

Re: [Fastboot] [RFC] Kdump: Dump Capture Mechanism

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-03/3554.html>

From: Eric W. Biederman (*ebiederm_at_xmission.com*)

Date: 03/10/05

To: Vivek Goyal <vgoyal@in.ibm.com>

Date: 10 Mar 2005 00:50:28 -0700

Vivek Goyal <vgoyal@in.ibm.com> writes:

> *Hi,*
>
> *Well this discussion has been going on for quite sometime now that*
> *what's the best way to capture the dump? There seems to be two lines of*
> *arguments.*
>
> *Export ELF view through /proc/vmcore*
> -----
> *This basically involves retrieving saved core image headers and*
> *exporting those through /proc/vmcore interface. Further user space*
> *applications can be built on top of it to do advanced processing.*
>
> *Do Everything in user space*
> -----
> *The whole idea is that do all the processing from user space (preferably*
> *from ramdisk or so).*
>
> *When it comes to requirements, Distros and developers seem to be having*
> *somewhat different requirements.*
>
> *Distros:*
> -----
> *- Fully automate the dump generation/capture process.*
> *- Configure everything in advance (like, dump storage location).*
> *- Upon crash, store dump image at pre-configured location and reboot*
> *into production kernel ASAP.*
>
> *Developers:*
> -----
> *- Keyword is simple and easy to use solution.*
> *- Should work well in a development environment where, not necessarily*
> *all the components (user space, kernel space) are in perfect harmony*
> *and things are yet to be stabilized.*

Linux–Kernel: Re: [Fastboot] [RFC] Kdump: Dump Capture Mechanism

- >
- > *IMO, exporting /proc/vmcore is a good idea. It offers wide variety of*
- > *choices to both developers and distros.*
- >
- > – *It provides the basic dump capturing mechanism in kernel.*
- >
- > – *Developers can store the dump image locally (cp) or transfer it over*
- > *network (scp, ftp) using standard utilities and don't have to deal*
- > *with additional user space utilites specifically designed for this*
- > *purpose.*

No just a magic kernel interface.

And with /sbin/kdump as written this already works.

You just need to use a pipe.

```
kdump |ftp – someone@somewhere.net
```

- > – *Developers can directly run gdb on /proc/vmcore generated image and*
- > *do the limited debugging without need of any other dump*
- > *capture/analysis utility.*

There are a lot of disharmonies that can happen here. 32bit dump kernel 64bit dumped kernel etc. Or vice versa. Those kinds of things can cause problems if you are not careful.

And even worse you can't directly use gdb unmodified on 32bit systems because it will be a 64bit core dump, with unreliable virtual addresses.

- > – *Distros can build additional fully automated dump saving solutions on*
- > *top of /proc/vmcore. Be it a init script or a custom initial ramdisk*
- > *or something else.....*
- >
- > *So the whole idea is, that /proc/vmcore and user space solutions can co–*
- > *exist. And let the user/distros choose between these based on their*
- > *requirements.*
- >
- > *I was planning to implement /proc/vmcore. Do you have any comments or*
- > *suggestions?*

My gut reaction, is a server that talks the gdb–stubs protocol to export the core dump in read–only mode has about the same advantages, and doesn't require a kernel patch. Plus it does not require an thorough audit. I freely admit that having to copy all the entire core dump can be overkill.

Anything in the kernel that we use for crash dump capture must be hardened, against all manner of insanity. Placing something in user space hugely limits the worst that can happen, facilitates debugging and facilitates auditing for reliability, enormously.

Linux-Kernel: Re: [Fastboot] [RFC] Kdump: Dump Capture Mechanism

Every line of code that does not go into the kernel now, saves days of debugging later. Every kernel abomination I have had to look at has been because too much code was in the kernel. lustre/openib gen1/drbd. I like running my slow path code in the virtual machine that is a process. So many problems are ruled out immediately.

Without /proc/vmcore a crash capture kernel will simply be a robust kernel that can run at a non-default address. With /proc/vmcore we have transformed the kernel into a special purpose tool, that really can't be used for other things.

So if we can do it in user space let's do it there.

For simplicity I do support putting it all in one user space package so only one set of tools needs to be learned.

If you are going to do something, I recommend making it it's one in kernel filesystem. That is the only semi-sane way I can see to wrap magic /proc/vmcore.

Eric

p.s. I still remember the several of the kludges in the current crashdump-* patches. They don't set expectations for a robust, clean, and minimal kernel implementation.

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>