Container Live Migration

Adrian Reber

2020, June 30

Red Hat Blog:

Container migration with Podman on RHEL

https://www.redhat.com/en/blog/container-migration-podman-rhel

Agenda

Use cases

Details

Demos

Future

Definition:

Container Live Migration

Transfer Running Container

Serialize on Source System

Transfer to Destination System

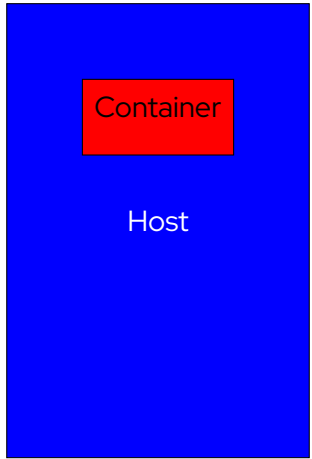
Checkpoint/Restore in Userspace

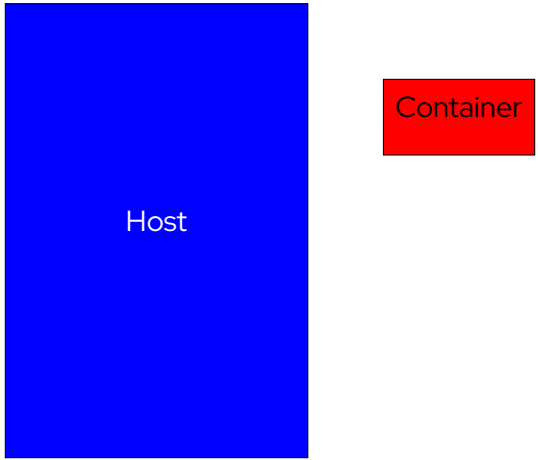
CRIU

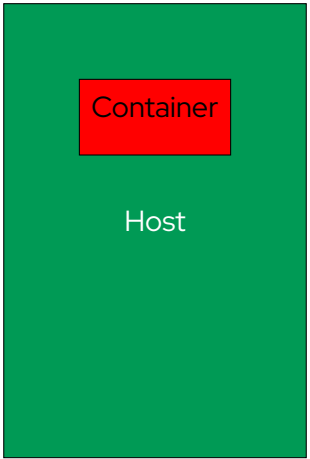
Multiple Integrations Exist

Use Cases

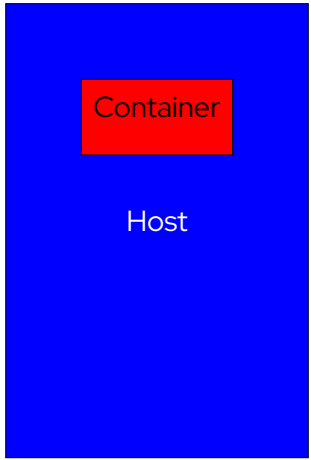
Reboot and Save State

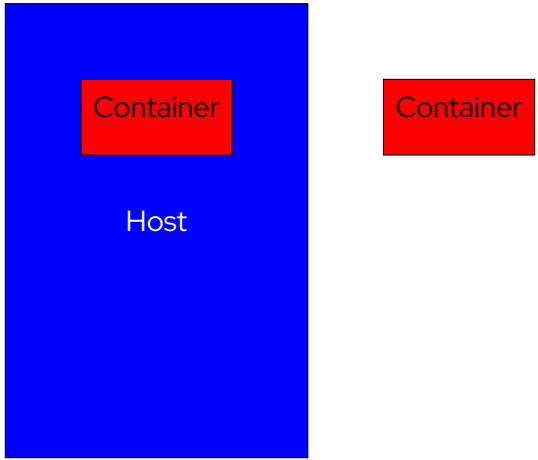


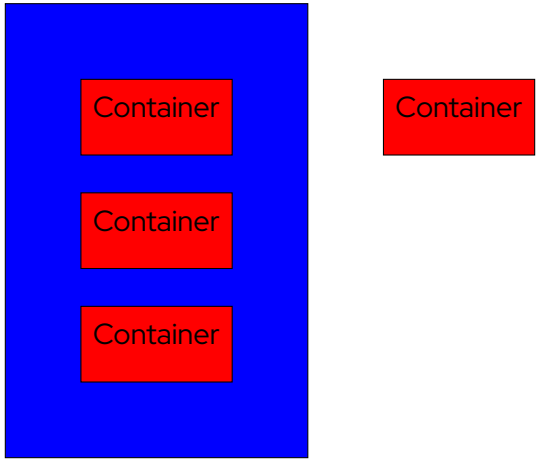




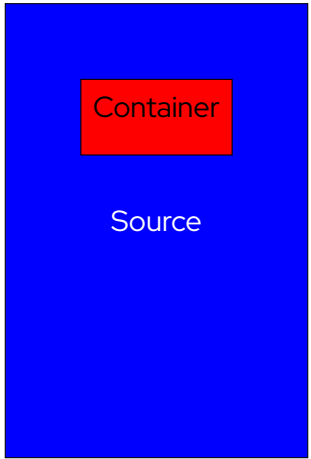
Quick Startup

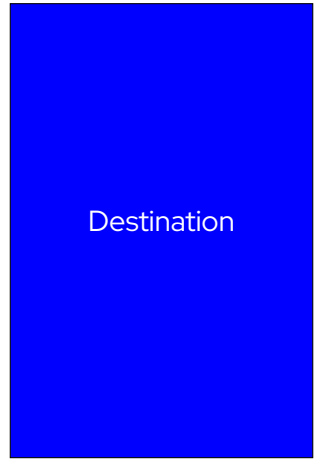


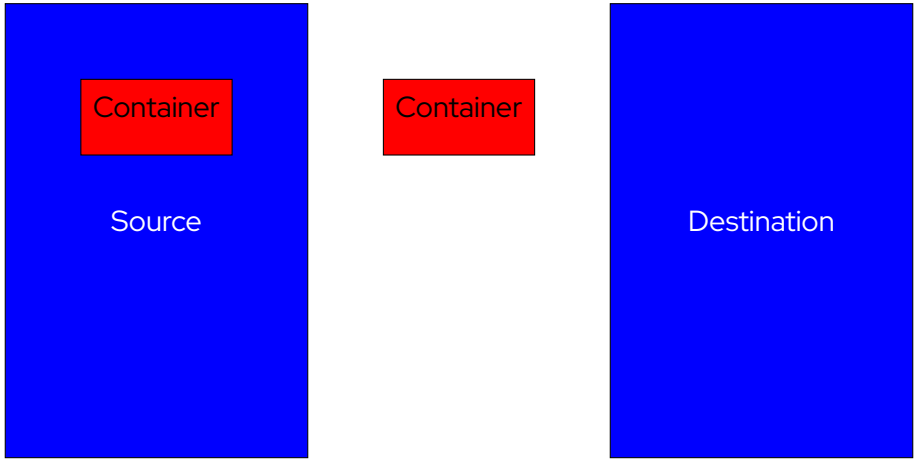


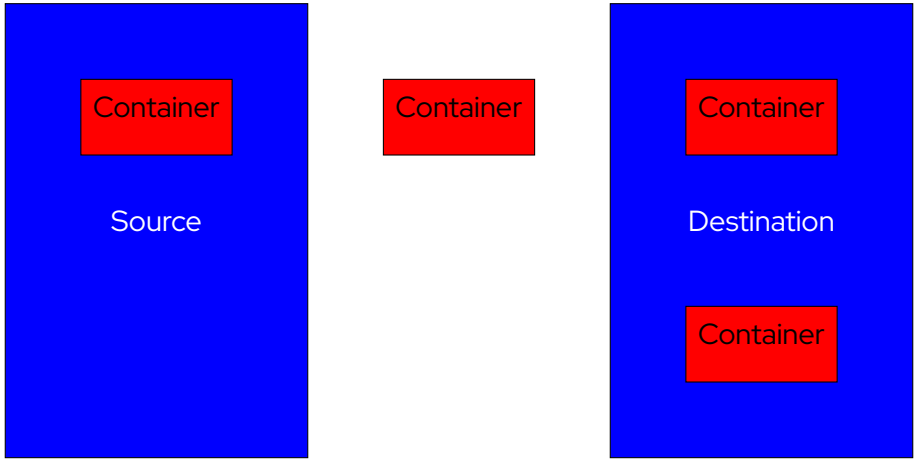


Container Live Migration








Container
Source
Container
Container
Destination
Container

CRIU

First Step: Checkpointing

Seize Process Using

`ptrace()`

Collect Details From

/proc/<PID>/*

Parasite Code

Parasite Code

Most favorite part

Parasite Code

And the craziest

Parasite Code

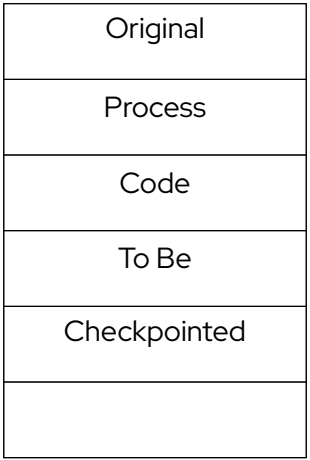
Injected into the process

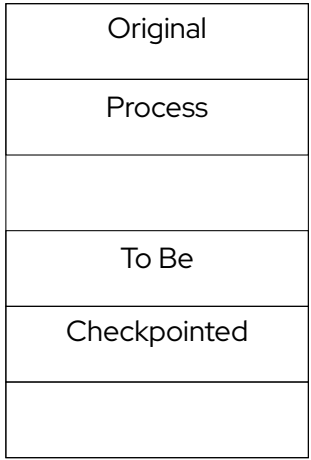
Parasite Code

Daemon waiting for commands

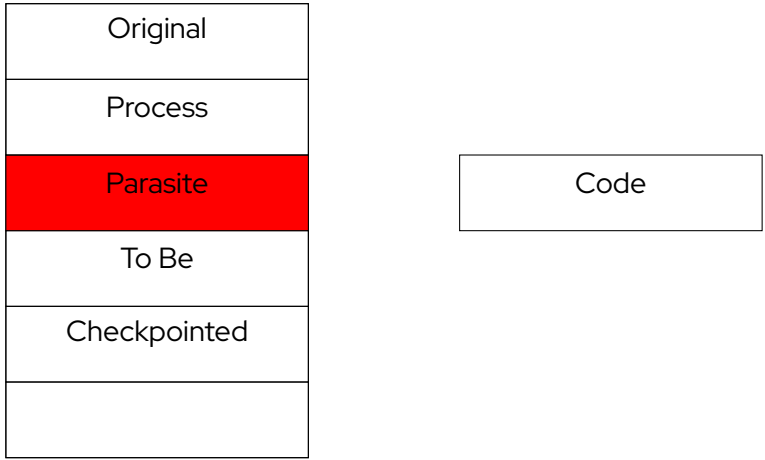
Parasite Code

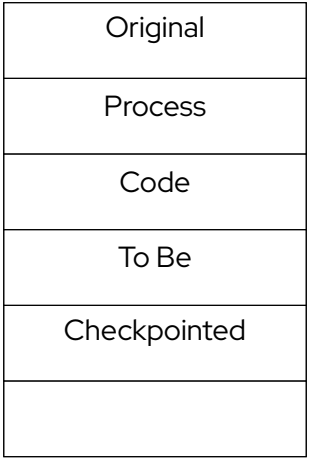
Removed after usage





Code





Checkpointing Finished

Checkpointing Finished

All relevant information written

Checkpointing Finished

Target process is killed

Checkpointing Finished

Or continues to run

Container Live Migration

SELinux

Linux Security Summit EU 2019

https://sched.co/Tymj

2015: CRIU LSM support

During checkpointing

Read `/proc/PID/attr/current`

During restore

Write `/proc/PID/attr/current`

```
1  if (!strstartswith(last, "unconfined_")) {
2    pr_err("Non unconfined selinux contexts not supported %s\n", last);
3    freecon(ctx);
4    return -1;
5  }
```

- `setsockcreatecon(3)` for parasite daemon
- Write to `/proc/PID/attr/current`
- Allow `dyntransition`
- Do not set context of threads
- Allow writing to `/proc/sys/kernel/ns_last_pid`
- Fix socket labels
- Pre-create CRIU log files with appropriate labels
- Fix file descriptor leaks

Second/Last Step: Restoring

Read Checkpoint Images

`clone()` For Each PID/TID

LPC: CRIU and the PID dance

`clone3()` with Linux 5.5

`https://linuxplumbersconf.org/event/4/contributions/472/`

PID dance

`open()` `/proc/sys/kernel/ns_last_pid`

`write()` (PID - 1) to `ns_last_pid`

`close()` `ns_last_pid`

`clone()`

`getpid()`

Avoiding the PID dance (2010):

`eclone()`

`https://lore.kernel.org/patchwork/patch/198220/`

Avoiding the PID dance (2019):

`clone3()`

"In general, clone3() is extensible and allows for the implementation of new features."

`https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=7f192e3cd316ba58c`

clone3() with set_tid

```
1       struct clone_args args = {0};
2       pid_t *set_tid;
3       set_tid[0] = 2020;
4       args.set_tid = set_tid;
5       args.set_tid_size = 1;
6       syscall(__NR_clone3, args, sizeof(struct clone_args));
```

CRIU Morphs Itself

Open and position file descriptors

CRIU Morphs Itself

Map memory pages

CRIU Morphs Itself

Load security settings

CRIU Morphs Itself

Jump into restored process

Container Live Migration

Container Live Migration

OpenVZ

Container Live Migration

Borg

Container Live Migration

LXC/LXD

Container Live Migration

Docker

Container Live Migration

Podman

Podman: daemonless

Podman: rootless

Podman: Checkpoint/Restore

October 2018

Podman: Checkpoint/Restore

Required runc and CRIU changes

Podman: Container Live Migration

June 2019

Podman: Container Live Migration

Required runc, CRIU, SELinux changes

Checkpoint includes File System Changes

```
# podman run --rm -d adrianreber/wildfly-hello
699f33eb7fecbc5bbb00400be0aa79c888dbc63a54cac7bd2eed836a57d8a68a
# podman inspect -l --format "{{.NetworkSettings.IPAddress}}"
10.88.0.247
# curl 10.88.0.247:8080/helloworld/
0
# curl 10.88.0.247:8080/helloworld/
1
# podman container checkpoint -l --export=/tmp/chkpt.tar.gz
699f33eb7fecbc5bbb00400be0aa79c888dbc63a54cac7bd2eed836a57d8a68a
# scp /tmp/chkpt.tar.gz rhel08:/tmp
```

```
# podman container restore --import=/tmp/chkpt.tar.gz
699f33eb7fecbc5bbb00400be0aa79c888dbc63a54cac7bd2eed836a57d8a68a
# podman inspect -l --format "{{.NetworkSettings.IPAddress}}"
10.88.0.247
# curl 10.88.0.247:8080/helloworld/
2
# curl 10.88.0.247:8080/helloworld/
3
```

```
# podman container restore --import=/tmp/chkpt.tar.gz -n hello1
d02feeec894d77f66cc82484fe77ae369396a85f6d05594dc156c21e685942dd
# podman container restore --import=/tmp/chkpt.tar.gz -n hello2
735efb4fee6961d3eee069beb28dde5cbc6fc46c1a32a43ecc993d04c02015b2
# podman inspect --format "{{.NetworkSettings.IPAddress}}" hello1
10.88.0.248
# podman inspect --format "{{.NetworkSettings.IPAddress}}" hello2
10.88.0.249
# curl 10.88.0.248:8080/helloworld/
2
# curl 10.88.0.249:8080/helloworld/
2
```

Future:

`kubectl migrate`

Future:

Non-root checkpoint/restore

Summary

- CRIU can checkpoint and restore containers
- Integrated in different containers engines
- Used in production
- Reboot into new kernel without losing container state
- Start multiple copies
- Migrate running containers

https://lisas.de/~adrian/container-live-migration-article.pdf

https://asciinema.org/a/249922

https://asciinema.org/a/249918

https://lisas.de/~adrian/posts/2019-Apr-10-criu-and-selinux.html

https://criu.org/Podman

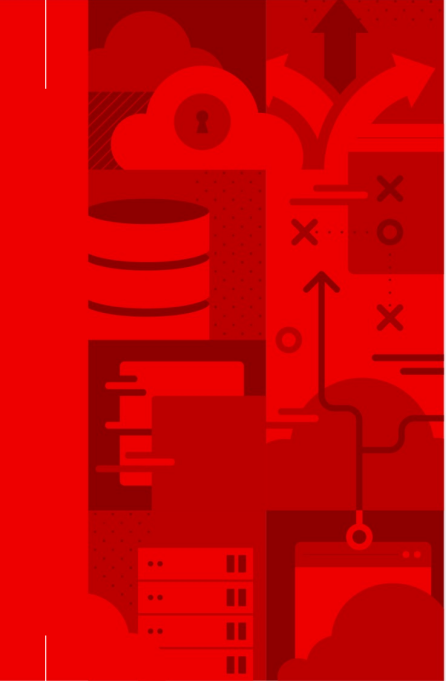
https://twitter.com/adrian__reber

https://www.redhat.com/en/blog/container-migration-podman-rhel

https://cfp.all-systems-go.io/ASG2019/talk/E88Z7V/

https://sched.co/Tymj

https://linuxplumbersconf.org/event/4/contributions/472/



Thank you

Red Hat