



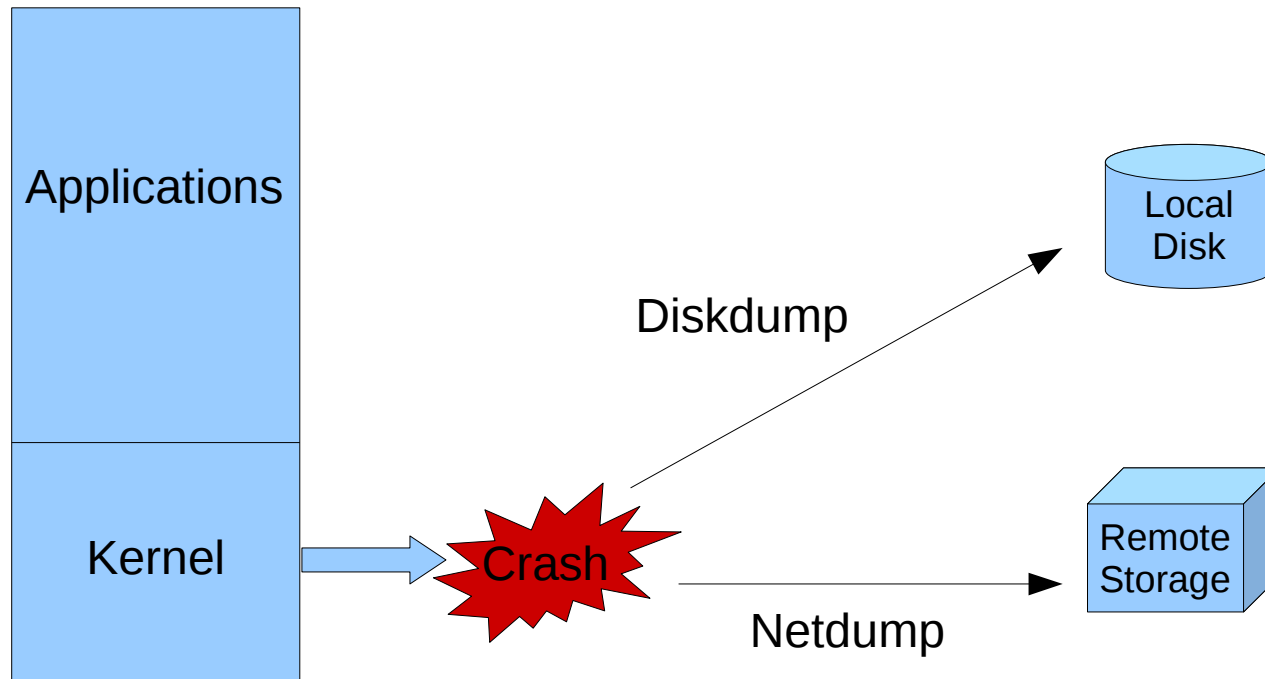
What's the Fuss About Fastboot and New Kernel Crash Dumping Mechanism

Vivek Goyal
Senior Software Engineer
RedHat

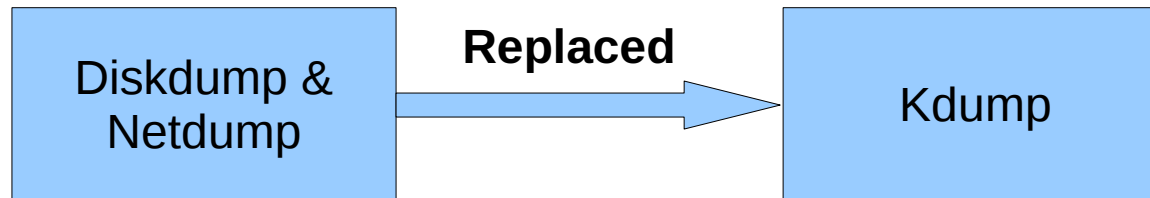
Agenda

- Kernel Crash dumping (RHEL4 and RHEL5)
- What changed and why change
- Fastboot/Kexec
- Kdump
- Relocatable kernel
- How to configure and use kdump
- Dump filtering
- Driver test matrix

Kernel crash dumping in RHEL4

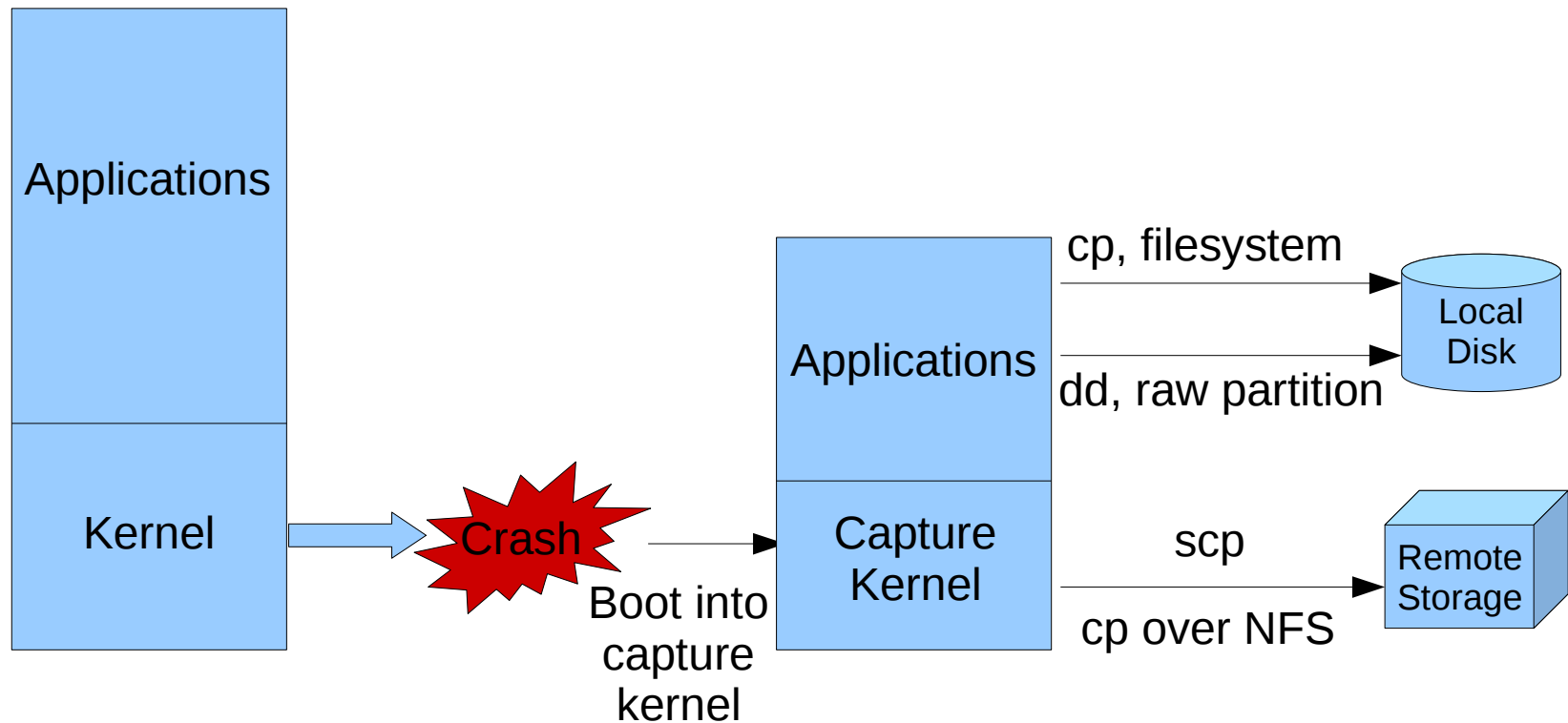


What changed in RHEL5

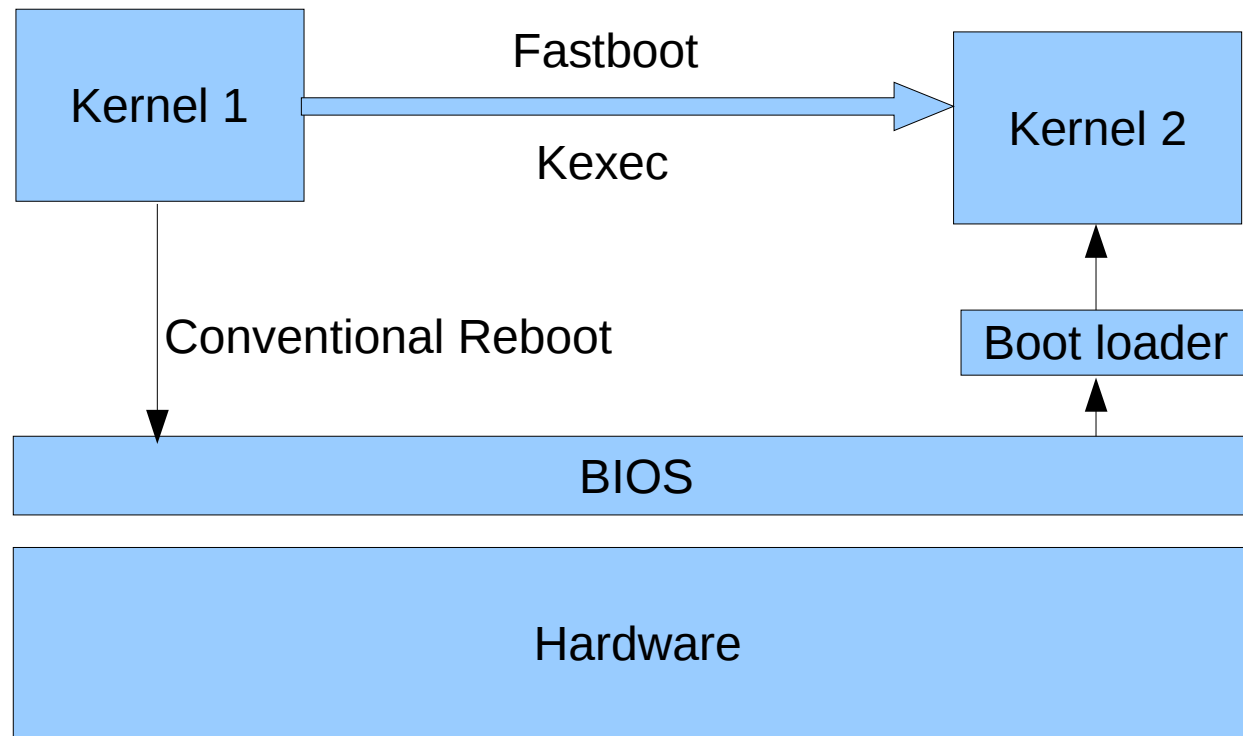


- Reliability
 - Don't trust a crashed kernel
- Flexibility
- Upstream solution
- Supported arch
 - x86, x86_64, ppc64, IA64

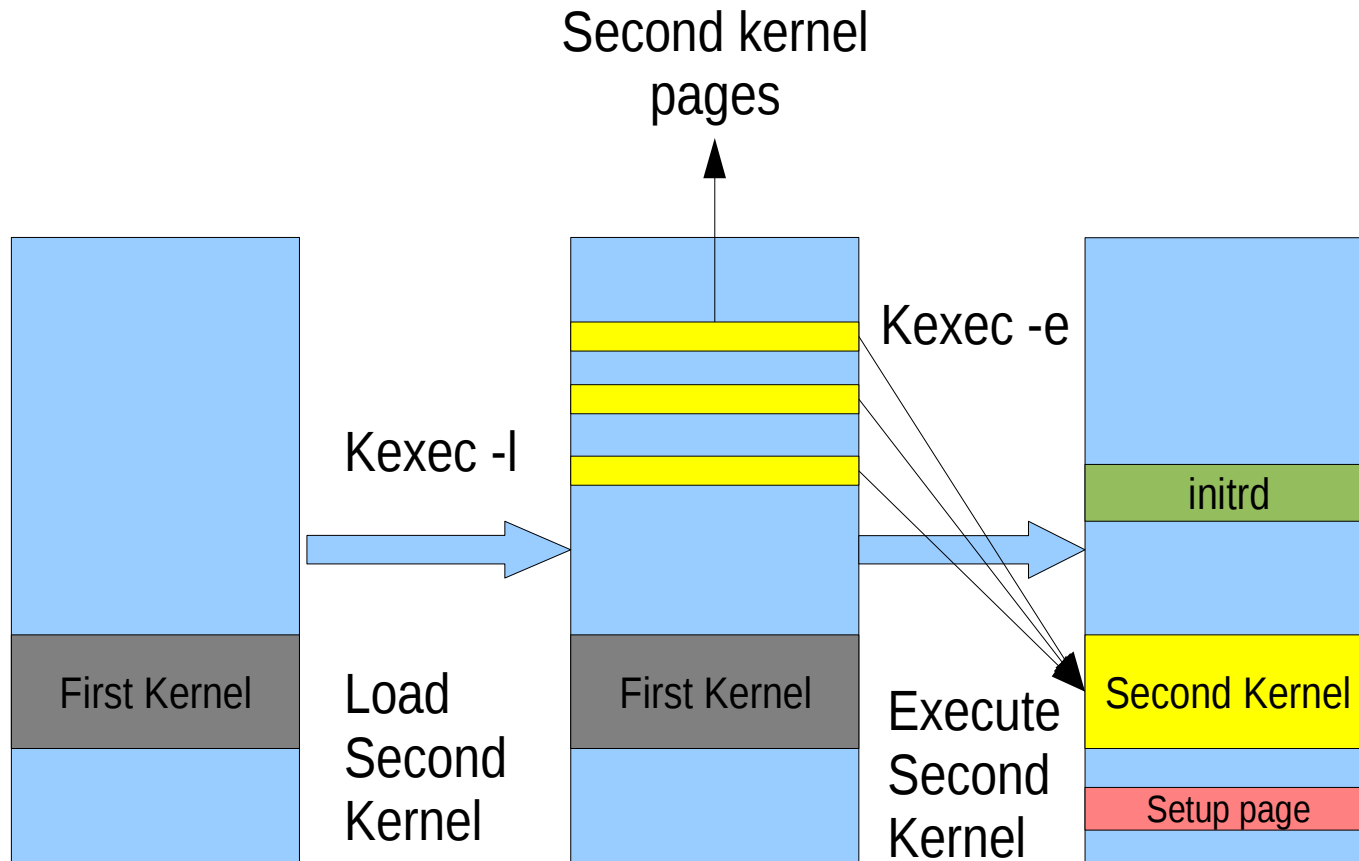
Kernel crash dumping in RHEL5



Fastboot/Kexec

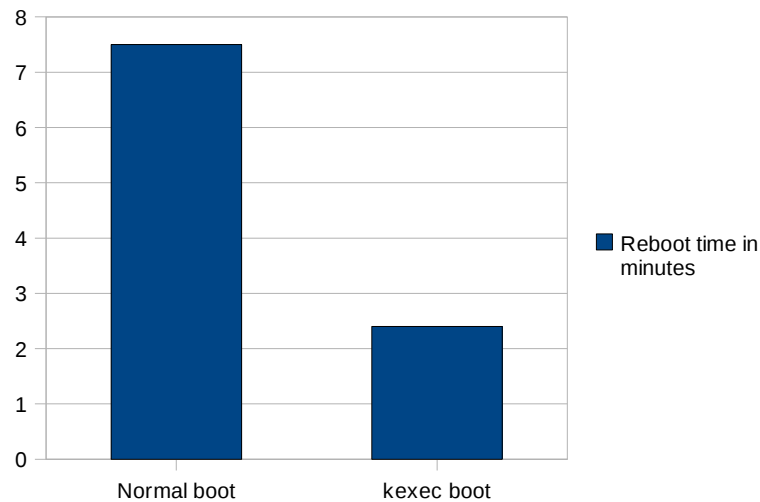


Kexec design



How fast is kexec?

- Test Hardware: x86_64, 64 processor, 128 GB RAM
- Measured reboot time
- Reboot time reduced by 70% on test system

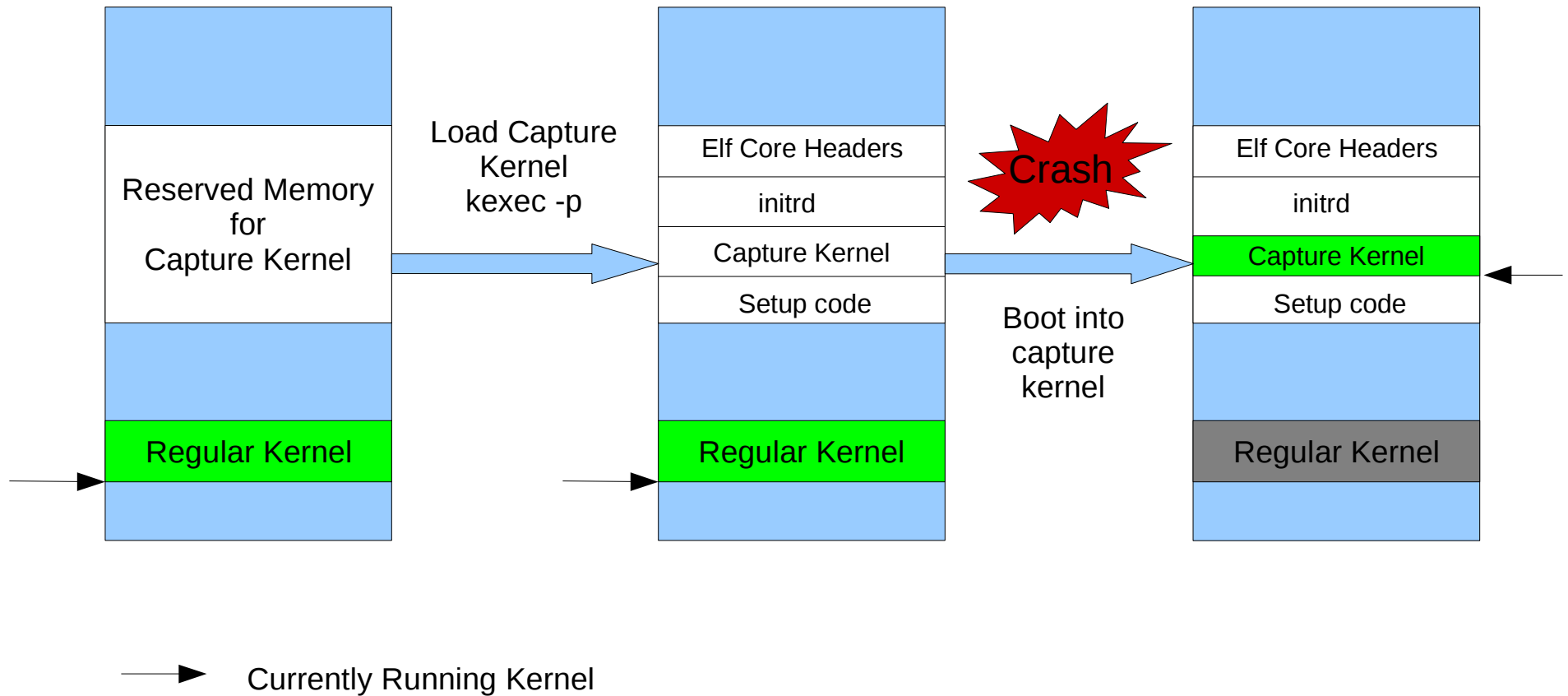


Normal Boot	7.5 minutes
Kexec Boot	2.2 minutes

How to use Kexec

- *yum install kexec-tools*
- Load Kernel
 - */sbin/kexec -l <kernel-to-load> --initrd=<initrd-to-load>
--command-line=<command-line>*
- *reboot*

Kdump design



Elf format dump file

- Kernel core exported through /proc/vmcore
 - Standard format
 - gdb can open the dump
- All memory chunks represented by PT_LOAD type headers
- All cpu states are captured by NT_PRSTATUS type Elf notes
- Standard tool can operate on /proc/vmcore to save it
 - cp, scp, dd etc.

Relocatable kernel

- Same kernel binary can run from different physical addresses
- Allows one to use regular kernel as capture kernel
- Currently i386, x86_64 and IA64 kernels are relocatable
- ppc64 uses a separate kernel binary as capture kernel
- x86
 - Retains relocation information
 - Performs relocation at run time
 - Kernel compile and run time virtual addresses are different
- x86_64
 - Kernel text region mappings are updated early
 - Kernel compile and run time virtual addresses are same

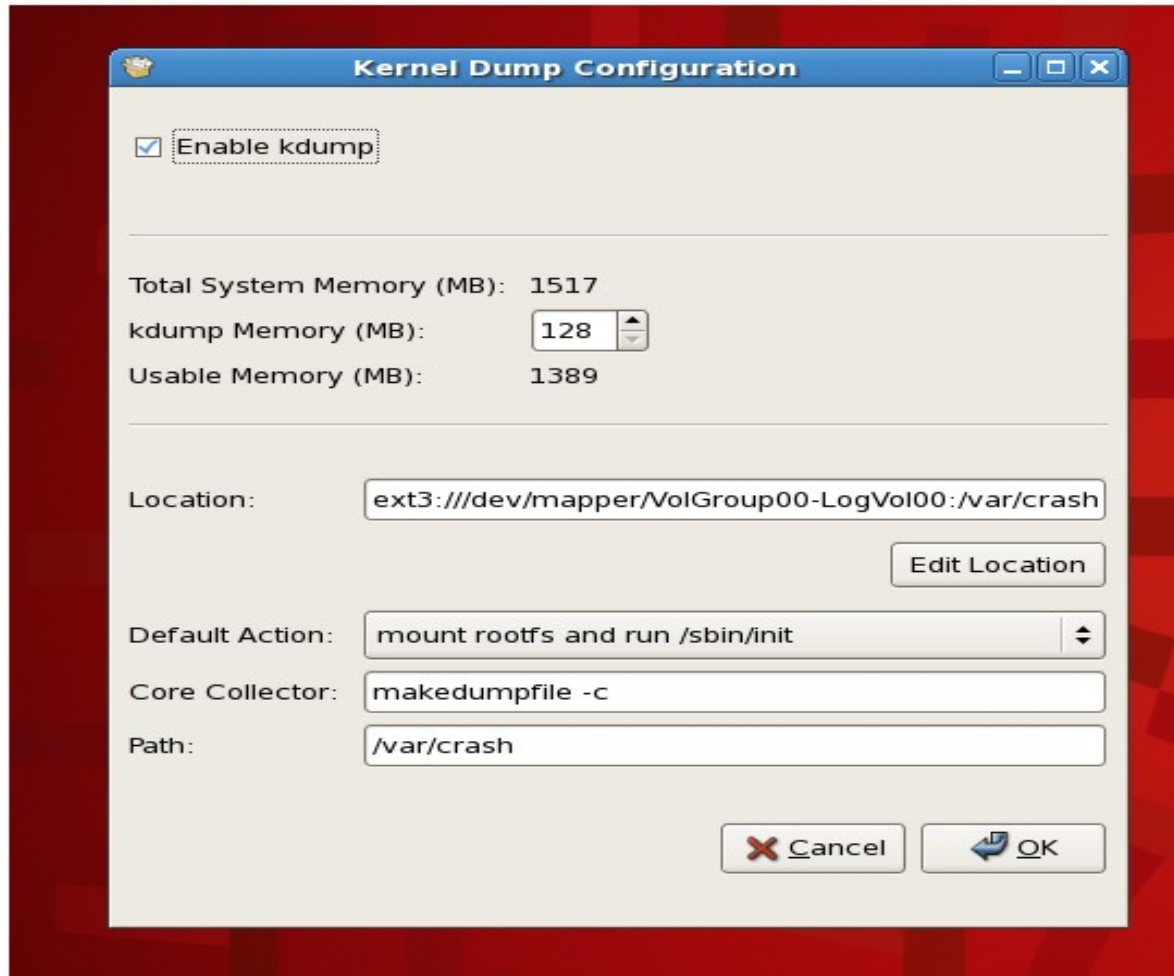
Enable kdump at installation

- Select “system-config-kdump” and “kexec-tools” packages
 - Optional package in “Base System”
- Enable kdump at first boot time
 - Specify amount of memory reserved for capture kernel
- Enable kdump service
 - *chkconfig kdump on*
 - Or use *system-config-kdump*

How to enable kdump later

- Install relevant packages
 - *yum install kexec-tools*
 - *yum install system-config-kdump*
- Reserve memory for capture kernel
 - Use *system-config-kdump*
- Reboot machine
- Enable kdump service
 - *chkconfig kdump on*
 - Or use *system-config-kdump*

Configuration: GUI interface



What is configurable

- Amount of memory to reserve for crash kernel
- Dump Destination
 - Local file-system
 - NFS
 - SCP
 - Raw partition dump
- Default Action
 - Reboot; halt; shell; mount root and run init
- Dump filtering Options
 - makedumpfile

Behind the scenes

- /boot/grub/menu.lst
 - Modified for `crashkernel=X@Y` parameter
- /etc/kdump.conf
 - Modified for rest of the options

Advance configuration

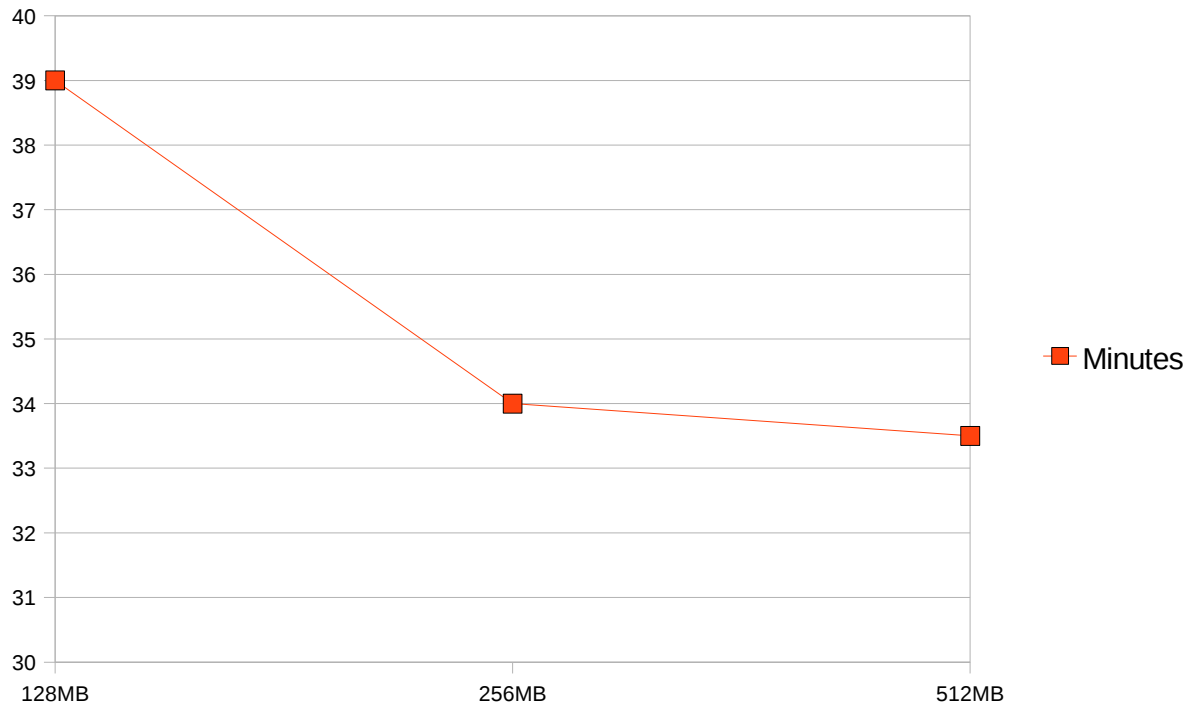
- More configuration options in `/etc/kdump.conf`
 - `extra_bins`
 - Load extra bin/scripts into `initrd`
 - `kdump_post`
 - Specify if some binary/scripts need to be run after saving dump. Handle success/failure.
 - `extra_modules`
- `/etc/sysconfig/kdump`
 - Various command line, kernel version related option
 - No need to touch it normally

How much memory to reserve?

- Primarily depends on architecture
 - 128 MB for x86 and x86_64
 - 256 MB for ppc64
 - 256 MB (small servers) or 512MB (big servers) for IA64

How fast is dumping?

- Data for a test system
 - RHEL5.2, x86_64, 64 processor, 128 GB RAM



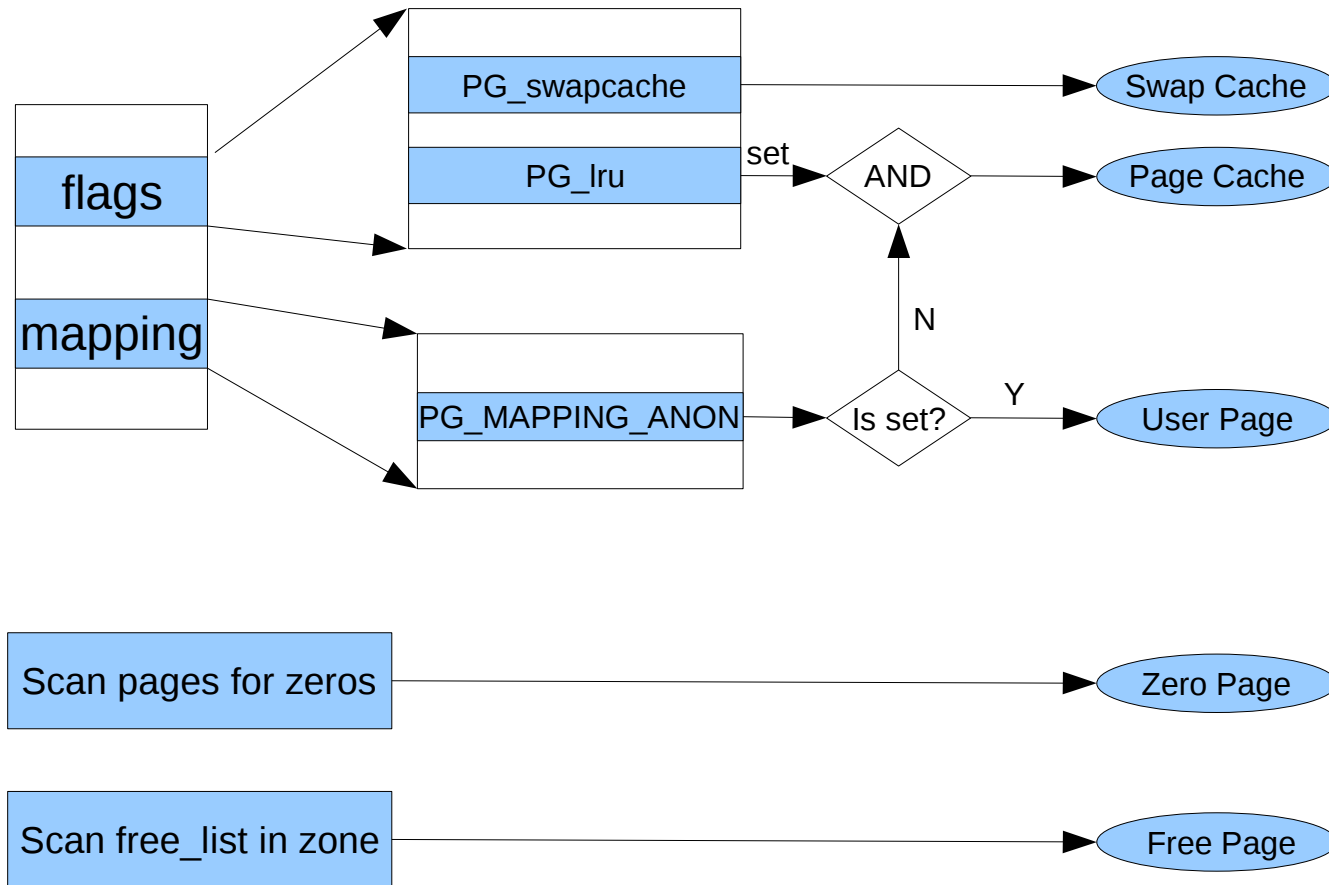
Dump filtering

- makedumpfile is the dump filtering tool
- All filtering takes place in user space
- Output Format
 - ELF format
 - Kdump compressed format
- Allows compression of output pages
- Multiple dump filtering levels

Filtering levels

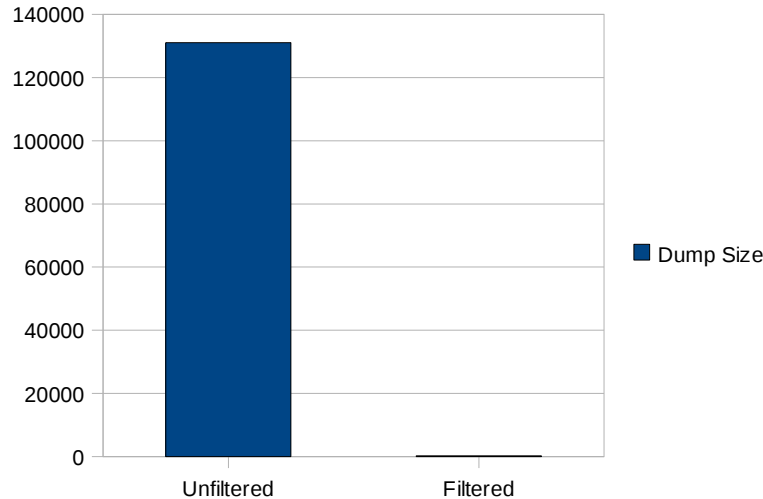
Dump Level	Zero Page	Cache Page	Cache Private	User Data	Free Page
0					
1	x				
2		x			
4		x	x		
8				x	
16					x
31	x	x	x	x	x

Filtering design

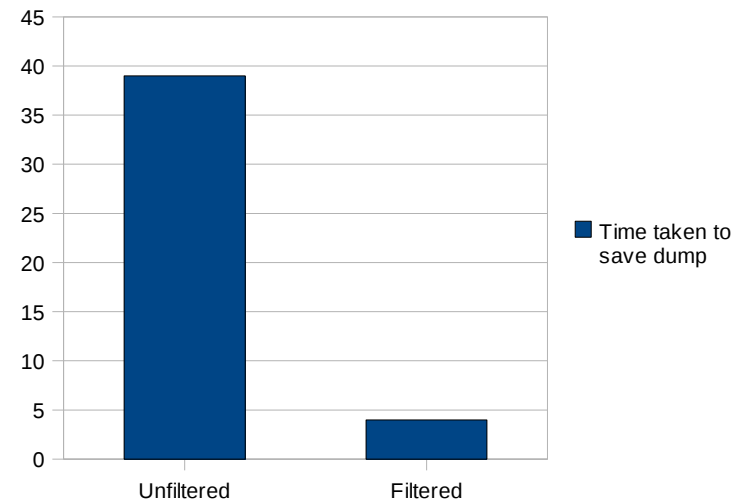


How effective is filtering?

- Freshly booted system; mostly free pages
- 128 MB reserved for second kernel; Filtering level highest



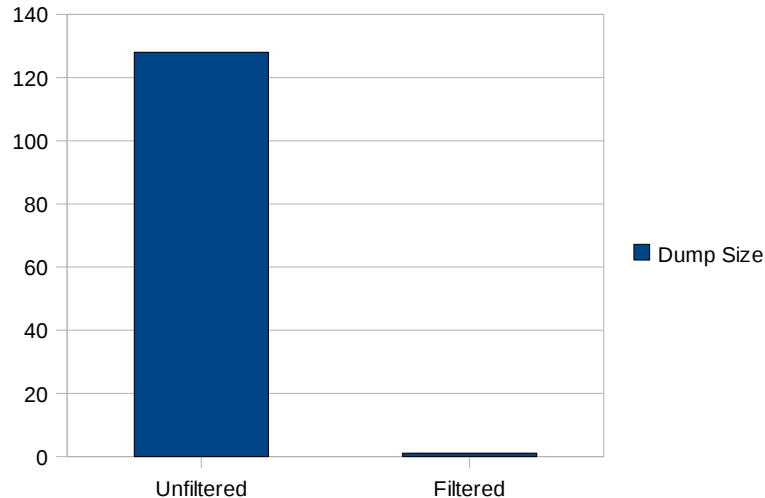
Unfiltered	128GB
Filtered	234MB



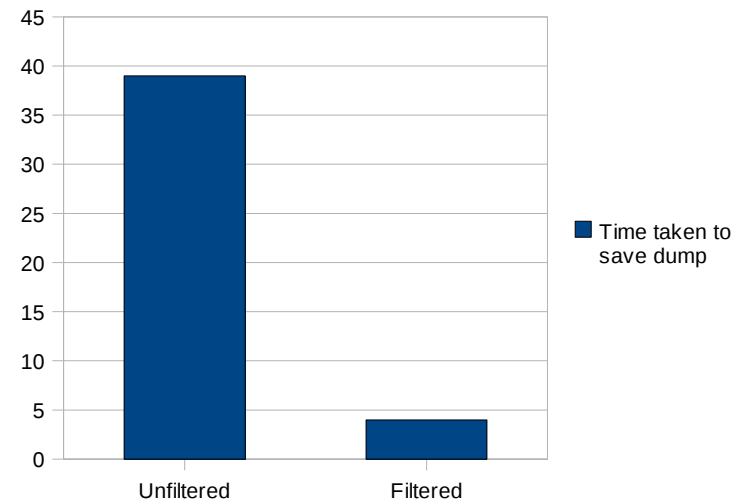
Unfiltered	39 Minutes
Filtered	4 Minutes

How effective is filtering? Contd.

- Wrote a huge file with random numbers to fill page cache
- 128 MB reserved for second kernel; Filtering level highest



Unfiltered	128GB
Filtered	1.08 GB



Unfiltered	39 Minutes
Filtered	5 Minutes

Is this the perfect world

- Best effort is made to capture the dump
- Device driver initialization issues
 - Software reset capability
 - Reset device at initialization if in capture kernel

Driver test matrix (storage)

Driver/Controller	x86	X86_64	ppc64	IA64
megaraid_sas				
megaraid_mbox				
mptfusion				
mptspi				
mptsas				
sym53c8xx				
lpfc				
cciss				
serveraid				
ipr				
adpxxxx				
aic79xx				
aacraid				
aic94xx				
stex				
qla1280				

Driver test matrix (networking)

Driver/Controller	x86	X86_64	ppc64	IA64
e100				
e1000				
e1000e				
tg3				
q802.1/bonding				
bnx2				

Mailing lists

- Kexec, Kdump or makedumpfile issues
 - kexec@lists.infradead.org
- “Crash” Issues
 - crash-utility@redhat.com

Questions?

Thank You