1 Dump page table of processes in the linux kernel

1.1 Why?

As we know, when a process is running, it uses virtual address to access data. Then, the MMU (Memory Management Unit) transforms the virtual address into physical address, which is used to access the physical memory.

The HMTT3 system can collect physical memory addresses directly by monitoring the signals on the DIMM. But for application optimization, it will be more useful if we can collect the virtual memory trace as well, because the virtual address is more correlative with application semantic.

In Linux, each process is maintained a page table for transforming a virtual address into physical address. In our situation, we need to do the contrary transformation: give a physical address monitored by HMTT3, lookup the page table to get the corresponding virtual address. One thing to note: the page table will change during the process lifetime, it will malloc or free entries dynamically, since the process might malloc or free the memory space dynamically in heap space. So we need to monitor the page table updates in the kernel during the process lifetime to catch all page table changes.

1.2 How?

We overwrite each function in kernel which is corresponded to update page table (e.g set a pte) and record each update operation, so we can get the page table update trace as well (we will refer it to kernel trace). Figure 1 shows the dump kernel trace framework.

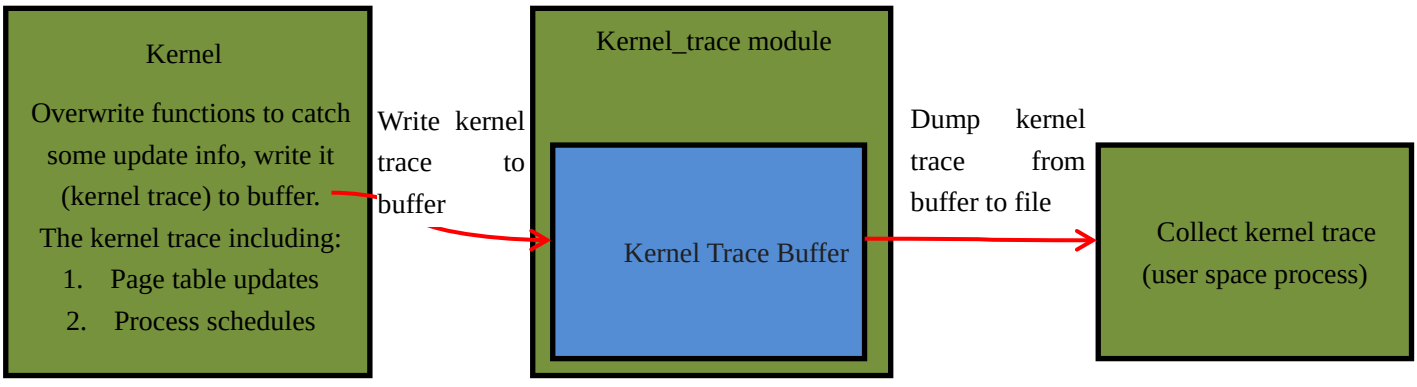

Figure 1. Dump kernel trace framework

We reserve some physical memory space for kernel trace buffer. The kernel will write each kernel trace into the buffer, meantime, a process named collect_trace will periodically dump kernel trace from buffer into a file.

1.3 About **source code**

The source code we provide has 4 components:

- Kernel_modification: including modified kernel files to overwrite page table updates and process schedules,
- Kernel_trace_hmtt3: it is a module, make an interface for kernel to dump kernel trace into buffer, and control begin/stop collecting kernel trace when receive commands from user.
- Control_and_collect: including begin_hmtt_trace and stop_hmtt_trace app which are used to send commands to kernel_trace_hmtt3 module. And collect_kernel_trace is used to dump kernel trace from buffer into a .kt file
- Kernel_trace_analyze: optional, analyze the kernel trace and do some statistic work for debug, it is a useful start to learn how analyze the kernel trace we get.

There are 3 files needed to be updated for dumping page table in kernel. They are: pgtable.c, pgtable.h, pgtable_64.h. We use x86_64 arch, so the path in kernel source for each is:

- arch/x86/mm/pgtable.c
- arch/x86/include/asm/pgtable.h
- arch/x86/include/asm/pgtable_64.h

To get process schedule trace, the schedule() function which is in kernel/sched.c need to be overwrite.

2 Setup

1. Replace the files into the Linux kernel (2.6.32.18) with the files in the kernel_modification. Suppose now we are in the kernel source root dir, the files path are:
   - arch/x86/mm/pgtable.c
   - arch/x86/include/asm/pgtable.h
   - arch/x86/include/asm/pgtable_64.h
   - kernel/sched.c
2. Rcompile the kernel and install it in the system, we will use this modified kernel to dump kernel trace (including page table updates and process schedules). Before reboot with the new kernel, we need to reserve some physical memory space, see Step 3.
3. Reserve some physical memory space for HMTT3 configuration and kernel trace buffer: In CentOS 5.3, we modify the /boot/grub/menu.lst file, find the kernel and add “mem = available_size” in the end. In our system for example, the total physical memory size is 4GB, we want to observe the highest 0.25GB memory space and leave 3.75GB for OS normal usage, as showed in figure2. The kernel we use is like:

   ```
   # title CentOS (2.6.32.18-pt_addr_HMTT3)
   #          root (hd0,0)
   #            kernel /boot/vmlinuz-2.6.32.18-pt_addr_HMTT3 ro root=LABEL=/
   ```

rhgb quiet mem=4864M

initrd /boot/initrd-2.6.32.18-pt_addr_HMTT3.img

After modify the grub config, reboot the new kernel.

4. Change Dir to “kernel_trace_hmtt3”, Make the kernel_trace module and install it:

   # cd /xx/kernel_trace_hmtt3

   # make

   # ./kernel_trace_hmtt3_load

   First you need to make the file “kernel_trace_hmtt3_load” executable, this can be done by:

   # chmod +x kernel_trace_hmtt3_load

   If install module successfully, you will see the 1 new module in the system with lsmod cmd: kernel_trace_hmtt3. And 2 new char dev under the /dev dir: /dev/config_space and /dev/ kernel_trace_hmtt3.

5. Change Dir to control_and_collect, make the 3 source files:

   # cd ../control_and_collect

   # make

   It will produce 3 executable application:

   - begin_hmtt_trace: use to begin dumping kernel trace
   - stop_hmtt_trace: use to stop dumping kernel trace
   - collect_kernel_trace: use to dump kernel trace from reserved buffer to file

6. collect kernel trace:

   # ./collect_kernel_trace tmp 60

   The first parameter is the filename and the second parameter is the time interval to check whether the buffer is valid to dump. Because we have not begin dumping the kernel trace, the output is nothing (no data has written to the kernel trace buffer), the readptr and writeptr are 0.

7. Open another shell and begin dump kernel trace

   #cd /xx/control_and_collect

   # ./begin_hmtt_trace 2

8. Run an application to be collected kernel trace and memory trace, like SPECCPU2006

   # runspec xxx

9. After the application finished, stop dump kernel trace

   # cd /xx/control_and_collect

   #./stop_hmtt_trace 2

10. Terminate the collect kernel trace application, just type “Ctrl + c”. The kernel trace file is “tmp.kt”

11. Analyze the kernel trace file:

    # cd ../ kernel_trace_analyze

    # make

    # ./kernel_trace_analyze ../control_kernel_trace_hmtt3/tmp.kt

12. Remove the kernel trace module:

    # cd /xx/kernel_trace_hmtt3

    # ./ kernel_trace_hmtt3_unload

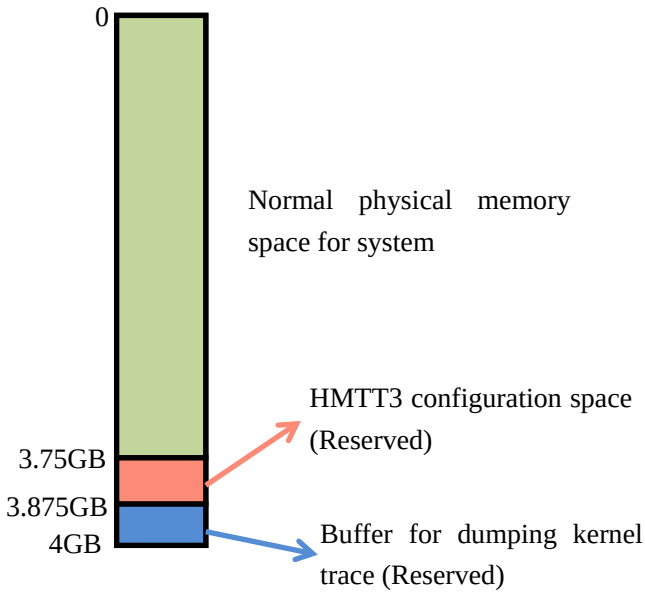

Figure 2. Physical memory space usage

3 Contact

If you have any questions about using or understanding the source codes, please feel free to contact Licheng Chen, the email is:

chenlicheng@ict.ac.cn