The Taming of the Kernel Dump

Presenter: Petr Tesařík

Date: 14th November 2023

Location: LPC Richmond, VA

Background

Motivation

- A wheel that was reinvented too many times:
  - lcrash

Motivation

- A wheel that was reinvented too many times:
  - lcrash
  - crash

Motivation

- A wheel that was reinvented too many times:
  - lcrash
  - crash
  - makedumpfile

Motivation

- A wheel that was reinvented too many times:
  - lcrash
  - crash
  - makedumpfile
  - *(kdumpid)*

Motivation

- A wheel that was reinvented too many times:
  - lcrash
  - crash
  - makedumpfile
  - *(kdumpid)*
- Let’s make THE wheel! Reliable. Universal.
  One wheel to rule them all.

Motivation

- A wheel that was reinvented too many times:
  - lcrash
  - crash
  - makedumpfile
  - *(kdumpid)*
- Let’s make THE wheel! Reliable. Universal.
  ~~One wheel to rule them all.~~

Design Goals

- Mimic actual hardware
- Cross-platform
- Support non-Linux operating systems
- Small enough for crash kernel
- Fast enough to run on crashed systems
- Expose detailed information
- Reasonably easy to use

Challenges

Architecture Zoo

	crash	makedumpfile	libkdumpfile
alpha	YES	NO	NO
aarch64 (arm64)	YES	YES	YES
arm	YES	YES	YES
ia32 (x86)	YES	YES	YES
ia64	YES	YES	NO
loongarch64	YES	YES	NO
mips	YES	NO	NO
mips64	YES	YES	NO
powerpc	YES	YES	NO
ppc64	YES	YES	NO
riscv64	YES	YES	NO
s390	YES	NO	NO
s390x	YES	YES	YES
sparc64	YES	YES	NO
x86_64	YES	YES	YES

Architecture Zoo

	crash	makedumpfile	libkdumpfile
alpha	YES	NO	NO
aarch64 (arm64)	YES	YES	YES
arm	YES	YES	YES
ia32 (x86)	YES	YES	YES
~~ia64~~	~~YES~~	~~YES~~	~~NO~~
loongarch64	YES	YES	NO
mips	YES	NO	NO
mips64	YES	YES	NO
powerpc	YES	YES	NO
ppc64	YES	YES	NO
riscv64	YES	YES	NO
s390	YES	NO	NO
s390x	YES	YES	YES
sparc64	YES	YES	NO
x86_64	YES	YES	YES

Adding Architectures

- There is a HOWTO:
  https://github.com/ptesarik/libkdumpfile/wiki/Adding-Architectures
  - Page Table Translation (or whatever the hardware does)
  - Linux Support
  - CPU Registers

Format Zoo

- Common
  - bare metal: ELF, kdump (makedumpfile)
  - virtual: qemu, libvirt, Xen ELF, xc_core, xc_save
- Vendor-specific
  - SADUMP, Ramdump, S390DUMP, z/VM vmdump
- Historic
  - LKCD, diskdump

Adding Formats

- No HOWTO document :-(
- Requires good knowledge of the internal file cache API
- Some formats may need non-existent infrastructure
- Dump formats often poorly documented
  - KDUMP is defined by makedumpfile sources
  - SADUMP is non-public and reverse-engineered

Conversion PoC

- Makedumpfile once partially converted
- v1.6.8 (~100 commits behind now)
  - File reads: easy, some fallouts (e.g. cache stats)
  - Address translation: mostly fine (except SADUMP and Xen)
  - VMCOREINFO: somewhat messy
- Complies with current Linux development trends:

```
19 files changed, 741 insertions(+), 4401 deletions(-)
```

Other Wild Ideas

GDB Server

- https://github.com/ptesarik/kdump-gdbserver
  - gdbserver frontend to crash dump files
  - written in Python using pykdumpfile bindings
  - assumes distinct kernel-space and user-space addresses

Write Support

- How to design the API?
  - Some preparatory work (CoW clone of the attribute hierarchy)
  - Replace `makedumpfile` input *and* output routines?
    - Initialize input context
    - Adjust `file.pagemap` and `file.format` of a cloned context
    - Write output context
  - What about split dump files?

Thank you!