Fast Checkpoint Restore for AMD GPUs with CRIU

Felix Kuehling
Rajneesh Bhardwaj
David Yat Sin

What is CRIU?

- Checkpoint Restore in User space
  - https://criu.org/Main_Page
- Cannot Checkpoint processes with device files (yet)
  - Device specific state unknown / invisible to CRIU
  - E.g. GPU-specific state:
    - VRAM buffer objects
    - GPU VA mappings
    - User mode queues
- CRIU plugin mechanism
  - Shared object, loaded before dumping or restoring
  - No real device plugin that exists
  - More new hooks required for GPU devices

CRIU - High Level Architecture

- Works on a process tree, parent task and all its descendants
  - On restore, the PIDs remains the same

- Checkpoint
  - PTRACE_ATTACH/SEIZE/INTERRUPT/* to stop/freeze process execution, like a debugger attach
  - Inject a parasite code that drains file descriptors, signals, timers, events, etc. through RPC API
  - Zero copy the memory (RAM) contents via vmsplice system call
  - Serializes all the data using protobuf and writes to the disk

- Restore
  - Master restore process forks all child processes, sets up position independent code (PIC) mapping in each
  - Each child process restores own state, morphs into the previously saved process
  - Restoring processes synchronized through several stages
  - Last stage jumps to PIC code: Unmaps CRIU, fixes up restored VA mappings
  - Master restore process then detaches and yields rt-sigreturn to continue execution of the restored processes

CRIU support for ROCm™

- CRIU modifications
  - 3 new plugin hooks in CRIU
  - Support for device file VMAs
  - Under review by CRIU community: https://github.com/checkpoint-restore/criu/pull/1556
- AMDGPU plugin (amdgpu_plugin.so)
  - Built as part of CRIU
  - Expected to be upstreamed to CRIU
  - Current WIP: https://github.com/RadeonOpenCompute/criu
- KFD ioctl APIs to support plugin
  - Expected to be upstreamed to Linux kernel
  - Under review by DRI and AMDGPU community
  - Current WIP: https://github.com/RadeonOpenCompute/ROCK-Kernel-Driver/commits/fxkamd/criu-wip

AMDGPU PLUGIN (amdgpu_plugin.so)

- Supports saving of GPU device files:
  - /dev/kfd
  - /dev/dri/renderD*
- Drains GPU state from KFD
  - Uses new KFD ioctls
  - Copies VRAM contents using SDMA engine
  - Supports multiple GPUs
  - Supports ROCm compute applications only
  - Does not support
    - Vulcan compute
    - OpenGL
    - Video decode/encode acceleration
- Saves GPU state to new protobuf image file (kfd-<id>.img)

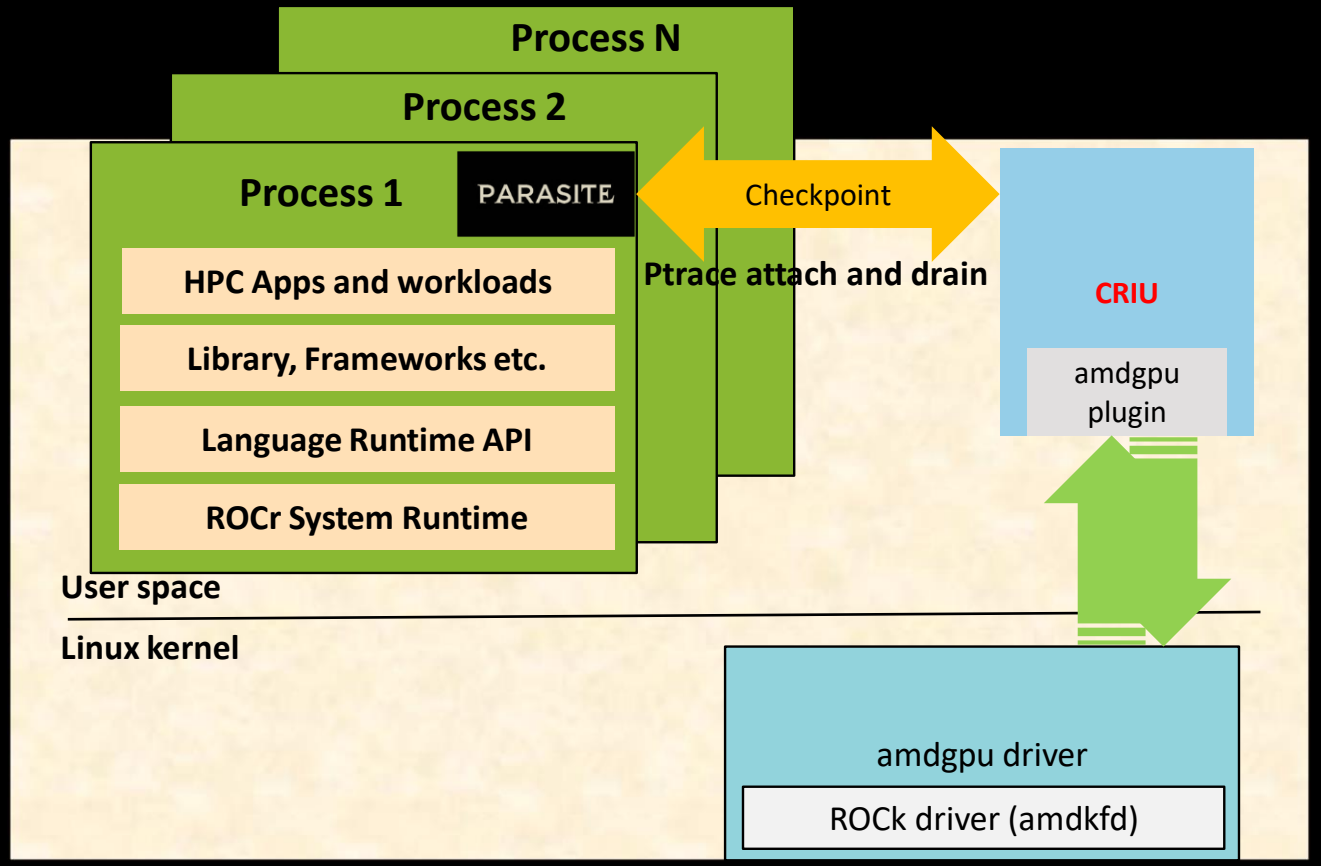
AMDGPU PLUGIN (amdgpu_plugin.so)

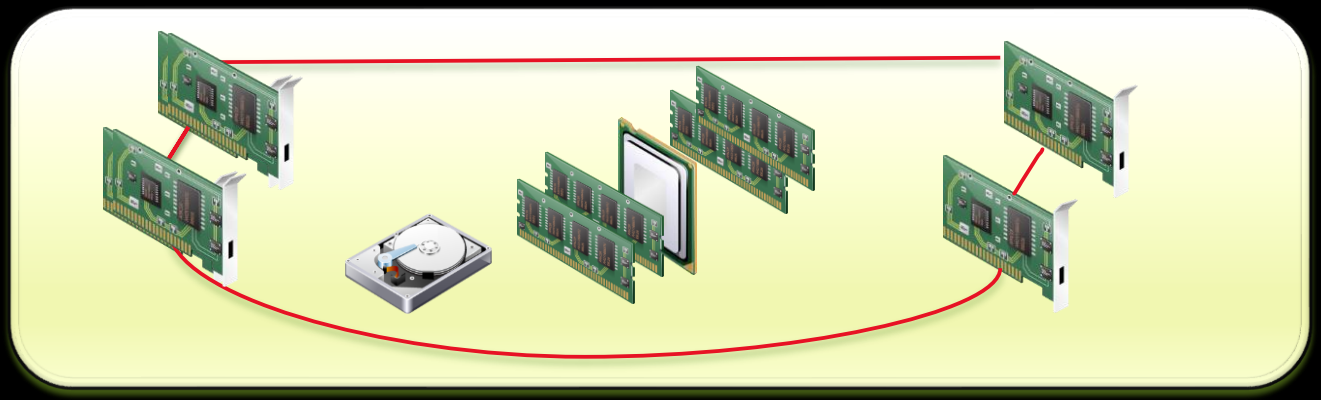
Plugin Hook	Status	Description
DUMP_EXT_FILE	EXISTS	Saves all KFD state and VRAM contents for the process
RESTORE_EXT_FILE	EXISTS	Prepares all the KFD state and restores VRAM contents for the process
UPDATE_VMA_MAP	NEW	Updates restored memory mappings, file paths
RESUME_DEVICES_LATE	NEW	Restarts queues, MMU Notifiers, Restores SVM ranges, makes process ready to run in almost the last CRIU restore phase
HANDLE_DEVICE_VMA	NEW	Detect a suitable plugin to handle device file VMA with PF | IO mappings

AMDKFD IOCTLs (amdgpu kernel driver)

- API definitions are currently under community review
- Authorization checks for ioctl calls affecting remote processes or restoring privileged state
  - CAP_SYS_ADMIN and PTRACE_ATTACHED flags

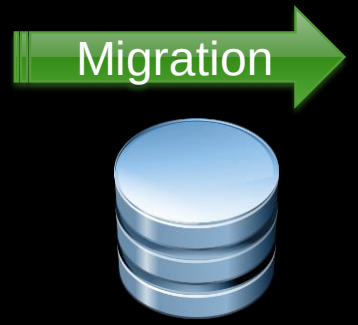
Name	Caller Context	Description
CRIU_PAUSE	CRIU (ptrace)	ioctl to make sure all queues are evicted
CRIU_PROCESS_INFO	CRIU (ptrace)	ioctl to determine current process information such as various objects and their types
CRIU_DUMPER	CRIU (ptrace)	ioctl to save BO metadata and other KFD state
CRIU_RESTORER	Target process	ioctl to restore the BOs and other KFD state
CRIU_RESUME	CRIU (remote)	ioctl registers MMU Notifiers, restores SVM ranges, restarts queues (late stage after VMAs locations are finalized by CRIU)





amdgpu plugin drains the GPU state:

- *Topology and device cgroups*
- *Memory Allocations*
- *GPU virtual address mappings, including memory and doorbells*
- *HMM virtual address range attributes*



SECURITY CONSIDERATIONS

Threats:

- Read access to remote process state
- Control of remote process execution
- Write access to privileged HW state

Mitigation:

- Read access to remote process requires ptrace attached caller
- Control of remote process execution
  - Requires ptrace attached caller during checkpoint
  - Requires CAP_SYS_ADMIN to resume execution of restored process
- Write access to privileged state requires CAP_SYS_ADMIN

Demo

UPSTREAM STATUS

- Yes, It will be available upstream!
- Plugin and APIs are under review
- Finalizing work on HMM support and CRIU Image streamer
- Next Steps:
  - Work with the Linux kernel community to accept the new APIs
  - Get the CRIU Pull requests merged
  - Improve image size and speed up further

DISCLAIMER AND ATTRIBUTIONS

DISCLAIMER

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.