Intro (Page 1 and 2):

Hello, everyone. I'm Elena, a member of the Globalization (G11N) SIG in openEuler community. We plan to make a series of mini course videos. In these mini courses, we'll be sharing techs and knowledge about openEuler OS and hope you'll find them helpful. Today, in the first episode, I'm going to talk about kdump, a kernel crash dumping mechanism.

Page 3

Have you ever wondered what would happen if a sudden crash occurred while a system was running? Of course, you'd first try to fix the problem using the temporary protection mechanism watchdog or something similar. But what next? Well, it's also important to copy the memory image of the crashed environment to the file system. That way we can then debug the dump file to find the cause of the kernel crash.

Page 4

A variety of crash dumping solutions have been developed over a period of time, such as Linux Kernel Crash Dump (LKCD), mini kernel dump (MKdump), tough dump and, eventually, kdump.

Page 5

Kdump provides a reliable dump generation and capturing mechanism. It is simple, easy to configure, and provides a great deal of flexibility in terms of dump device selection, dump saving mechanism, and plugging-in filtering mechanism. Kdump supports x86, x86\_64, arm, arm64, ppc, s390, as well as various other architectures.

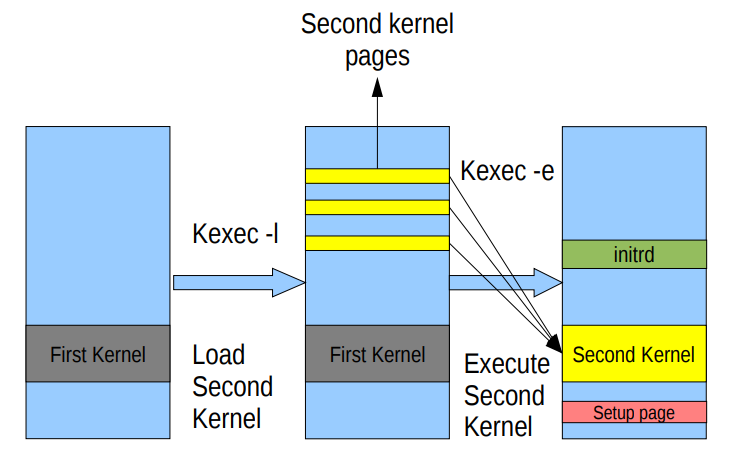
Page 6

Kdump involves two kernels: The first kernel, or production kernel, is the kernel we normally work with. The second kernel, or capture kernel, is the kernel used for collecting crash dumps.

When a kernel crash occurs, kdump uses kexec to boot into a second kernel and creates a memory image (vmcore) that can help us determine the cause of the crash.

Page 7

Here's a diagram to better explain.

 (Source: Vivek Goyal)

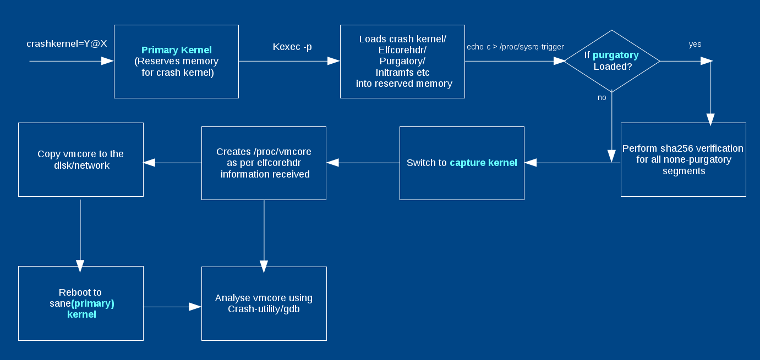
Kexec is a mechanism of the Linux kernel that allows a new kernel to be booted from the one currently running. The difference between a standard system boot and a kexec boot is that the hardware initialization normally performed by the BIOS or firmware (depending on architecture) is not performed during a kexec boot. This can save developers a lot of time.

Page 8

Kexec consists of two components: kernel space and user space. The kernel space implements a system call kexec\_load() which facilitates the pre-loading of a new kernel. This new kernel can be executed later by reboot. The user space, called kexec-tools, loads the second kernel and reboot to the loaded kernel in case of a system crash or panic.Page 9

So, how does this whole kdump thing work exactly?

1. The production kernel reserves a chunk of memory.
2. The capture kernel is pre-loaded in the reserved memory.
3. A kernel oops or panic occurs.
4. The system boots into the second kernel.
5. The system creates a **/proc/vmcore** file, which indicates the memory image of the panicked kernel.
6. Copy vmcore for future analysis.
7. Reboot to the production kernel.
8. Analyze the vmcore file.



Page 10

Next, I'd like to share some basics of kdump on openEuler.

Kdump is installed on popular Linux operating systems by default, so that we can simply execute scripts to use the kdump service. To start, stop, or query the status of kdump, run the following command: **systemctl *start|stop|status* kdump**.

On openEuler, there are two kdump configuration files. One is **/etc/kdump.conf**, where we can use **path /var/crash** to specify the directory storing the vmcore file, and use **keep\_old\_dumps** to specify the number of vmcores to store (**keep\_old\_dumps -1** indicates only the vmcore of the latest panic is stored). The other configuration file is **/etc/sysconfig/kdump**. In this file, the **KDUMP\_COMMANDLINE\_APPEND** option appends arguments to the current command line.

Page 11 (Demo)

Let's watch a demo of how kdump works on openEuler.

(Demo)

Page 12

OK, we saw how kdump works, but the question now is why is it necessary to have kdump installed in a Linux environment?

Well, first, kdump is flexible. Because dump is captured from a newly booted kernel, it supports network dumping to a wide range of devices, including local drives, NFS areas, CIFS shares or FTP and SSH servers.

It's also highly reliable. The crash dump is captured from the context of a freshly booted kernel but not from the context of the crashed kernel.

What's more, kdump is efficient. We can filter out extraneous pages and compress the dump, and so handle large dumps in a short time.

Lastly, it's easy to use. When writing dumps over a network, we can use existing file system facilities to share dump space without special preparations.

Page 13

Like any other solution, kdump also has its limitations. First, kdump cannot be used for issues that occur before kdump is initialized, for example, for early boot problems. For such cases, you'll have to use a standalone dump tool. Second, devices are not shut down or reset after a crash, which might result in driver initialization failure in capture kernel. Third, non-disruptive dumping is impossible.

Ending

Well, that's all the basics of kdump I want to share with you today. We'll do more similar technical presentations, so stay tuned for more. Bye for now!