Kexec/Kdump 实现与应用

Impl and Appl of Kexec/Kdump

202312

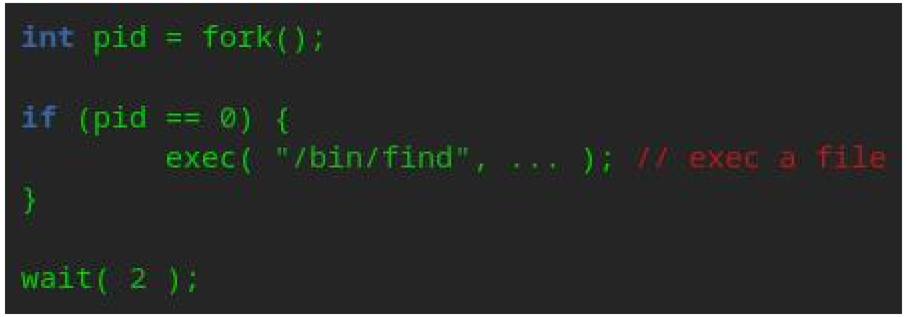
songshuaishuai@uniontech.com

Content

- What is Kexec/Kdump
- How to use Kexec/Kdump
- Kexec/Kdump Impl -- A big picture
- Kexec/Kdump Impl -- Q&A
- Next ...

What is Kexec/Kdump（userspace）

- exec()



```
int pid = fork();

if (pid == 0) {
        exec( "/bin/find", ... ); // exec a file
}

wait( 2 );
```

- coredump
    - ```*(int*)(NULL) = 1 ; ``` // Segmentation fault (core dumped)
    - gdb <executable_path> <coredump_file_path>

What is Kexec/Kdump

- What is Kexec

  - directly boot into a new kernel from current kernel w/o firmware initialization

  - reduce the time required from a reboot and friendly for kernel development .. + openeuler/nvwa

  - related softwares: kexec-tools + kernel

  > IMO, Kexec is a OS loader (prepare/load/execute)

- What is Kdump

  - When panic use Kexec to quickly boot to a 2nd kernel where you can dump 1st kernel memory

  - related softwares: kexec-tools + kernel + makedumpfile + Crash

How to use Kexec/Kdump

- Kexec

  1. kexec -l vmlinux | kexec_[file]_load() syscall

  2. kexec -e | reboot(,,LINUX_REBOOT_CMD_KEXEC,) syscal

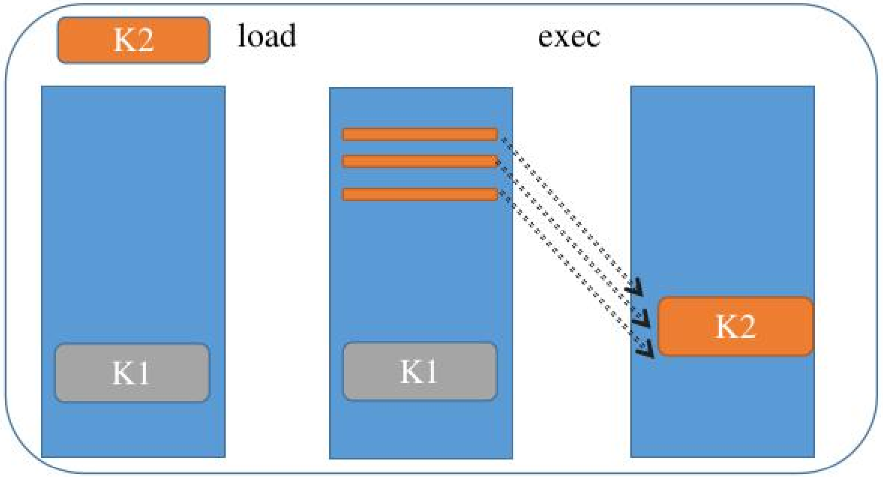
- Kdump

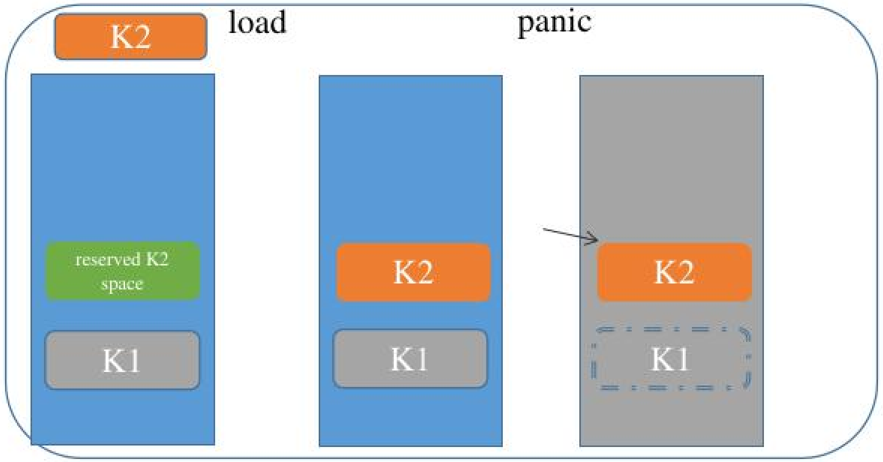
  0. K1 cmdline set `crashkernel=`

  1. kexec -p vmlinux | kexec_[file]_load() syscall

  2. boot to K2 when panic

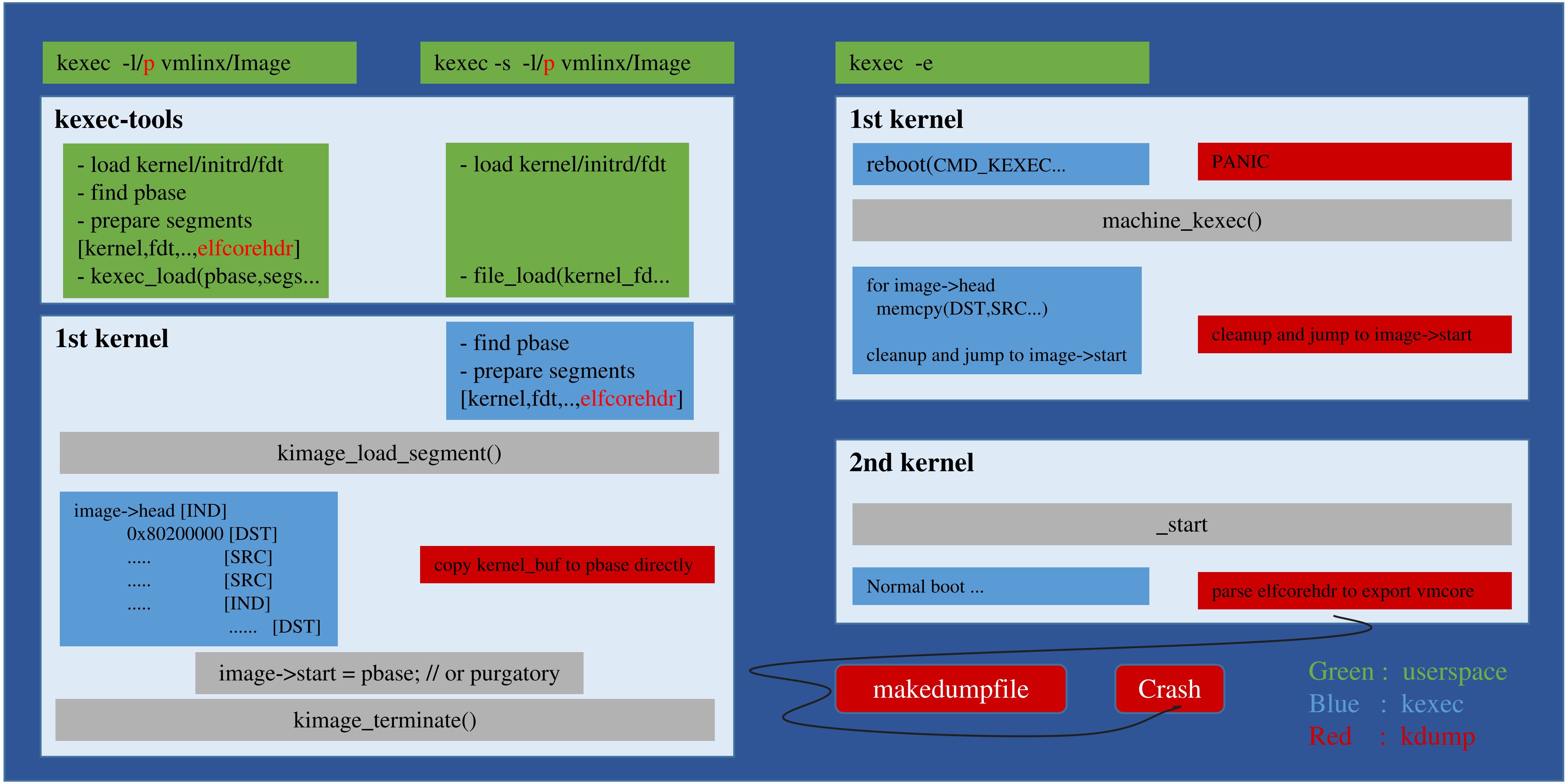
  x. makedumpfile /proc/vmcore as a dumpfile | Crash





Kexec/Kdump Impl -- A big picture

- kernel/kexec*.c
- arch/*/purgatory/
- arch/*/kernel/*kexec*
- Documentation/admin-guide/kdump/



kexec -l/p vmlinx/Image

kexec -s -l/p vmlinx/Image

kexec -e

kexec-tools

- load kernel/initrd/fdt
- find pbase
- prepare segments [kernel,fdt,..,elfcorehdr]
- kexec_load(pbase,segs...

- load kernel/initrd/fdt
- file_load(kernel_fd...

1st kernel

- find pbase
- prepare segments [kernel,fdt,..,elfcorehdr]

kimage_load_segment()

```
image->head [IND]
        0x80200000 [DST]
        .....         [SRC]
        .....         [SRC]
        .....         [IND]
                ......   [DST]
```

copy kernel_buf to pbase directly

image->start = pbase; // or purgatory

kimage_terminate()

1st kernel

reboot(CMD_KEXEC...

PANIC

machine_kexec()

```
for image->head
  memcpy(DST,SRC...)

cleanup and jump to image->start
```

cleanup and jump to image->start

2nd kernel

_start

Normal boot ...

parse elfcorehdr to export vmcore

makedumpfile

Crash

Green : userspace
Blue : kexec
Red : kdump

elf format
DYN/EXEC/CORE

Image format/header

OF_kexec :
initrd/usable-memory-range/kalsr-seed

ARCH boot protocol

psABI Spec
asm manual

kernel mapping

Kexec/Kdump Impl -- Q&A

- Q1. The difference between kernel_load() and kernel_file_load()
    - SYSCALL_DEFINE4(kexec_load, unsigned long, entry, unsigned long, nr_segments, struct kexec_segment __user *, segments, unsigned long, flags)
    - SYSCALL_DEFINE5(kexec_file_load, int, kernel_fd, int, initrd_fd, unsigned long, cmdline_len, const char __user *, cmdline_ptr, unsigned long, flags)
    - Actually, kexec_file_load() offloads the work kexec-tools did before calling kexec_load() to kernel
- Q2. Will the loaded vmlinux corrupt the current kernel's memory ?
    - e.g. The K1 was loaded at 0x80200000, load the same kernel image, would it corrupt the K1 memory?
    - Kexec: Just tag the addresses from kernel image as DST,SRC,IND when loading, `kexec -e` trigger the real memory copying at the end of machine_kexec() where !ie !mmu
    - Kdump: The `crashkernel=` of K1 reserved the memory for paniced kernel which wouldn't be mapped/used via K1, so we can directly kmap|copy kernel image to the reserved memory when loading

Kexec/Kdump Impl -- Q&A(cont.)

- Q3. How the paniced kernel(K2) fetch the 1st kernel(K1)'s memory info?
  - K1 creates chosen/linux,elfcorehdr to hold the K1's memory info and pass it to K2 :
    - 1. PT_NOTE: vmcoreinfo (e.g. init_ns, phys_ram_base, PAGE_OFFSET) and crash_note (e.g. regs, CRASHTIME)
    - 2. PT_LOAD: all memory that 1st kernel used
  - K2 parses the chosen/linux,elfcorehdr to export /proc/vmcore as a elf CORE file

- Q4. Whatif the loaded segment is different with the executing segment?
  - Use KEXEC_PURGATORY to digest all segments when loading and re-check them when executing

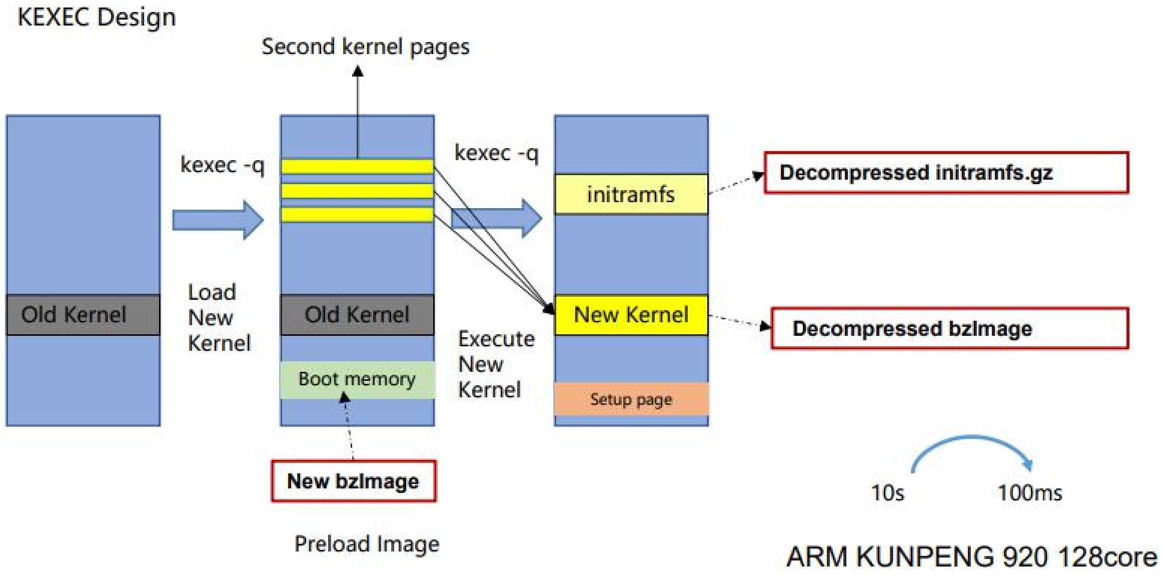
Next (Kdump)

- Use Kdump to anaylze kernel panic
  - apt install kdump-tools && reboot ; PANIC ; crash /var/crash/dump.XXX

- Improve distro’s Kdump toolchain
  - make the toolchain stable
    - kdump-tools.deb | kexec-tools | makedumpfile | Crash
  - backport/upstream kdump support for new ARCHes or kernel changes
    - Crash : pull/150 : add loongarch64 support from ut004615 :-p
    - kexect-tools’ support for new chosen::linux,usable-memory-range dts property

Next (Kexec)

- To bisect kernel Images, use Kexec instead of gru
    - 4.19[bad] -<...>- 5.10 [good] -- 6.0 [good]



- a Kexec user -- openeuler/nvwa
    - a "system" live update tool using **kexec** and **criu**
    - use criu to hibernate and resume apps,
        - freeze | dump to disk/mem | restore | thaw
        - but there are some apps/contexts can't be dumped [1]
    - use kexec to boot 2nd kernel "more quickly" [2]
        - use reserved physical continuous Pages instead of vmalloc'ed Pages to copy

[1]: https://criu.org/What_cannot_be_checkpointed
[2]: Google : fosdem.org 2022 Seamless_Kernel_Update.pdf

References

- linux source code
  - kernel/kexec*.c
  - arch/*/purgatory/
  - arch/*/kernel/*kexec*
  - Documentation/admin-guide/kdump/
- lore.kernel.org/kexec
- crash-utility/crash
- makedumpfile/makedumpfile
- horms/kexec-tools
- openeuler/nvwa

Thanks