Debugging Linux Kernel Using Kdump

Vivek Goyal (vgoyal@in.ibm.com)
Haren Myneni (haren@us.ibm.com)
Dave Anderson (anderson@redhat.com)

Various Debugging Methods

- printk()
- Oops Messages

  The main trick is having 5 years of experience with those pesky oops messages

  -- Linus Torvalds
- Debuggers
  - kdb
  - kgdb

Various Debugging Methods Contd..

- Kprobes
- Out of tree crash dumping mechanisms
  - LKCD
  - diskdump
  - netdump

There is no panacea.

Crash Dump Overview

- System memory snapshot is saved upon failure
- Crash dump can be used for post-crash analysis
- Useful in debugging system failures, especially in field installations
- Kdump is a crash-dumping mechanism that is mainline

Background

- Dump capture from crashing kernel's context
  - Resource lockup
  - Corrupt data structures
- Dedicated dump drivers
  - Limited number of target devices
  - Maintenance was a big issue
  - Dependency on crashing kernel reduced and not eliminated completely

Background Contd..

- Stand-alone dumpers
  - Need to maintain low level, hardware-specific code
  - Filtering is not possible
- Kernel reboot-based dumper
  - Memory constraint might prevent capturing full dump
  - Significant amount of code being run in crashing kernel context
  - Core kernel invasive code

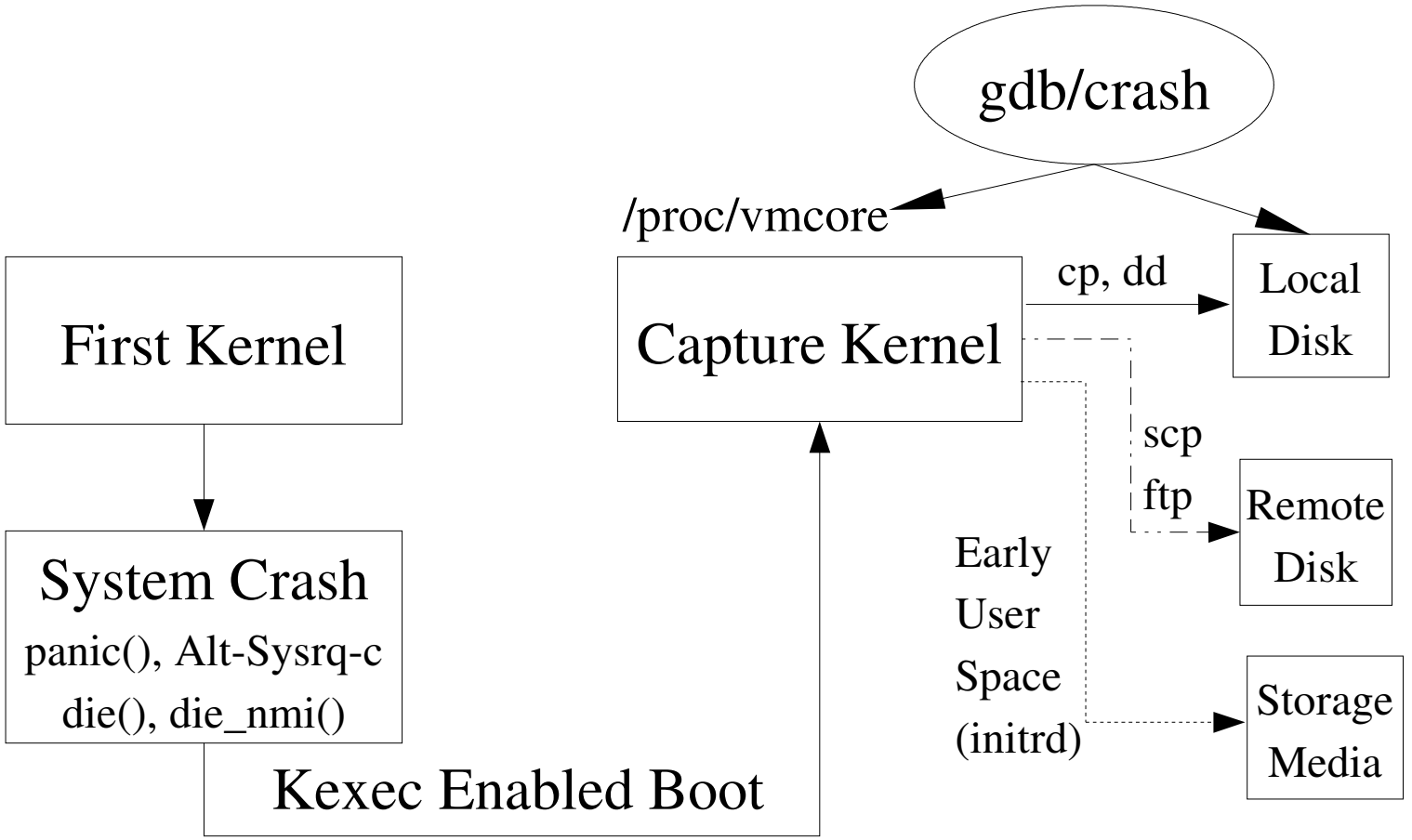
Design Goals of Kdump

- Simple and minimally invasive into the kernel code
- Highly reliable
- Available on most architectures
- Easy to maintain
- Flexibility in terms of dump contents and targets
  - Full dump or kernel-pages only dump
  - Dump to disk or across the network
- Ease of use

Design Overview

- A new kernel, often called capture kernel, is booted after the crash
- Previous kernel's memory is preserved
- Dump is captured from the context of capture kernel
- Kernel-to-kernel boot loader enables booting a new kernel after a crash
- Kexec is the underlying kernel-to-kernel boot-loader

Kdump Overview



Kdump - How To

- Loading the capture kernel

  kexec -p <kernel-image> --append=<options>

- Execution of capture kernel
  - *panic()*
  - *Alt-Sysrq-c*
  - *die()*
  - *die_nmi()*

Kdump – Capturing the Dump

- Accessing dump image in ELF Core format
  - /proc/vmcore
- Accessing dump image in linear raw format
  - /dev/oldmem

Kdump – Analysis Tools

- gdb
  - Virtual view of memory
  - Can debug linearly mapped region of memory
  - User space utility to regenerate ELF headers to create the ELF headers for vmalloc regions
- crash
  - Physical view of memory

Advantages

- Increased reliability
  - Dump is captured from a newly booted kernel
- Enhanced flexibility
  - Dump image can be saved to virtually any storage media supported by kernel
  - Filtering mechanism can be plugged in

Advantages Cont'd.

- Ease of use
  - Standard utilities can be used to save the dump image either locally or remotely
  - Standard analysis tools like gdb can be directly used for limited debugging

Limitations

- Devices are not shutdown/reset after a crash which might result in a driver initialization failure in capture kernel
- Non-disruptive dumping is not possible

Current Status

- i386, x86_64 and ppc64 ports are mainline
- ia64 port is in progress
- Available in Fedora™ Core 5

To-Dos

- Implementing a dump filtering mechanism
- Relocatable kernel for binary image unification
- Device Driver Hardening
- Crash tool improvement
  - A to-do list is available at http://people.redhat.com/anderson/crash.TODO.html

Downloads

- Kdump patches for kexec-tools and test reports are available at:

  http://lse.sourceforge.net/kdump/

Mailing Lists

- Kexec/Kdump
  - fastboot@lists.osdl.org
- Crash
  - crash-utility@redhat.com

Hands-On
(Kdump)

Configure Kdump (User Space)

- Download latest kexec-tools.
  - http://www.xmission.com/~ebiederm/files/kexec/kexec-tools-1.101.tar.gz
- Download and apply latest kdump patch for kexec-tools
  - http://lse.sourceforge.net/kdump/patches/kexec-tools-1.101-kdump9.patch
  - cd kexec-tools-1.101
  - patch -p1 < ../kexec-tools-1.101-kdump.patch

Configure Kdump (User Space) Cont'd.

- Build and install
  - ./configure
  - make install

Configure Kdump (Kernel)

- Two kernels need to be built in order to make this feature working
- Build first kernel
  - Change Makefile to give a version name (may be “2.6.17-1M”)
  - Enable "kexec system call" feature (in Processor type and features) (CONFIG_KEXEC=y)
  - Enable “Magic SysRq Key” (CONFIG_MAGIC_SYSRQ=y) (in Kernel Hacking/Kernel Debugging)
  - Enable “Compile the kernel with debug info”. (CONFIG_DEBUG_INFO=y)

Configure Kdump (Kernel) Cont'd.

- For simplicity, build appropriate disk, network and filesystem driver in kernel.
- Also enable support for initrd and LVM if required.
- make; make modules_install; make install
- Boot into first kernel with the command line parameter "*crashkernel=Y@X*". Use appropriate values for X and Y. Y denotes how much memory to reserve for the second kernel, and X denotes at what physical address the reserved memory section starts. Ex. “ crashkernel=64M@16M“
- Use kexec to reboot faster (If available in currently booted kernel)

kexec -l <kernel-image> --append=”`cat /proc/cmdline` crashkernel=64M@16M ”

Configure Kdump (Kernel) Cont'd.

- Build Second Kernel
  - Change Makefile to give a version name (may be “2.6.17-16M”)
  - Enable “High Memory Support” (4GB) (CONFIG_HIGHMEM4G=y)
  - Enable "kernel crash dumps" feature (in Processor type and features). (CONFIG_CRASH_DUMP=y)
  - Specify a suitable value for "Physical address where the kernel is loaded" (in Processor type and features). Typically this value should be same as X, start of reserved region. For example, 16 MB or 0x1000000. (CONFIG_PHYSICAL_START=0x1000000)

Configure Kdump (Kernel) Contd..

- Enable other required kernel options (disk, network, filesystem, initrd, and so on).
- Also enable support for initrd and LVM if required.
- make; make modules_install;

- Pre-load capture kernel
  - *kexec -p <capture kernel vmlinux image> --args-linux --append="root=<root-dev> init 3 irqpoll maxcpus=1"*

Notes:

Specify maxcpus=1 if capture kenrel is SMP.
Specify suitable kernel command options and initrd, if need be.

Invoke dumping

- Force Panic
  - Drop to execution level 3 (*init 3*)
  - *echo c > /proc/sysrq-trigger*
- System boots into capture kernel

Access/Save Dump Image

- Save Dump (/proc/vmcore)
  - *cp /proc/vmcore <destination directory>*
- Save Dump (/dev/oldmem)
  - *mknod /dev/oldmem c 1 12*
  - *dd if=/dev/oldmem of=<destination directory>*

Kdump in FC5

- Separate pre-compiled capture kernel
- Automation through scripts
  - Capture kernel is pre-loaded automatically
  - Dump is saved automatically

Kdump in FC5 Cont'd.

- Install following packages
  - *yum install kernel-kdump*
  - *yum install kexec-tools*
  - *yum install kernel-debuginfo*
- Enable kdump service
  - *chkconfig kudmp on*

Kdump in FC5 Cont'd.

- Reboot to the FC5 kernel with boot parameter "*crashkernel=64M@16M*"
- Kdump capture kernel gets loaded automatically
  - */etc/rc.d/init.d/kdump* init-script
- Upon system crash, the capture kernel boots and the dump image is saved in local disk
  - */var/crash/<date-time-stamp>/* directory

Kdump in FC5 Cont'd.

- Crash Analysis
  - *crash /usr/lib/debug/lib/modules/<kernel-version>/vmlinux /var/crash/<crash-dir>/vmcore*

Dump Analysis
(Crash)

Introduction to Crash

- A universal tool to analyze kernel crash dumps
- Supports various dump formats
  - diskdump
  - netdump
  - LKCD
  - Kdump
- Can examine live system's kernel internals through */dev/mem*

Crash: Build Process

- Download from Dave's site
  - http://people.redhat.com/anderson/
- Can be installed directly from binary RPM
- Can be built using source RPM
- Can be built from source tar ball

Crash: Invocation

- Needs two arguments if run on a dumpfile
  - kernel object file, vmlinux
  - dumpfile (/proc/vmcore for kdump)
  - *crash vmlinux vmcore*
- "vmlinux" should be built with “-g” C flag
  - CONFIG_DEBUG_INFO=y

Crash: Invocation Cont'd.

- If the dump was taken from a kernel built without debug info, rebuild the kernel with debug info
  - CONFIG_DEBUG_INFO=y
- Invoke crash in any of the following ways
  - *crash vmlinux vmlinux.dbg vmcore*
  - *crash System.map vmlinux.dbg vmcore*

Demo Problem

- Pre-load the capture kernel
- Build and insert the sample module
- Capture the dump and analyze the problem with "crash"

Legal Statement

Questions?