**一种Linux内核调试方法-Kdump使用介绍**

**一、Kdump简介**

Kdump 是一种基于 kexec 的 Linux 内核崩溃捕获机制。将 kernel 崩溃前的内存镜像保存至文件，开发人员通过分析该文件找出 kernel 崩溃的原因，从而进行系统改进。本文首先介绍了 kdump 的基本概念和工作原理，其次阐述了Kdump在COS系统上的安装、配置和使用方法。

**1.1 Kexec**

Kexec 是一个快速重启 Linux 系统的工具，可以从当前正在运行的内核直接引导到另一个内核，其快速主要体现在从内核开始启动而不是从BIOS开始，不经过硬件初始化程序。对带有众多数量的外设的大型服务器而言，Kexec 可以大大减少了重新启动引起的系统宕机时间。对内核和系统软件开发者而言，Kexec 可以帮助迅速重新启动系统，而不必每次都要再经历耗时的硬件初始化阶段。

Kexec快速重启系统的流程如下图所示。

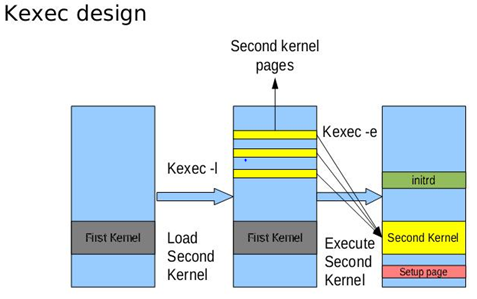


图1 Kexec重启系统流程图

Kexec是实现Kdump机制的关键，它包括2个组成部分：

（1）内核空间的系统调用 kexec\_load，负责在标准内核（production kernel 或 first kernel）启动时将捕获内核（capture kernel 或 second kernel）加载到指定地址。

（2）用户空间的工具 kexec-tools，他将捕获内核的地址传递给标准内核，从而在系统崩溃的时候能够找到捕获内核的地址并运行。

没有 Kexec 就没有Kdump。先有Kexec 实现了在一个内核中可以启动另一个内核，才让Kdump 有了用武之地。Kexec原来的目的是为了节省 kernel 开发人员重启系统的时间，谁能想到这个“偷懒”的技术却孕育了最成功的内存转储机制呢？

**1.2 Kdump**

Kdump的概念出现在2005年左右，是迄今为止最可靠的内核转存机制，已经被主要的linux厂商选用。Kdump是一种先进的基于kexec的内核崩溃转储机制。当系统崩溃时，Kdump通过 kexec 启动到第二个内核。第二个内核通常叫做捕获内核，以很小内存启动以捕获转储镜像。第一个内核保留了内存的一部分给第二内核启动用。由于 kdump 利用 kexec 启动捕获内核，绕过了 BIOS，所以第一个内核的内存得以保留并转储。

Kdump实现内存转储的过程如下图所示。

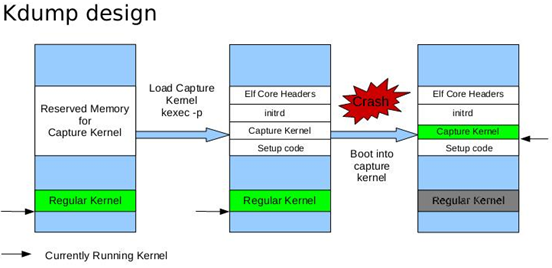


图2 Kdump的内存转储流程图

Kdump 需要两个不同目的的内核，标准内核和捕获内核：

（1）标准内核：正常运行的内核，是捕获内核服务的对像。

（2）捕获内核：在标准内核崩溃时启动，与相应的 ramdisk 一起组建一个微环境，用以对标准内核下的内存进行收集和转存。

**二、Kdump工作原理**

实现Kdump的内存转储功能需要构建两个内核，构建方式有以下 2 种可选：

（1）构建一个单独的自定义转储捕获内核以实现内存转储；

（2）将系统内核本身作为转储捕获内核，这就不需要构建一个单独的转储捕获内核。

方法（2）只能用于可支持可重定位内核的体系结构上；目前 i386，x86\_64，ppc64 和 ia64 体系结构支持可重定位内核。构建一个可重定位内核使得不需要构建第二个内核就可以捕获转储，但是某些情况下还是需要构建一个自定义转储捕获内核以满足特定要求。

对于i386和x86\_64架构的CPU：

a.如果kernel不可重定位，使用vmlinux格式内核。

b.如果kernel可重定位，使用bzImage或vmlinuz格式内核。

在内核崩溃之前所有关于内核映像的必要信息都用 ELF 格式编码并存储在保留的内存区域中。ELF 头所在的物理地址被作为命令行参数（fcorehdr=）传递给新启动的转储内核。

在 i386 体系结构上，启动的时候需要使用物理内存开始的 640K，而不管操作系统内核转载在何处。因此，这个 640K 的区域在重新启动第二个内核的时候由 kexec 备份。

在第二个内核中，“前一个系统的内存”可以通过两种方式访问：

(1) 通过 /dev/oldmem

一个“捕捉”设备以“raw”（裸的）格式“读”/dev/oldmem这个设备文件并写出到文件。命令：dd if=/dev/oldmem of=/home/v10/oldmem1。

这是内存的“裸”的数据转储。这种raw格式的文件解析工具需要足够“智能”从而可以知道从哪里可以得到正确的信息。ELF 文件头（通过命令行参数传递过来的elfcorehdr）可能会有帮助。

(2) 通过 /proc/vmcore

这个方式是将转储内存输出为一个 ELF 格式的文件，通过文件拷贝命令（比如 cp，scp 等）将信息读出来，由makedumpfile将转储文件压缩保存至指定目录（默认/var/crash）。

系统重启后，使用apport工具解压经过压缩的转储文件，得到一个叫VmCore的文件。通过crash工具解析VmCore文件，可以查看系统崩溃之前的堆栈、函数跟踪等信息。

Kexec是用新内核去覆盖原内核位置，而Kdump是预留一块内存来加载第二个内核(和相关数据)，Crash后第二个内核在原位置运行(不然就达不到相关目的了)，收集第一个内核的相关内存信息。在Kdump中Kexec算是一个引导器，类似GRUB2。

**三、使用Kdump分析内核崩溃原因**

**3.1 配置内核编译选项**

为了能够运行Kdump，在编译捕捉内核时需要配置以下内核选项：

（1）配置与kexec相关的选项：

* 在 "Processor type and features."选项中启用"kexec system call"。

CONFIG\_KEXEC=y

* 在"Filesystem" -> "Pseudo filesystems." 中启用"sysfs file system support"。

CONFIG\_SYSFS=y

* 在"Kernel hacking."中启用"Compile the kernel with debug info"。

CONFIG\_DEBUG\_INFO=y

（2）配置捕捉内核的与架构无关的选项：

* 在"Processor type and features"中启用"kernel crash dumps"。

CONFIG\_CRASH\_DUMP=y

* 在"Filesystems" -> "Pseudo filesystems"中启用"/proc/vmcore support"。

CONFIG\_PROC\_VMCORE=y

（3）配置捕捉内核的与架构相关的选项（Linux 内核支持多种 CPU 架构，这里只介绍捕捉内核在 i386 下的配置）：

* 在"Processor type and features"中启用高端内存支持。

CONFIG\_HIGHMEM4G=y

* 在"Processor type and features"中启用"Build a relocatable kernel"。

CONFIG\_RELOCATABLE=y

**3.2 安装Kdump工具包**

在COS下使用Kdump只需要安装linux-crashdump这个包就可以了，可以在新立得包管理器下搜索linux-crashdump或者用命令：

sudo apt-get install linux-crashdump

linux-crashdump主要安装了五个工具，分别是： kexec-tools，kdump-tools， makedumpfile，apport以及crash。

安装完毕以后，我们检查一下grub2的启动选项（cat /proc/cmdline），会发现在引导内核的命令linux后面多了一个参数：

crashkernel=384M-2G:64M,2G-:128M

crashkernel参数用来指定保留内存的大小，语法定义如下：

crashkernel=<range1>:<size1>[,<range2>:<size2>,...][@offset]

range=start-[end]

其中，start包含在指定范围内，而end不包含在指定范围内。

参考这个语法定义，我们就可以知道linux-crashkernel帮我们设定的保留区域的大小是：如果内存小于384M，不保留内存；如果内存大于等于384M但小于2G，保留64M；如果内存大于2G，保留128M。

在/etc/init.d中还会有一个kdump的service脚本，这个脚本主要是通过kexec把/boot下的linux内核镜像当作capture kernel放入保留内存里。

Kdump安装完后，重启系统，kdump自动启动。如果没有启动，可以通过命令service kdump start启动。

**3.3 查询Kdump运行状态**

（1）查询捕获内核是否加载

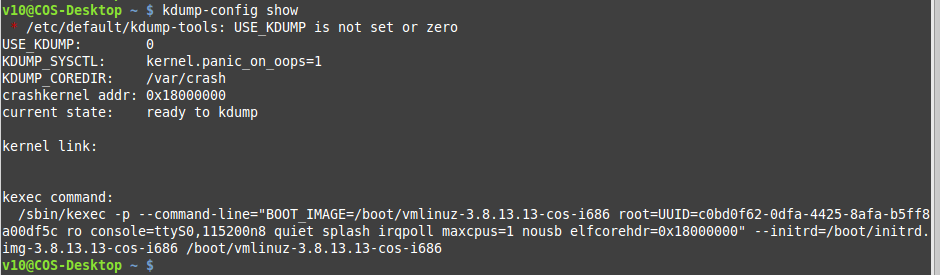
cat /sys/kernel/kexec\_crash\_loaded

若值为1，表示已加载，若为0表示未加载。



（2）查询Kdump运行状态

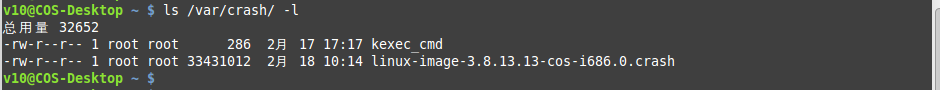
/usr/sbin/kdump-config show，状态为ready to kdump表示准备就绪。



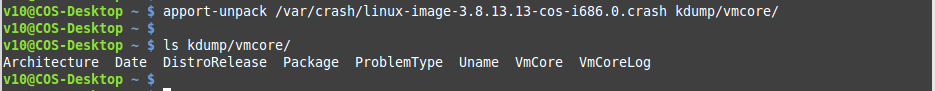
**3.4 分析转储文件**

在捕捉内核被加载进入内存后，如果系统崩溃开关被触发，则系统会自动切换进入捕捉内核。触发系统崩溃的开关有 panic()，die()，die\_nmi() 内核函数和 sysrq 触发事件，可以使用其中任意的一个来触发内核崩溃。

为了举例说明，使用sysrq触发内核崩溃。在root下使用命令：echo c > /proc/sysrq-trigger,内核崩溃后自动重启，重新登录系统后可以在/var/crash下看到生成的后缀为.carsh的内核转储文件。



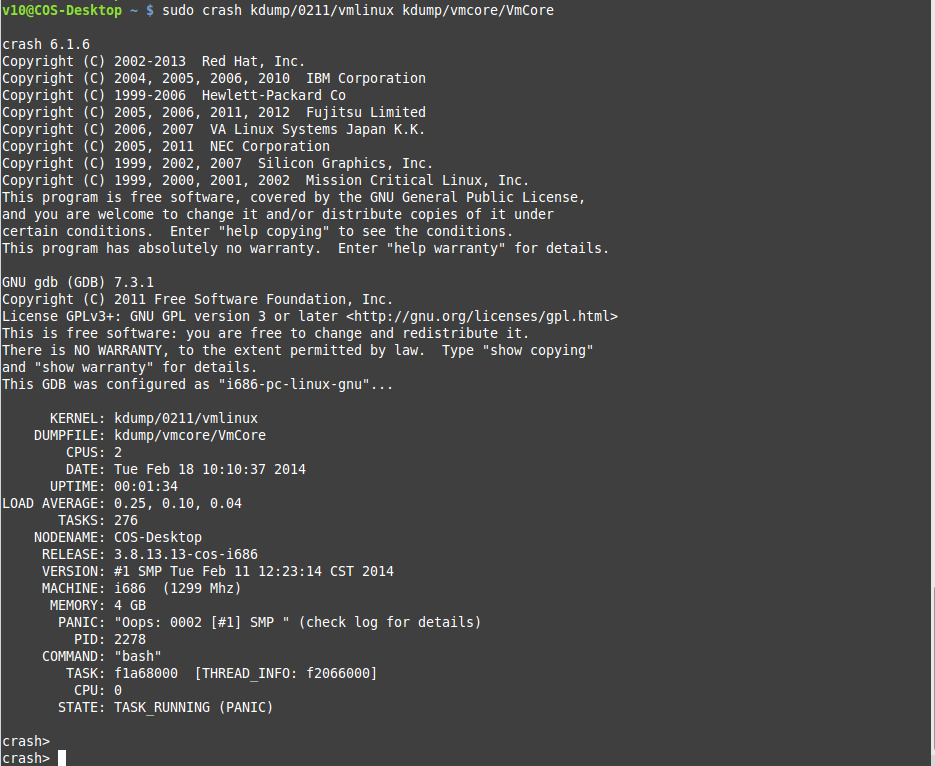
这个crash转储文件是经过makedumpfile压缩过的文件，需要使用apport-unpack工具解压该文件，解压后在对应目录下可以看到VmCore文件。



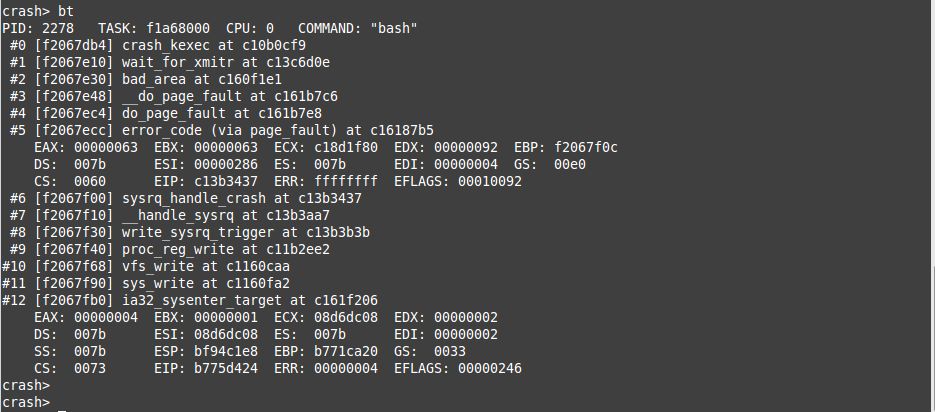
得到VmCore文件之后，我们就可以使用crash工具（基于GDB的调试器）来分析了，命令如下：

sudo crash /…/vmlinux /…/VmCore

其中vmlinux是与当前系统版本号相同的未压缩的内核镜像文件。



在crash里，我们可以用bt查看内核崩溃时的call trace：



也可以使用log、ps等命令查看相关信息。

**四、参考文献**

1. Kdump项目网站：<http://lse.sourceforge.net/kdump/>

2. <http://www.cnblogs.com/wwang/archive/2010/11/19/1881304.html>

3. Linux内核Kdump介绍：<https://www.kernel.org/doc/Documentation/kdump/kdump.txt>

4. Linux Kernel Crash Book. Igor Ljubuncic Dedoimedo

5. <http://www.dedoimedo.com/computers/kdump.html>