

# Lazy Process Migration

Adrian Reber <areber@redhat.com>

Mike Rapoport <rppt@linux.vnet.ibm.com>



Linux Plumbers Conference 2016  
November 1-4, Santa Fe, New Mexico, USA

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688386

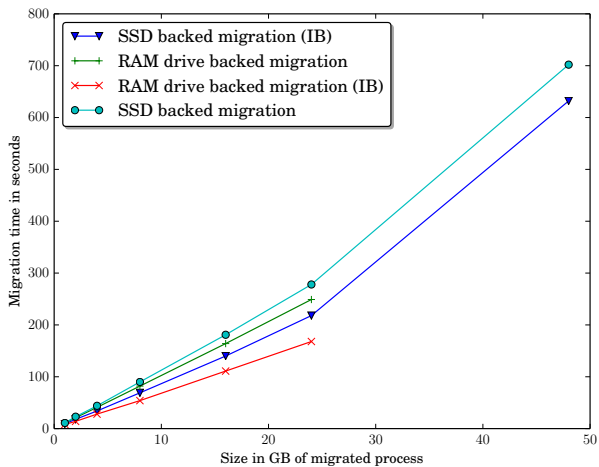


Background

Implementation

Future Plans

# Process Downtime During Migration



# Process Downtime During Migration

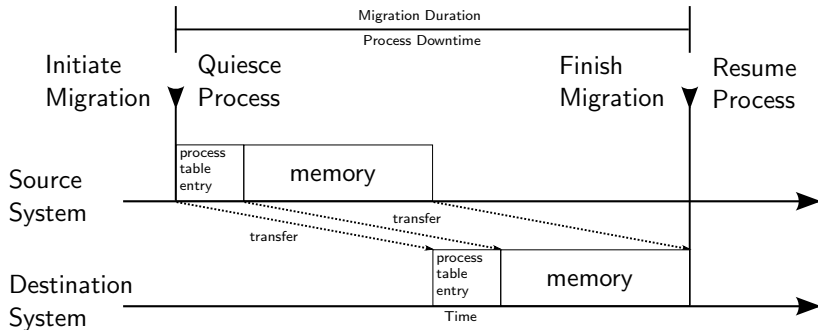


Figure : Process Migration

# Optimizations - Pre-Copy

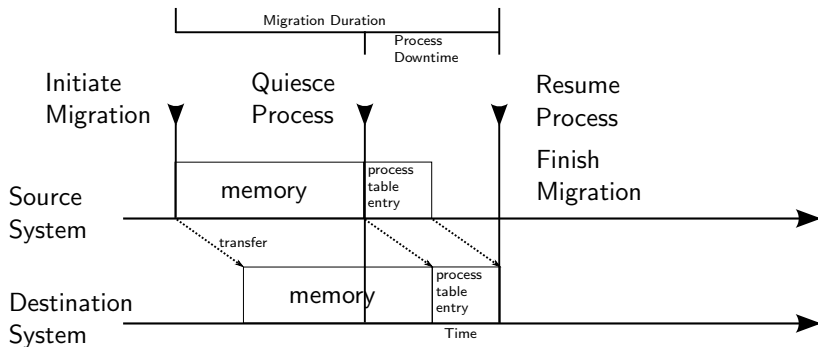
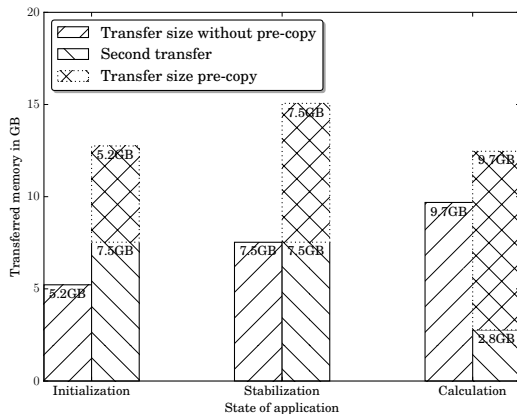
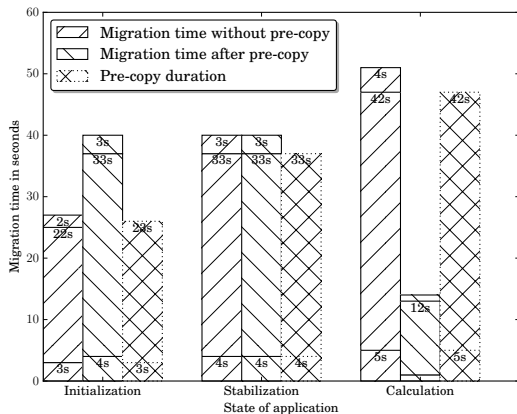


Figure : Pre-Copy Migration

# Possible Drawback Using Pre-Copy



# Optimizations - Post-Copy

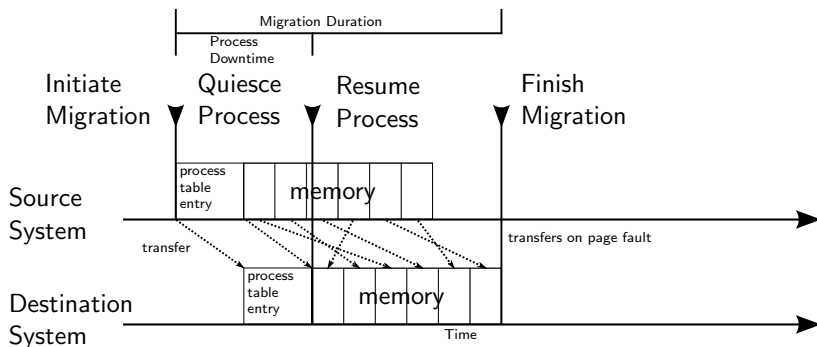


Figure : Post-Copy Migration

## CRIU And Userfaultfd

- ▶ Userfaultfd (UFFD) integration into CRIU
- ▶ Most pages can be handled by UFFD
  - Anonymous private mappings are already supported
  - Shared memory is planned
- ▶ Process downtime can be decreased
- ▶ To restore a 200MB process
  - transfer 200MB without Post-Copy
  - transfer 116KB with Post-Copy



# Lazy Migration Details 1

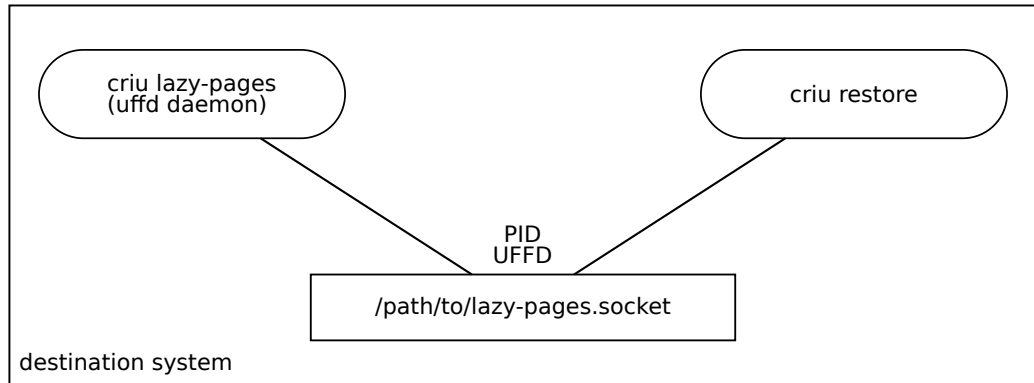
## Source (dump)

- ▶ Memory pages are marked as lazy during dump
- ▶ lazy memory pages are not written to disk
- ▶ Source system waits for requests to transfer lazy memory pages via TCP

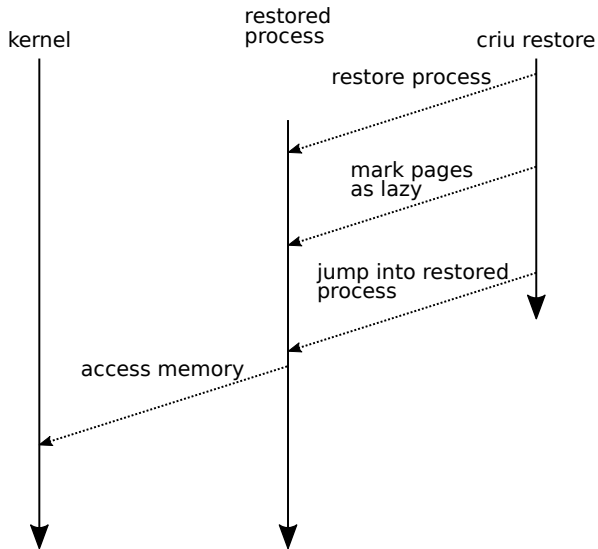
## Destination (restore)

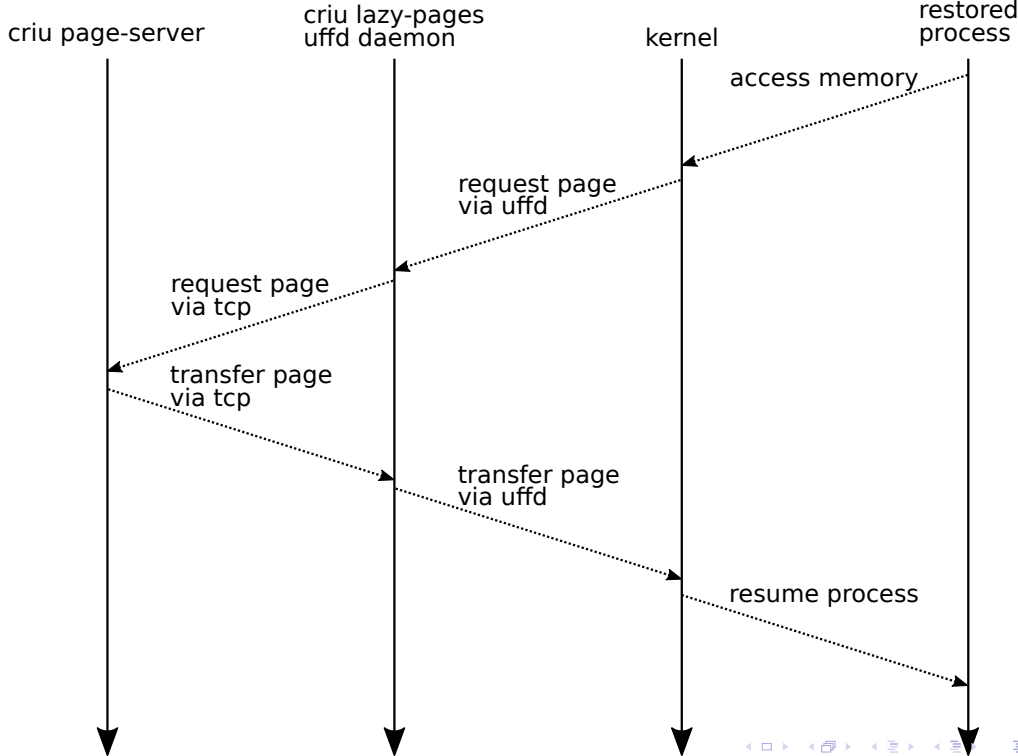
- ▶ Lazy-pages daemon registers memory areas with userfaultfd and connects to the source
- ▶ A process is restored with no memory
- ▶ Process accesses to memory generate page faults which are handled by lazy-pages daemon

## Lazy Migration Details 2



## Lazy Migration Details 3





## Current status

- ▶ In criu-dev branch:
  - local lazy restore works
  - remote lazy restore works
  - combination of pre-copy and post-copy works
- ▶ Kernel patches to userfaultfd<sup>1</sup> are under review on linux-mm<sup>2</sup>
  - non-cooperative mode (support for `fork()` and other events)
  - support for shared memory

---

<sup>1</sup><https://git.kernel.org/cgit/linux/kernel/git/andrea/aa.git/aa.git>

<sup>2</sup><http://www.spinics.net/lists/linux-mm/msg115992.html>

# Limitations

- ▶ A process that executes `fork()`, `madvise(MADV_DONTNEED)` or `mremap` will fail
- ▶ Shared (tmpfs) and hugelbfs mappings cannot be handled by `userfaultfd`
- ▶ Post-copy performance is far from optimal

## Future plans

- ▶ Add post-copy support to *p.haul*, *runc*, *lxc*
- ▶ Non-cooperative userfaultfd (`fork()` and other events) in CRIU and in the kernel
- ▶ Shared memory post-copy
- ▶ Nested userfaultfd
- ▶ Optimizations

# The end.

Thanks for listening.