Basic Kernel Crash Dump Analysis

Hardik Vyas

Technical Support Engineer (Kernel)

19/04/2016

Agenda

PART-1

- Basics of kernel panic and system hang
- kdump and different methods available to capture a kernel crash dump (vmcore)

PART-2

- Basics of crash
- Initial Analysis
- Memory Subsystem

Basics of kernel panic and system hang

What is kernel panic ?

- A kernel panic is an action taken by an operating system upon detecting an internal fatal error from which it cannot safely recover.

- It is possible for kernel code to indicate such a condition by calling the panic function declared in the header file "sys/system.h".

Basics of kernel panic and system hang

What is kernel panic ?

- There are four main causes of kernel panic:
  - Software problems
  - Hardware problems
  - Firmware/BIOS problems
  - Manual or condition panic

- Software problems:
    - o Bug in kernel
    - o Bug in kernel module
    - o Bug in unsigned (U) proprietary (P) kernel module
    - o Bug in hypervisor etc.

Examples of Software Problems:

- BUGs / Assertion errors

```
kernel BUG at net/sunrpc/sched.c:695!
kernel BUG at /opt/kaspersky/kav4fs/src/kernel/redirfs/rfs_inode.c:61!
```

causes of kernel panic, continued...

- Null pointer dereference

```
BUG: unable to handle kernel NULL pointer dereference at (null)
BUG: unable to handle kernel NULL pointer dereference at 0000000000000008
```

- Unable to handle kernel paging request

```
BUG: unable to handle kernel paging request at ffffea00028e8018
```

causes of kernel panic, continued...

■ Hardware problems:

- o Faulty RAM (Physical Memory)
- o Faulty Processor
- o Faulty PCI interfaces and USB devices etc.

Examples of Hardware Problems:

- Machine Check Exception (MCE)

  Kernel panic - not syncing: Fatal Machine check

causes of kernel panic, continued...

- Non-Maskable Interrupts (NMIs)

  Kernel panic - not syncing: NMI IOCK error: Not continuing

- Error Detection and Correction (EDAC)

  Kernel panic - not syncing: UE row 3, channel-a= 2 channel-b= 3 labels "-": (Branch=1 DRAM-Bank=1 RDWR=Read RAS=356 CAS=0 FATAL Err=0x4 (&gt;Tmid Thermal event with intelligent throttling disabled))

causes of kernel panic, continued...

- Firmware/BIOS problems:
  - o Bug in firmware
  - o Bug in BIOS.

Examples of Firmware Problems:

- Firmware bug causing Memory corruption.

causes of kernel panic, continued...

- Manual or condition panic:
  - o Panic on SysRq event (c)
  - o Panic on OOM detection
  - o Panic on hung task detection
  - o Panic on soft lockup detection etc.

Example of Manual or Condition Problems:

- Panic on SysRq event (c)

  PANIC: "SysRq : Trigger a crashdump"

- Panic on OOM detection.

  Kernel panic - not syncing: out of memory. panic_on_oom is selected

causes of kernel panic, continued...

- Panic on hung task detection.

  Kernel panic - not syncing: hung_task: blocked tasks

- Panic on soft lockup detection.

  Kernel panic - not syncing: softlockup: hung tasks

- Panic triggered by GAB module (Veritas cluster suite fencing module)

  Kernel panic - not syncing: GAB: Port h halting system due to client process failure"

- Panic triggered by HP Watchdog timer module [hpwdt]:

  Kernel panic - not syncing: An NMI occurred, please see the Integrated Management Log for details.

causes of kernel panic, continued...

- Panic triggered by NMI watchdog.

  ```
  Kernel panic - not syncing: Watchdog detected hard LOCKUP on cpu 0
  ```

- Panic triggered by Oracle RAC (cssdmonitor or cssdagent) via SysRq event (c)

  ```
  PANIC: "SysRq : Trigger a crashdump"
  ```

Basics of kernel panic and system hang

What data is required to determine the root cause of a kernel panic ?

- Kernel Crash Dump (vmcore) captured at the time of issue.

OR

- Complete Screen-shot of the kernel panic or console message.

Basics of kernel panic and system hang

What is system hang?

There is no standard definition of the system hang. The system hang is a "fuzzy" concept which depends on the "criteria" of the observer (end user).

Examples:

- User's are unable to Login on the system via console or ssh.
- Unable to ping the server.
- The system gets partially or completely stalled and most services become unresponsive.
- Respond to user inputs with an obvious latency (an unacceptable length of time according to the observer).

Basics of kernel panic and system hang

What data is required to determine the cause of system hang ?

Collect as much as possible information:

Examples:

- How did you determined that the system is in unresponsive state or hang ?
- What are the symptoms ?
- Details of any misbehavior or suspicious activity (resource utilization).
- Details of any recent changes made on the server prior to the incidence.
- Kernel Crash Dump (vmcore) captured at the time of incidence.

Basics of kernel panic and system hang

What are the different causes of the system hang ?

Some possibilities:

- Bug in kernel-space/user-space.
- Faulty hardware/firmware.
- Resource utilization.
- Bug or Unexpected behavior of Hypervisor etc...

kdump and different methods available to capture a kernel crash dump (vmcore)

Kdump and different methods available to capture a kernel crash dump(vmcore)

What is kdump?

- Kdump is a kexec based crash dumping mechanism for Linux kernel.
- kdump support in kernel:

```
$ grep CRASH_DUMP /boot/config-2.6.32-431.el6.x86_64
CONFIG_CRASH_DUMP=y
```

- Kdump can be configured to dump either directly to a device, to a file, or to some location on the network via NFS or SSH.
- Kdump and kexec are currently supported on the x86, x86_64, aarch64, ppc64, ia64, and s390x architectures.

Kdump and different methods available to capture a kernel crash dump(vmcore)

How does it work?

- Whenever the standard (production) kernel crashes, kdump uses kexec to boot into a crash (capture) kernel.

- The standard (production) kernel reserves a section of memory that the crash (capture) kernel uses to boot.

- Be aware that the memory reserved for the kdump kernel at boot time cannot be used by the standard kernel.

- Kexec enables booting the capture kernel without going through the BIOS, so contents of the first kernel's memory are preserved, which is essentially the kernel crash dump (vmcore).

Continued....

What are the different kdump configuration files and directives?

The following two configuration files are used to configure kdump.

- /etc/sysconfig/kdump { configuraiton file for kexec kernel }
- /etc/kdump.conf { configuration file for kdump service }

/etc/kdump.conf file.

path <path>

- "path" represents the filesystem path in which vmcore will be saved.
- If unset, will default to /var/crash.

Eg:

```
path /var/crash
```

b) /etc/kdump.conf file.

core_collector <command> <options>

- This allows you to specify the command to copy the vmcore.
- Default core_collector for other targets is: "makedumpfile -c --message-level 1 -d 31"

Eg:

```
core_collector makedumpfile -c --message-level 1 -d 31
```

- The option is a bit mask, having each page type specified like so:

zero pages	1
cache pages	2
cache private	4
user pages	8
free pages	16

- In general, these pages don't contain relevant information. To set all these flags and leave out these pages, use a value of -d 31.

default <reboot | halt | poweroff | shell | dump_to_rootfs>

- Used to set action to preform in case dumping to intended target fails.
- If no default action is specified, "reboot" is assumed default.

reboot: If the default action is reboot simply reboot the system and loose the core that you are trying to retrieve.

halt: If the default action is halt, then simply halt the system after attempting to capture a vmcore, regardless of success or failure.

- **poweroff:** The system will be powered down

- **shell:** If the default action is shell, then drop to an shell session inside the initramfs from where you can try to record the core manually. Exiting this shell reboots the system. Note: kdump uses bash as the default shell.

- **dump_to_rootfs:** If non-root dump target is specified, the default action can be set as dump_to_rootfs. That means when dump to target fails, dump vmcore to rootfs from initramfs context and reboot.

Eg:

```
default shell
```

How to configure kdump service ?

a) Set kernel boot parameter "crashkernel" to reserve memory for crash/kdump kernel.

- Three different formats for "crashkernel" parameter:

  o crashkernel=auto { Automatically calculates memory for crash kernel }

  o crashkernel=128M { Reserves 128M of memory for crash kernel }

  o crashkernel=0M-2G:128M,2G-6G:256M,6G-8G:512M,8G-:768M

b) Reboot the system.

c) Verify the memory reserved for crash/kdump kernel.

- Check /proc/cmdline to confirm system is booted with "crashkernel" parameter.

Eg:

```
# cat /proc/cmdline

ro root=UUID=a216d1e5-884f-4e5c-859a-6e2e2530d486
rd_NO_LUKS rd_NO_LVM LANG=en_US.UTF-8 rd_NO_MD
SYSFONT=latarcyrheb-sun16 crashkernel=128M
KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM reconfig
```

- Check /sys/kernel/kexec_crash_loaded file.

Eg:

```
# cat /sys/kernel/kexec_crash_loaded
1
```

1: crash kernel is loaded.

0: crash kernel is not loaded.

- Check /sys/kernel/kexec_crash_size file for memory reserved for crash kernel.

Eg:

```
# cat /sys/kernel/kexec_crash_size
134217728


# bc -q
scale=2
134217728/2^20
128.00    <<<----{ 128 MiB }
```

- Check /proc/iomem file for memory reserved for crash kernel.

```
$ grep -i crash /proc/iomem
  03000000-0affffff : Crash kernel


$ printf "%d\n" 0x0affffff
184549375


$ printf "%d\n" 0x03000000
50331648


$ bc -q
scale=2
(184549375 - 50331648)/2^20
127.99  <<<---{ 128 MiB }
```

Configure kdump for local device

- Typical example of kdump on local device:

```
# vi /etc/kdump.conf
path /var/crash
core_collector makedumpfile -l --message-level 1 -d 31
default reboot
```

- Note: If no default action is specified, "reboot" is assumed default.

- Enable kdump service.

  Eg:

  ```
  # chkconfig kdump on
  ```

- Start kdump service.

  Eg:

  ```
  # service kdump start
  ```

- Verify kdump service status.

  Eg:

  ```
  # service kdump status
  ```

How to test kdump setup ?

- Enable magic key "SysRq" to trigger kernel panic.

  Eg:

```
# echo 1 > /proc/sys/kernel/sysrq
```

- Trigger SysRq panic event (c) to crash the system manually.

  Eg:

```
# echo c > /proc/sysrq-trigger
```

How to verify kernel crash dump (vmcore) ?

- Check directory set by "path" in kdump.conf file for "vmcore" and "vmcore dmesg.txt" files.

Eg:

```
# tree /var/crash/
/var/crash/
└── 127.0.0.12015.06.0817:25:32
    ├── vmcore
    └── vmcoredmesg.txt
1 directory, 2 files
```

What is a kernel crash dump (vmcore)?

- Kernel crash dump (vmcore) is a dump of all the physical memory (RAM, registers, but not swap) at a particular point in time.
- Kernel crash dump captures the state of kernel at the moment of panic.

What kind of data is contained in a kernel crash dump (vmcore)?

- Kernel Space data:

  - o Kernel space data structure.
  - o Kernel ring buffer (“dmesg”)
  - o CPU state and its registers information
  - o Memory usage information
  - o Run queue information
  - o Mounted file systems information
  - o Disk device information
  - o In-flight I/O information
  - o Process state and their kernel mode stack information
  - o IRQ handling information
  - o etc.

- User Space data:

  - o User space data structure.
  - o Application data in physical memory etc.

When will the kernel crash dump (vmcore) be collected?

- The kdump service dumps the kernel crash dump (vmcore) only at the time of kernel panic.

- The kernel panic can be triggered after one of the following events:

  a) Manually:

   o Using SysRq facility.
   o Using NMI via IPMI tool.
   o Using NMI via virsh command.
   o etc.

b) Automatically:

o Due to bug kernel

o Due to bug in kernel modules

o Due to memory corruption

o Due to invalid pointer access

o Due to hardware problems

o Due to conditional switches

- Panic on OOM (vm.panic_on_oom = 1)

- Panic on hung task detection (kernel.hung_task_panic = 1)

- Panic on soft lockup detection (kernel.softlockup_panic = 1)

What are the different mechanisms available to capture kernel crash dump (vmcore) at the time of system hang?

Physical Machine

a) Using **SysRq**:

- How to enable SysRq key?

o Set the value of kernel.sysrq to 1 in /etc/sysctl.conf file.

```
# grep sysrq /etc/sysctl.conf
  kernel.sysrq = 1
```

o Execute sysclt -p for the changes to take effect.

```
# sysctl -p
```

OR

o Execute `# echo "1" > /proc/sys/kernel/sysrq`

- How to trigger panic?

o Execute `# echo "c" > /proc/sysrq-trigger`

OR

o Press key combination `[ Ctrl + Alt + SysRq + C ]`

b) Using NMI via IPMI tool:

- Execute `# ipmitool -I lan -H <Host> -U <User ID> -a chassis power diag`

c) Using NMI jumper pins or dump switch

d) Using HP iLO Virtual NMI Button

What are the different mechanisms available to capture kernel crash dump (vmcore) at the time of system hang?

Virtual Machine

a) Using SysRq: [libvirt] [VMware]

- Execute `# echo "c" > /proc/sysrq-trigger`

OR

- Press key combination `[ Ctrl + Alt + SysRq + C ]`

b) Using virsh dump command: [libvirt]

- virsh dump command sends a request to dump the core of a guest virtual machine to a file.

- Execute `# virsh dump <vm-name> /storage/<vm-name>.dump`

c) Using NMI via virsh command: [libvirt]

- Execute `# virsh inject-nmi guest-1`

d) Using gcore command: [libvirt]

- Identify the PID of the qemu-kvm process running the guest:

Eg:

```
# ps aux | grep qemu-kvm
qemu     12345  0.4  4.9 1320252 808212 ?      Sl
Sep10   6:55 /usr/libexec/qemu-kvm -name rhel6
```

- Capture an application core using gcore command against this PID:

```
# gcore <PID of guest process>
```

Eg:

```
# gcore 12345
Saved corefile core.12345
```

e) Using VMware® vmss2core tool: [VMWare] (Not supported by Red Hat Support)

- Suspend the virtual machine (.vmss file) or take a snapshot with memory (.vmsn file).
- Use tool vmss2core (VMWare Labs) to transform the raw dump into ELF dump.

Eg:

```
# vmss2core -N6 <vmName>.vmss
```

Download link for VMware® vmss2core tool:

https://labs.vmware.com/flings/vmss2core

f) Using xm command: [Xen Guest]

- You can use xm to perform a memory dump of an existing virtual machine. This command dumps the virtual machine's memory to the xendump file located in the /var/xen/dump/ directory
- Execute `# xm dump-core [-C] [domain-id]`

Who needs kernel crash dump (vmcore) and why?

- **Kernel developers:** To identify the cause of kernel panic. It helps them to fix the bug.

- **Enterprise customers:** To get the root cause analysis and take corrective action(s) accordingly.

What are the limitations of vmcore analysis?

- A vmcore is not useful when the system is not experiencing any problem.
- A vmcore is a snapshot. It does not capture a complete history. History reconstruction may not be possible.
- A vmcore analysis can be costly due to complexity and the possible need for in-depth kernel expertise.

Questions