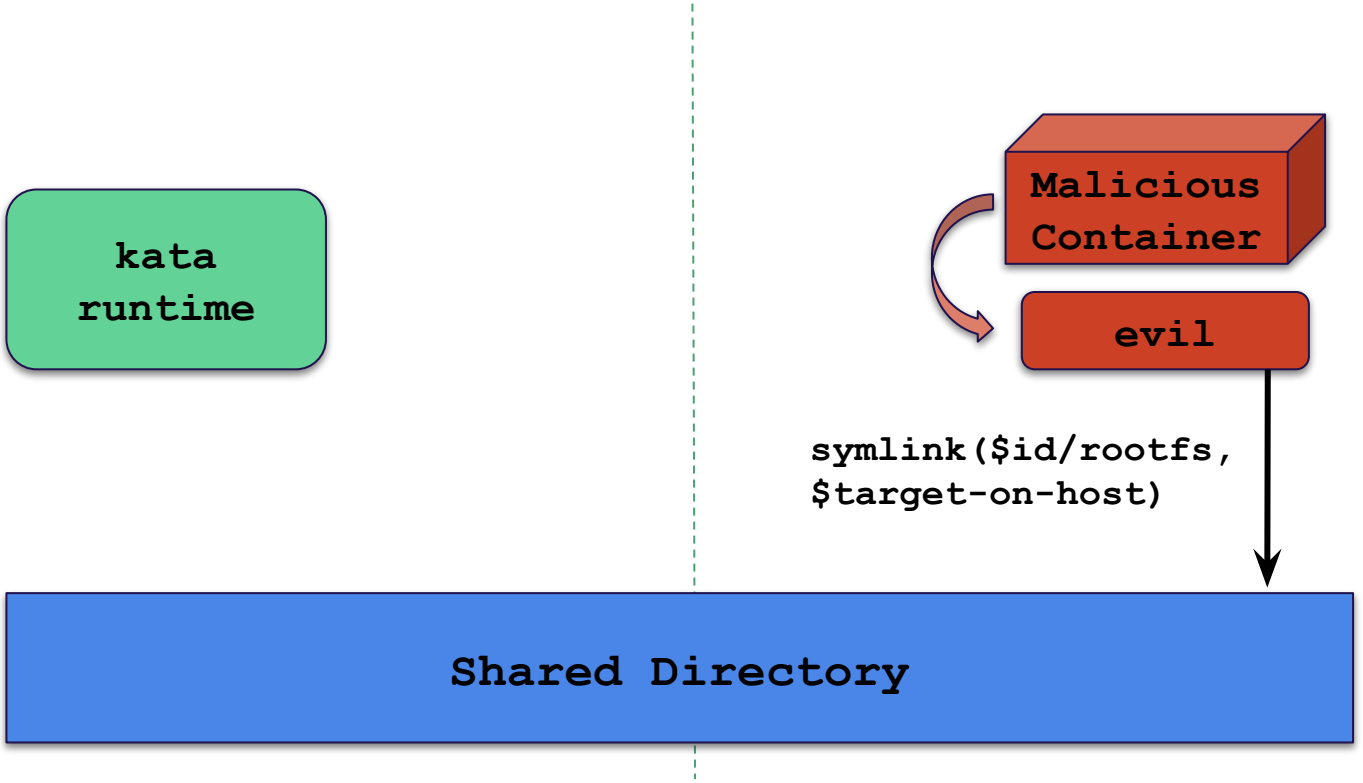
**Both mount and unmount follow symlinks!**

# Unmount Redirection Guest-to-Host DoS

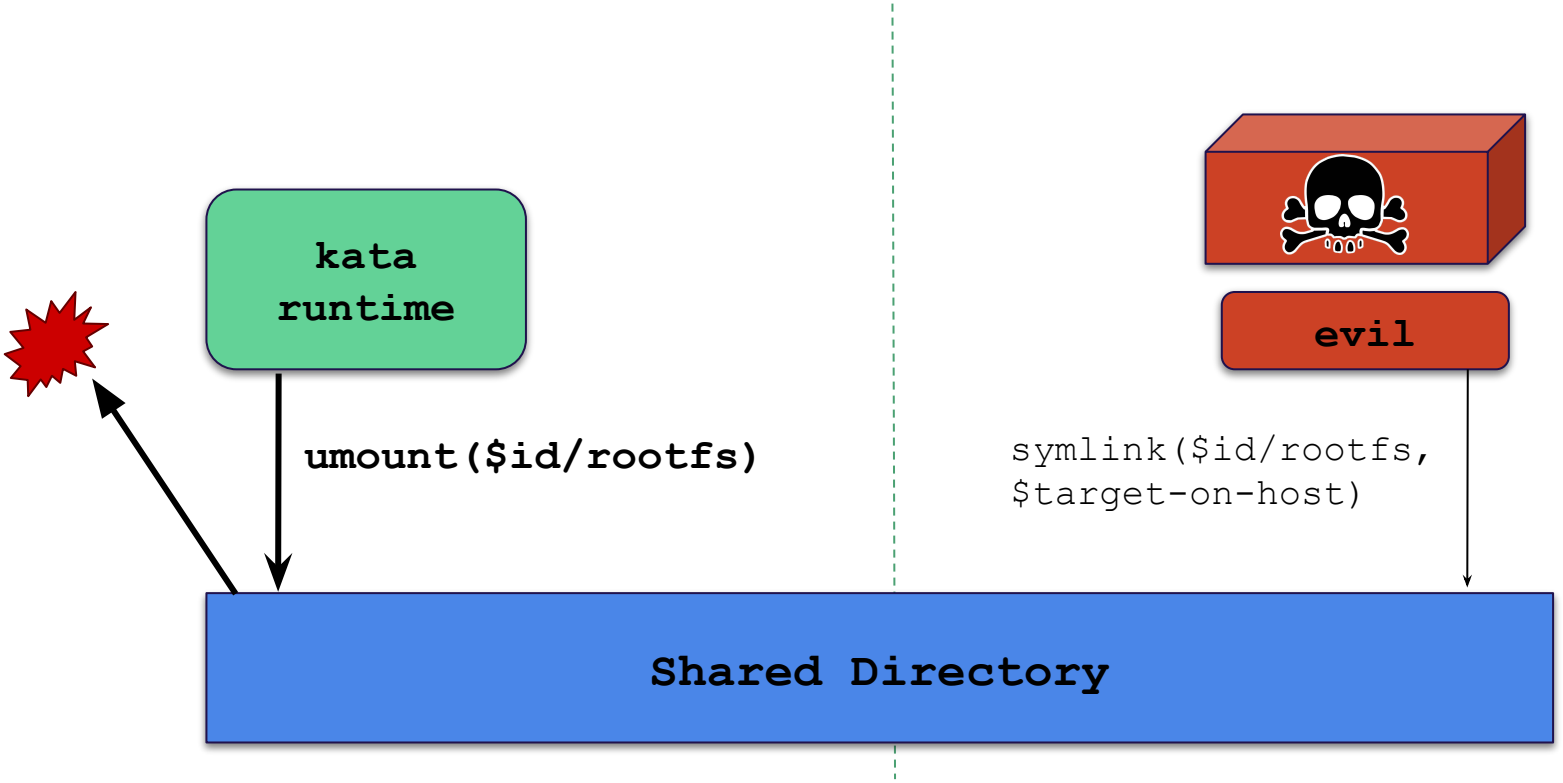
# Unmount Redirection Attack



# Unmount Redirection Attack



# Unmount Redirection Attack



# Unmount - Guest-to-Host DoS

- Targeting ‘/’ unmounts all mount points underneath it
  - `/proc, /sys, /dev, /tmp`
- Host is unusable, can no longer run containers
- CVE-2020-2024

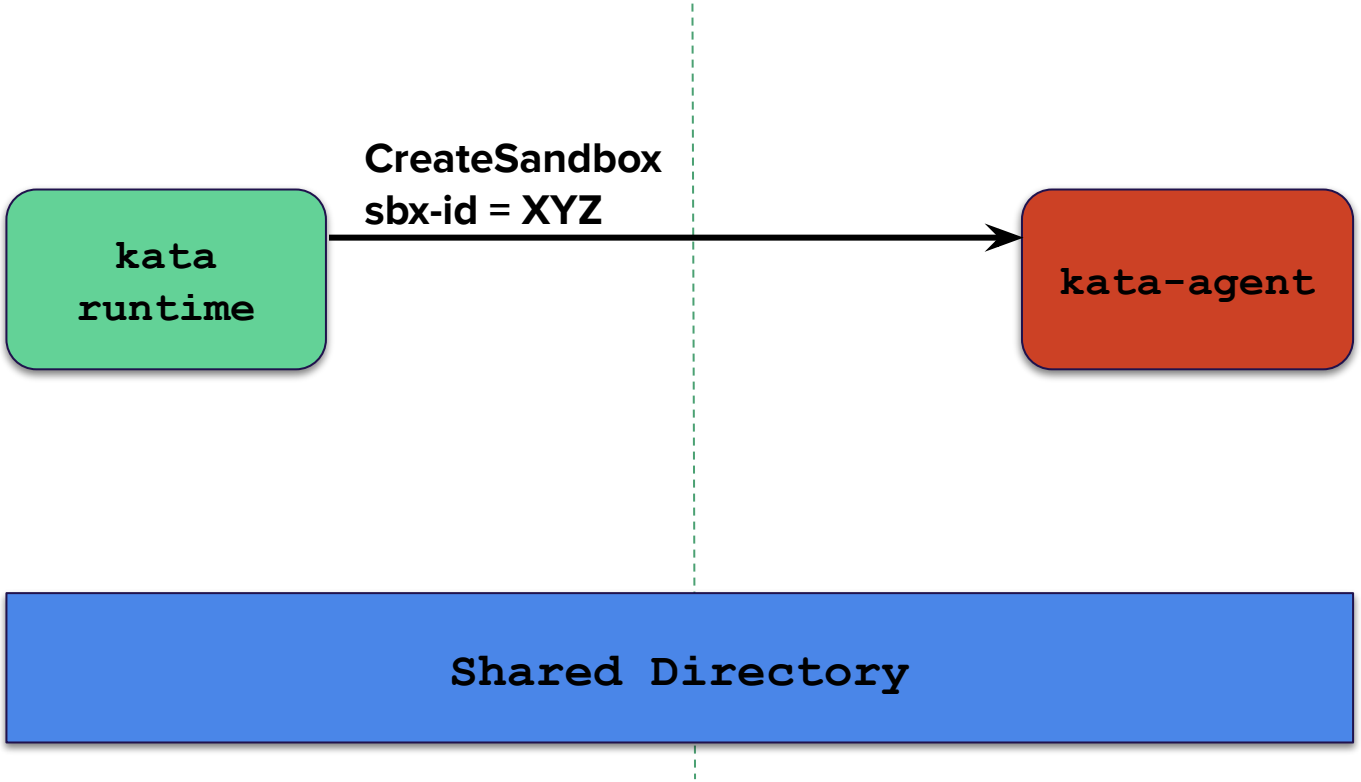
# **Demo**

Guest-to-Host DoS

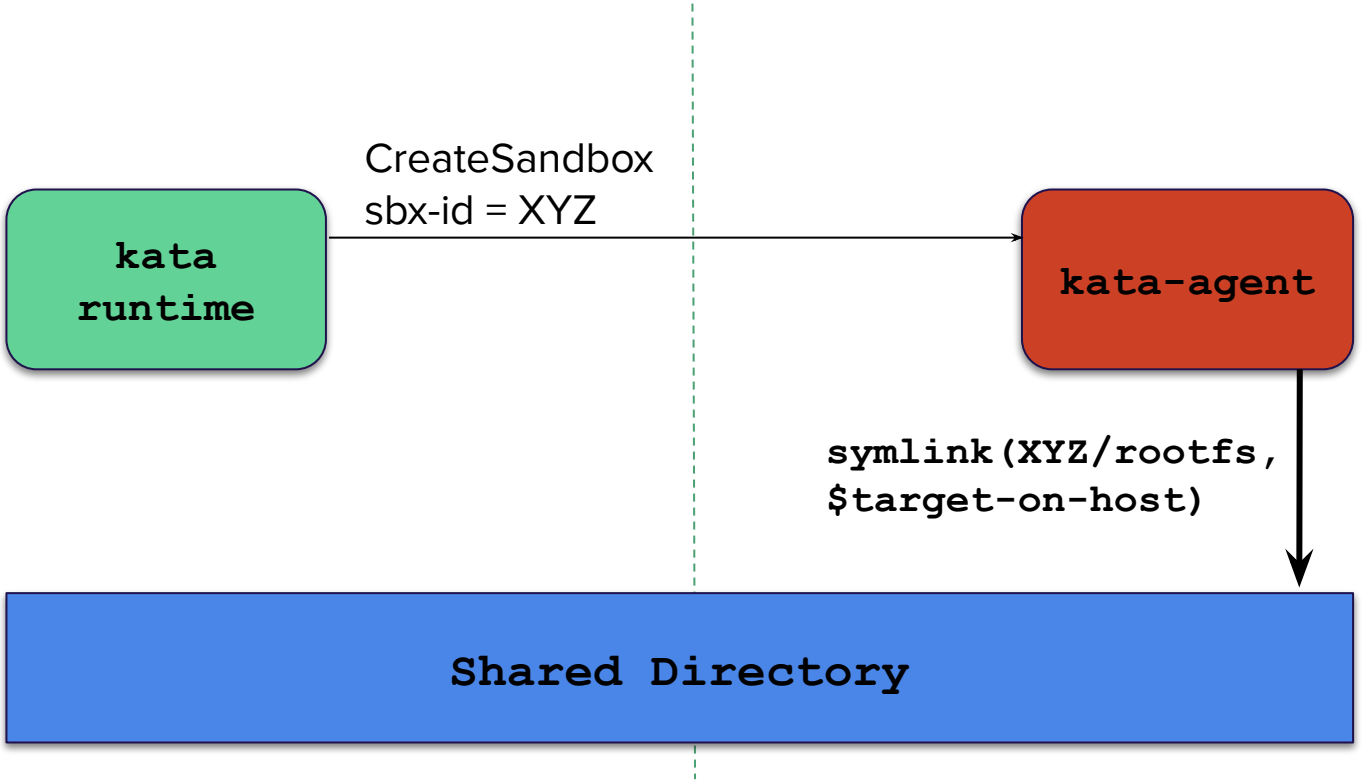
# **Image Mount Redirection**

Guest-to-Host RCE

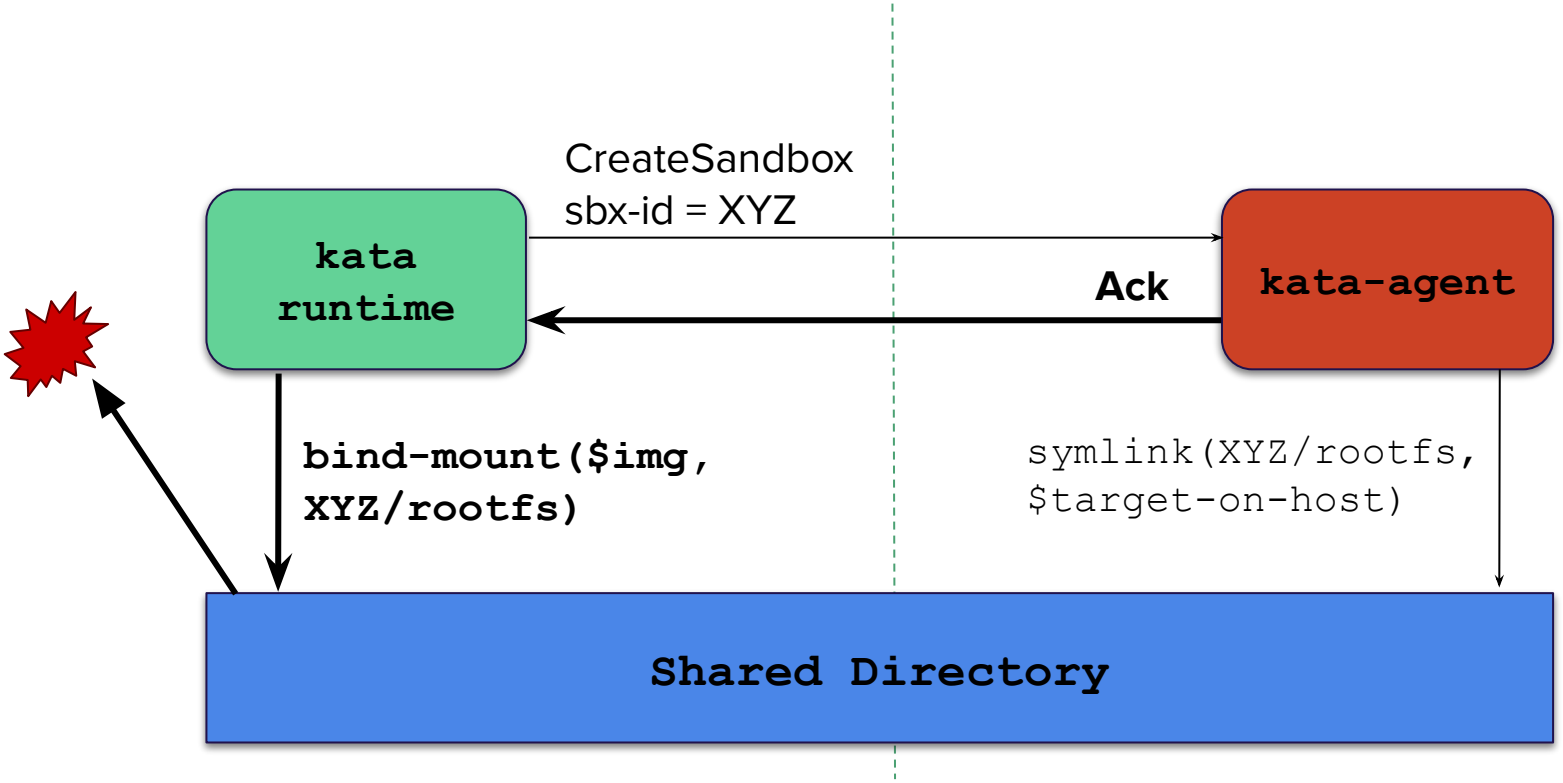
# Mount Redirection Attack



# Mount Redirection Attack



# Mount Redirection Attack



# Mount Redirection Attack

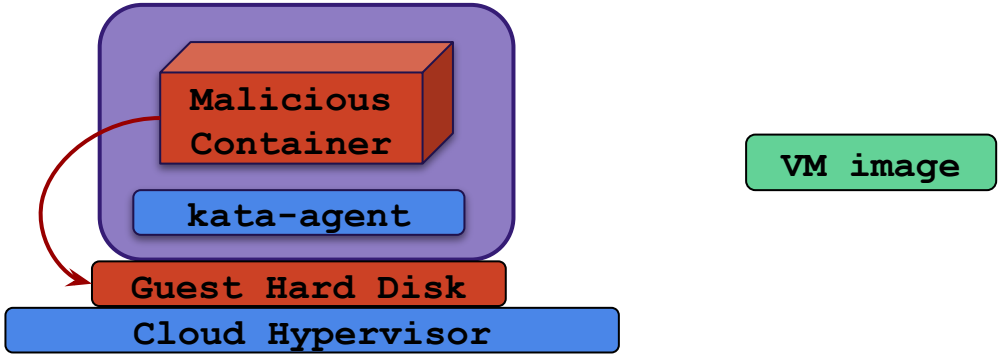
# Cloud Hypervisor

# Cloud Hypervisor (CLH)

- One of the the 3 VMMs options
- Kata didn’t work after container-to-guest PoC on CLH
- Inspected VM image, **kata-agent had garbage data!**

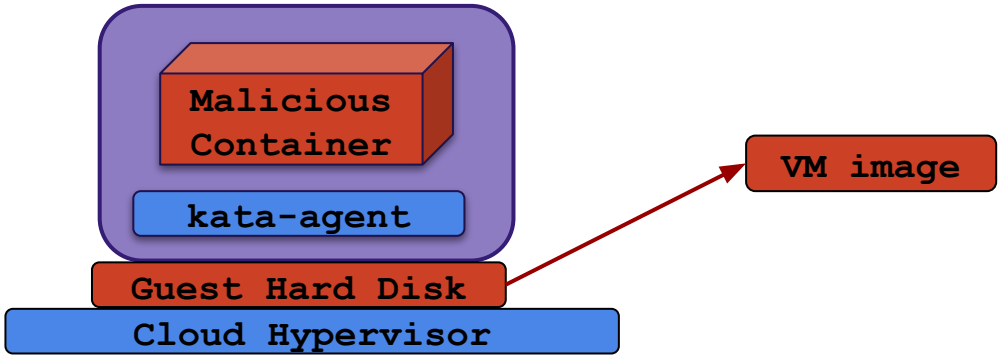
# Cloud Hypervisor (CLH)

- One of the the 3 VMMs options
- Kata didn’t work after container-to-guest PoC on CLH
- Inspected VM image, **kata-agent had garbage data!**



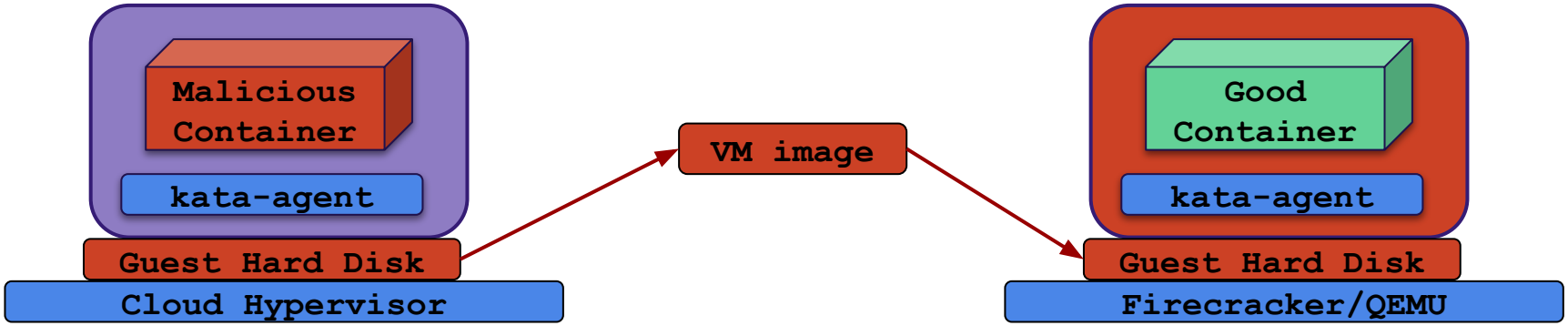
# Cloud Hypervisor

- One of the the 3 VMMs options
- Kata didn’t work after container-to-guest PoC on CLH
- Inspected VM image, **kata-agent had garbage data!**



# Guest-to-Future-Guests RCE (CVE-2020-2025)

- Kata+CLH commits guest HD changes to VM image
- A malicious guest can **control all future sandboxes**!
  - By defaults, all VMMs use the **same VM image**
- **That's bad for multi-tenancy**

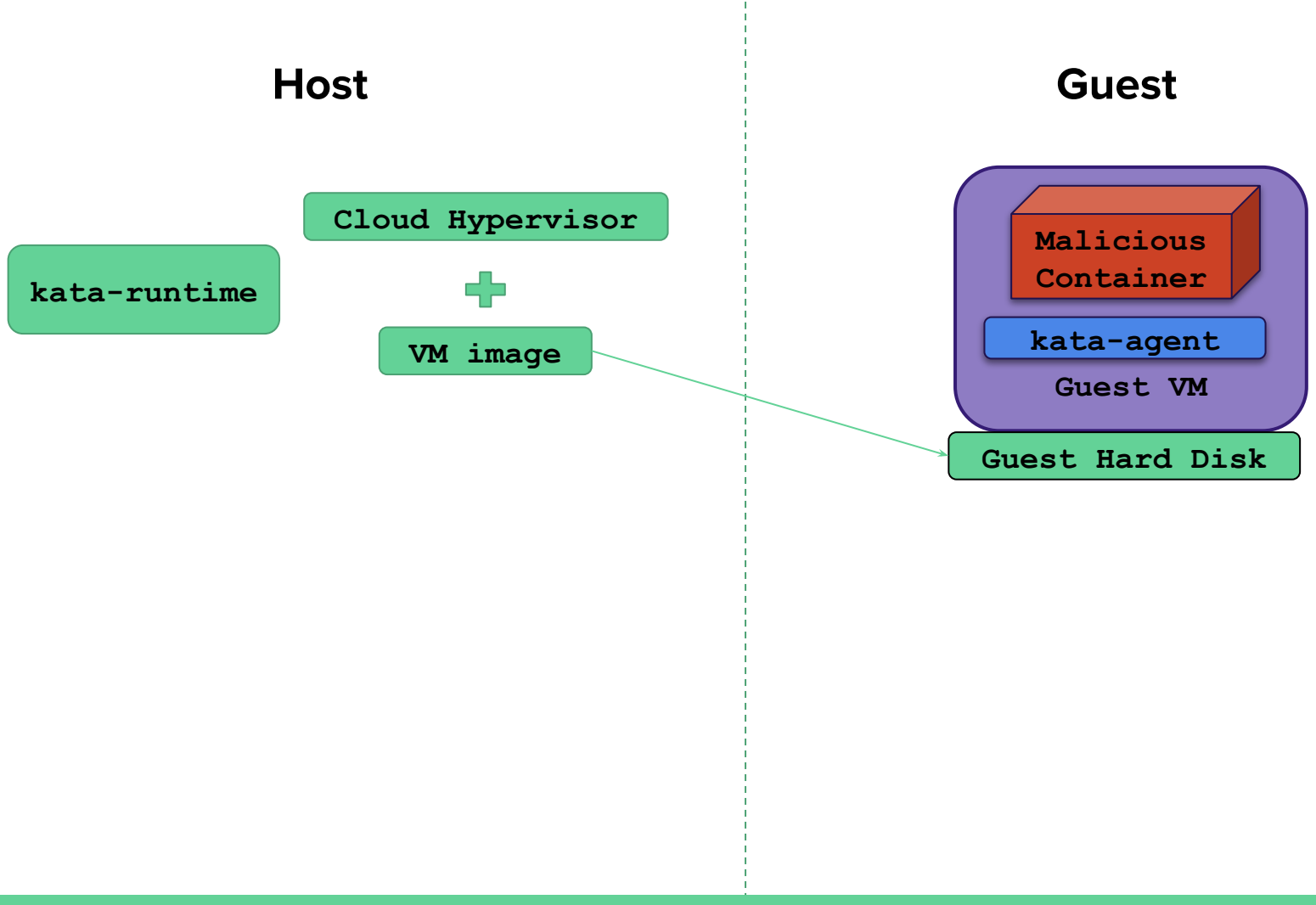


# Back to Redirecting Image Mount

- Guest needed to be compromised before ctr runs
- Malicious VM image = Guest malicious from boot
  - Can create the malicious symlink!

# Container-to-Host Code Execution

CVE-2020-2023 (Container-to-Guest)
CVE-2020-2025 (CLH commits to VM image)
CVE-2020-2026 (Mount Redirection)


Host
Guest
Cloud Hypervisor
kata-runtime
VM image
Malicious Container
kata-agent
Guest VM
Guest Hard Disk

## Host

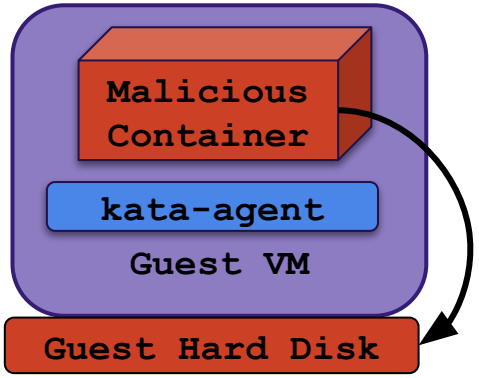
kata-runtime

Cloud Hypervisor

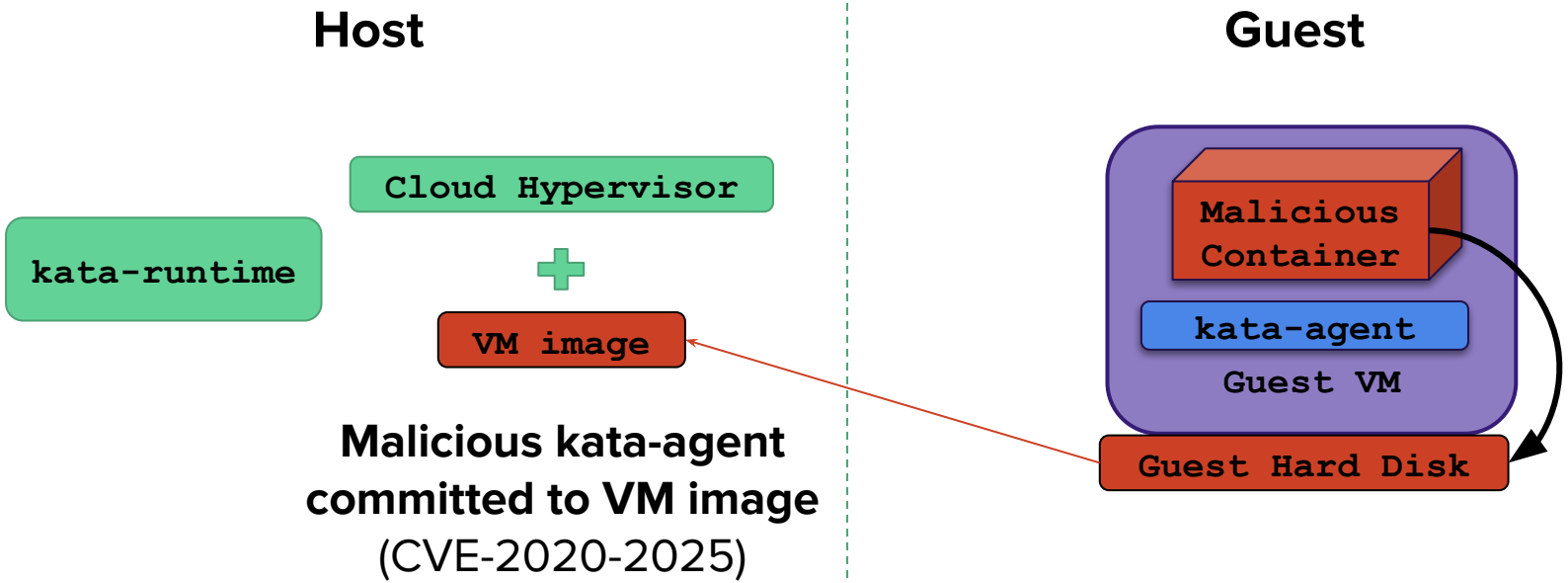
+

VM image

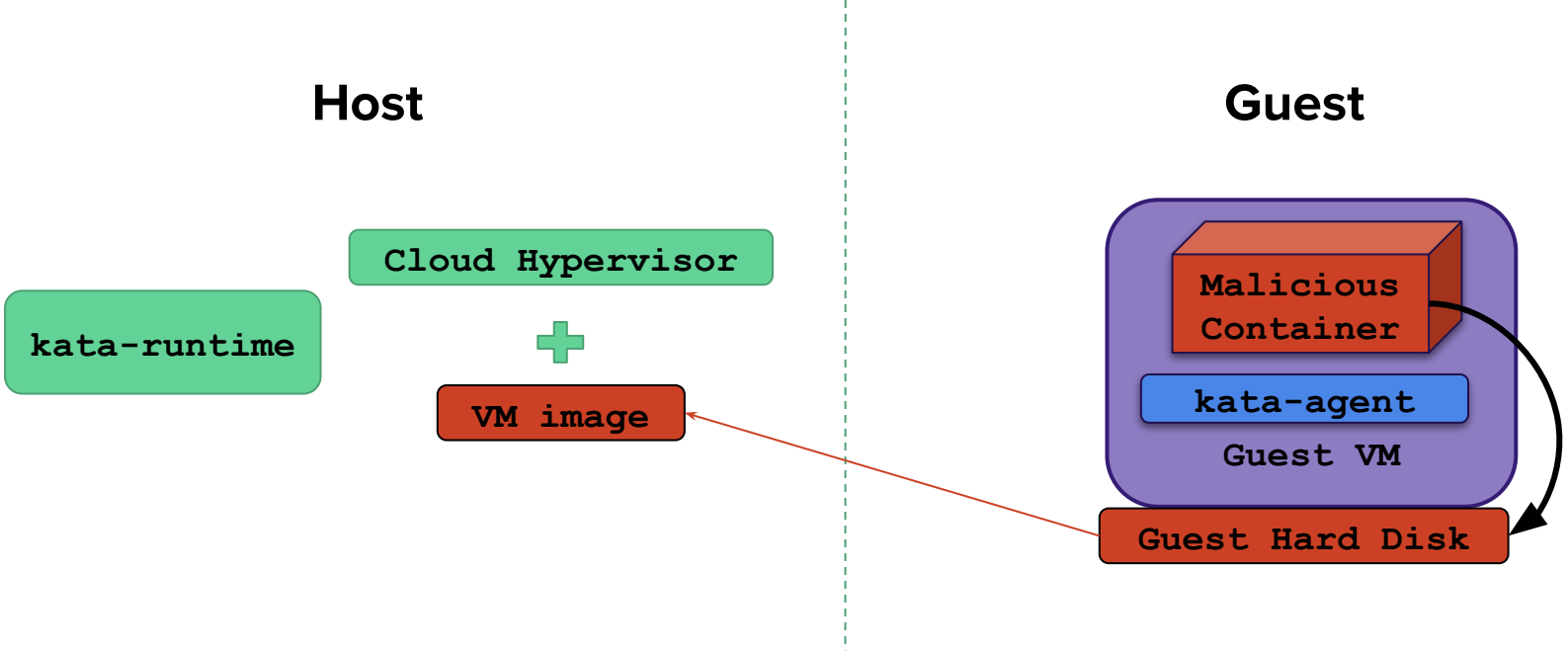
## Guest




**Container overwrites kata-agent binary on disk**
(CVE-2020-2023)

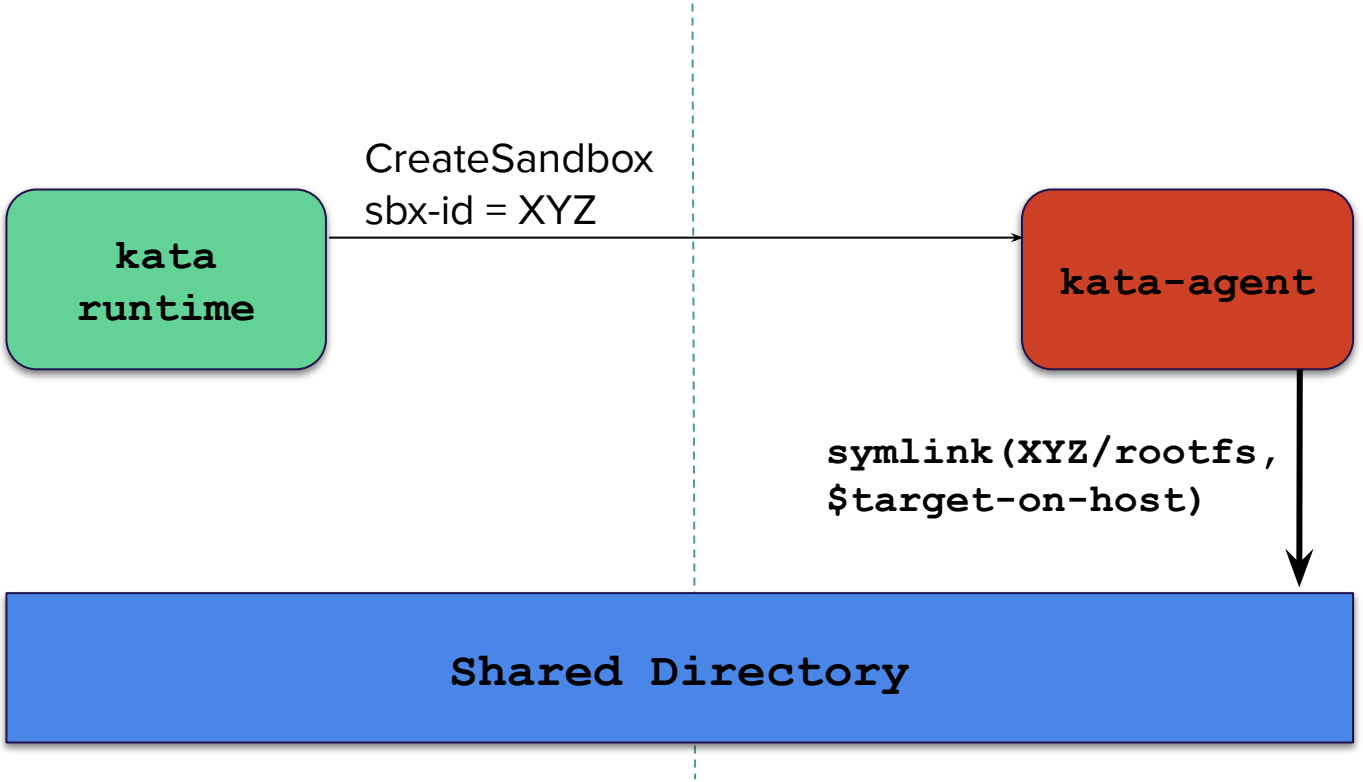



**Malicious kata-agent committed to VM image**
(CVE-2020-2025)

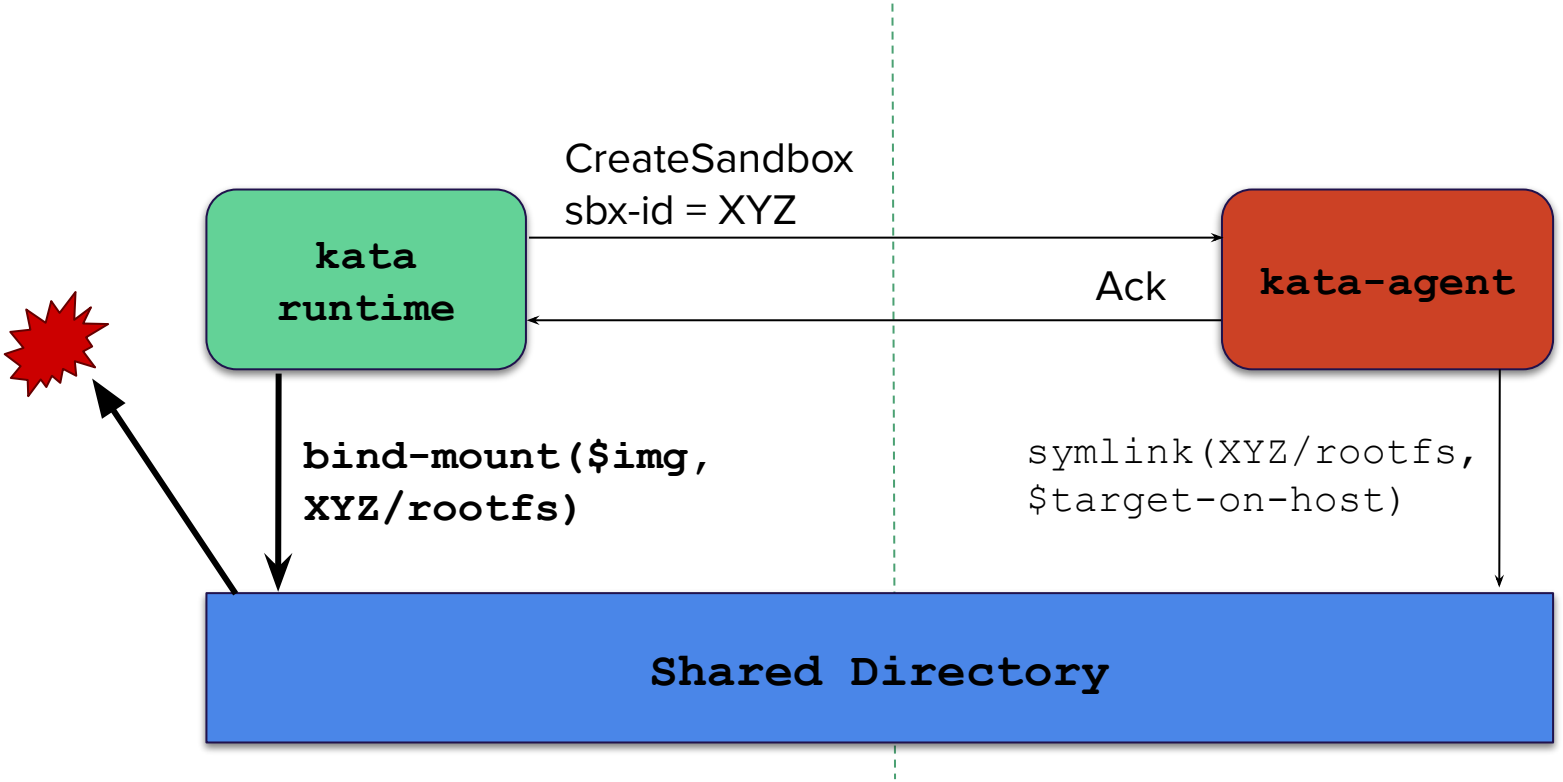



**Next time the malicious container is run, the guest runs a our evil kata-agent**

# Second Guest Redirects Mount



# Second Guest Redirects Mount



# **Demo**: Container-to-Host Code Execution

CVE-2020-2023 (Container-to-Guest)
CVE-2020-2025 (CLH commits to VM image)
CVE-2020-2026 (Mount Redirection)

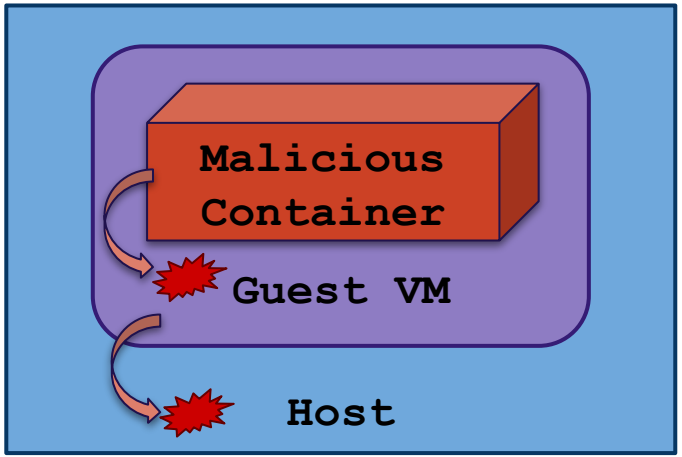
# Got Code Execution on Host!

# Shared Directory is a Big Attack Surface

- Issues with host apps using it
  - mount & unmount redirection (CVE-2020-2024/[illegible])
- Vulnerabilities within the mechanism itself
  - Virtio-fs daemon - Ctr-to-Host DoS (CVE-2020-10717)

# Summary of Vulnerabilities

- Container to Guest, device access, RCE
- Guest to Host, umount, DoS
- Guest to future Guests on CLH, RCE
- Guest to Host, mount, RCE
- Container to Host, virtio-fsd, DoS



# Disclosure

- All issues were responsibly disclosed and fixed by Kata Containers maintainers
  - CVE-2020-2023, CVE-2020-2024, CVE-2020-2025, CVE-2020-2026
  - Read more at https://github.com/kata-containers/community/tree/master/VMT/KCSA

# Takeaways

- Containers are only as secure as their configuration
  - Apply best practices
    - ➢ User namespaces / run as non-root
  - Drop unused privileges
    - ➢ **Who really needs `CAP_MKNOD`?**
- Sandboxes limit the attack surface, but aren’t magic
  - Enhancement, not a replacement
  - Enable security features
    - ➢ Kata integrity checks

# Questions?

# DAX (simplified)