Using CRIU for Computer Architecture and Software Optimization Studies

Or: Getting Results Faster and With Less Work

Christopher Covington

August 20th, 2015

Terminology

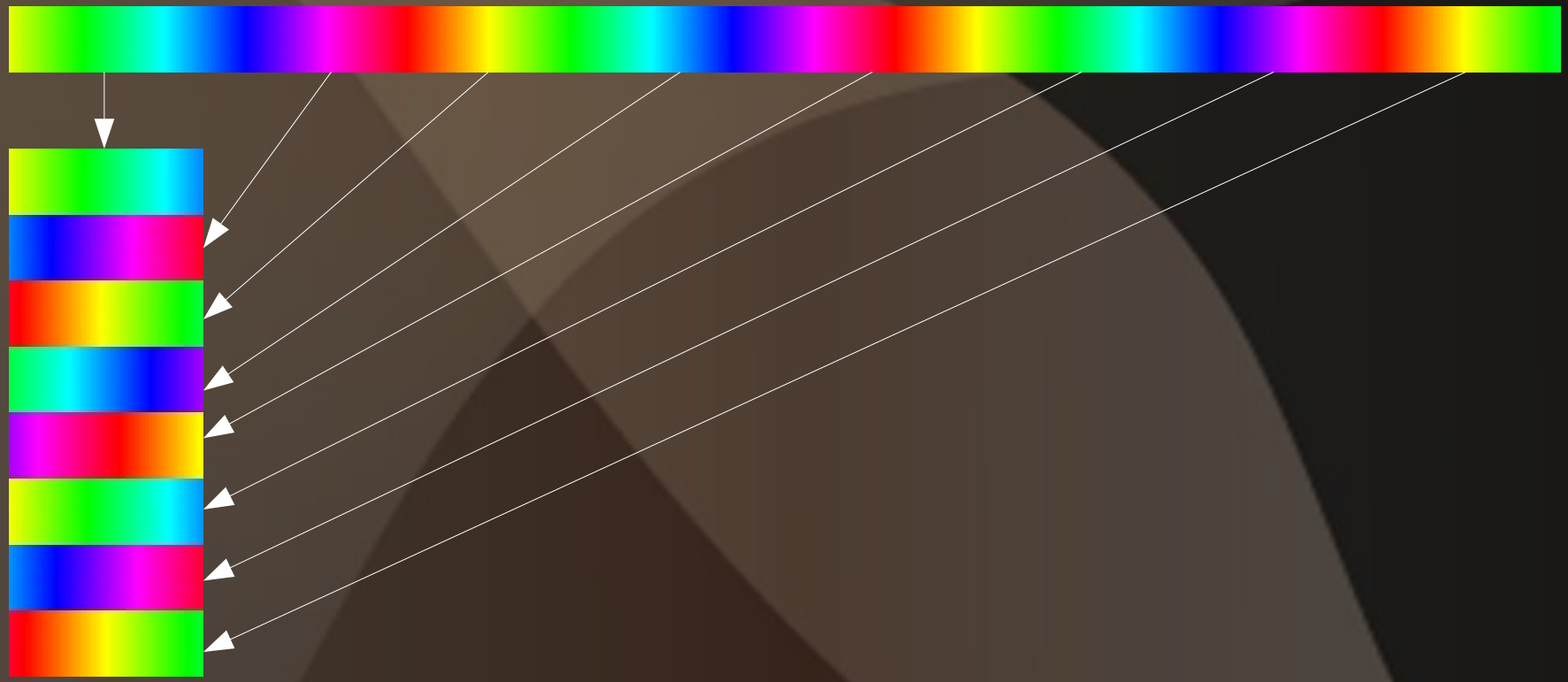
- How zoomed in or zoomed out is the checkpoint? ("Amount of state saved")
  - System level (QEMU snapshots)
  - OS level (TuxOnIce)
  - Application level (CRIU)
- What provides the checkpointing facilities? ("System architecture")
  - Externally driven (QEMU snapshots)
  - Self-hosting (TuxOnIce, CRIU)

Terminology

- System types
  - “Functional” models
  - “Timing” models
  - Hardware description language simulators
- Fast forwarding: dumping a checkpoint on a fast system and restoring it on a slow system

Long Application

Long Application, Fast Forwarded



Fast Forwarding Assumptions

- Determinism: Starting from the same initial state and running for the same duration faithfully recreates subsequent state
- Checkpointing: Checkpoints faithfully recreate initial state

Linux Facilities Used Alongside CRIU

- perf_events framework counting instructions for fast-forwarding (have tried software breakpoints via gdb)
- Stop signal (should maybe upgrade to cgroups freezer)

Fast Forwarding using CRIU

Architecturally executed instructions the basic unit of measurement.

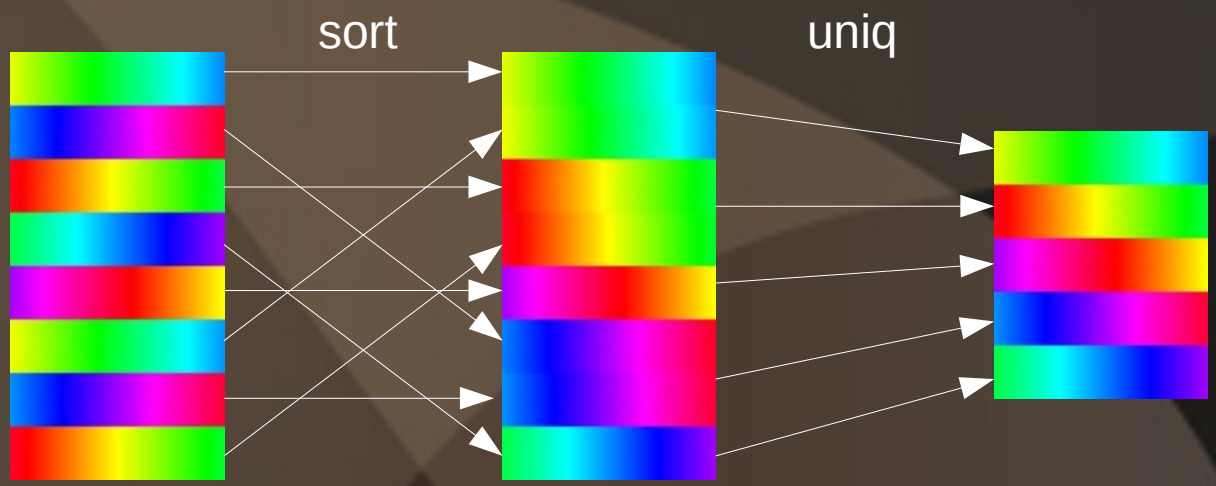
On fast system:

```
ptrace-wait $pid $(($isize * $inum))
criu dump -j -t $pid
```

On slow system:

```
criu restore
perf stat -t $pid
ptrace-wait $pid $isize
```

Sampling to Avoid Redundant Work



SMARTS statistical sampling

http://users.ece.cmu.edu/~jhoe/doku/doku.php?id=smarts_simulation_sampling

SimPoint k-means clustering

https://www.cs.ucsb.edu/~sherwood/pubs/IEEEMicro-phases.pdf

Setup/Special Case of 0 Instructions

```
stopexec logfile -- application arg1 arg2
criu dump -j -t $pid
```

Cold Start Effects

- Excess page faults observed immediately after restore
- Currently working around these by dumping checkpoints pretty far in advance
- Better approach?

Ptrace Poke Side Effects

- Sharing of physical pages is broken (copy-on-write kicks in) for first page when it is ptrace poked
- Not so significant for 4K pages, but potentially significant for 64K pages
- Could the copy-on-write be undone?
- Could the poke be done elsewhere? VDSO?

Dump and Restore of perf_events and ftrace

- Current implementation keeps perf_event file descriptors outside of the CRIU-dumped process tree
- Would it be useful to dump and restore perf_event file descriptors? What about ftrace?
- How to ignore, or count and compensate for, parasite activity (such as instructions) when dumpee is being traced?

Self-Restoring Checkpoints

- Analogous to the self-unpacking Linux kernel zImage, link restorer code, data, and executable together in a single binary
- For my use case, this is a system level checkpoint
- Can trim it down to contain only those values used a specific interval after restore
- Probably most useful on the slowest sorts of systems, providing portability between them
- “Intrinsic Checkpoints with Binary Modification” http://deepblue.lib.umich.edu/handle/2027.42/60726

Checkpoint Interoperability

- Speculative, but what if QEMU linux-user mode, CRIU, core dumps, and self-restoring checkpoints could interoperate?
- crit becomes a babelcheckpoint of sorts?

Thank You

Christopher Covington is an employee of the Qualcomm Innovation Center, Inc. The Qualcomm Innovation Center, Inc. is a member of the Code Aurora Forum, a Linux Foundation Collaborative Project.