Practicing Linux Crash/Panic Issue on Production and Cloud Server

Gavin Guo

Technical Lead - Sustaining Engineering

gavin.guo@canonical.com

Wechat

2019 Shanghai Open Source Summit China

Enterprise Case Study

Migrating KSM page causes the VM lock up as the KSM page merging list is too large

https://bugs.launchpad.net/ubuntu/+source/linux/+bug/1680513

Case Description

After **numad** is enabled and there are several VMs running on the same host machine, the **softlockup** messages can be observed inside the **VMs' dmesg**.

```
CPU: 3 PID: 22468 Comm: kworker/u32:2 Not tainted 4.4.0-47-generic
#68-Ubuntu
Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS
Ubuntu-1.8.2-1ubuntu1 04/01/2014
Workqueue: writeback wb_workfn (flush-252:0)
[<ffffffff81104388>] smp_call_function_many+0x1f8/0x260
[<ffffffff810727d5>] native_flush_tlb_others+0x65/0x150
[<ffffffff81072b35>] flush_tlb_page+0x55/0x90
```

Investigation on the VM side

This one seems a known issue. The bug is proactively handled by Linus when Dave Jones[3] issued the bug which happened on the bare metal machine. Tinoco[2] also found the bug in the nested KVM environment which happened when the IPI is sent out in the VCPU and it seems the problem coming from the LAPIC simulation of VMX. Chris Arges also involved in the debugging process and the debugging patch was given out by the Ingo Molnar, then Chris added some hacks to print out the debugging information. Unfortunately, after a long investigation, the root cause is still unknown.

[1]. smp/call: Detect stuck CSD locks https://patchwork.kernel.org/patch/6153801/

[2]. smp_call_function_single lockups https://lkml.org/lkml/2015/2/11/247

[3]. frequent lockups in 3.18rc4 https://lkml.org/lkml/2014/11/14/656

Investigation on the VM side

I've prepared a hotfix kernel which would resend the IPI and print out the information when the softlockup happens. **Unfortunately**, the hotfix kernel **doesn't** print out the **error message**. **That means my original thoughts are incorrect!**

The hotfix kernel source:

http://kernel.ubuntu.com/git/gavinguo/ubuntu-xenial.git/log/?h=sf000103690-csd-lock-debug

Host Machine - Hung task Backtrace

As I cannot find the clue inside the VMs, then try to investigate the host side.

Host Machine - Hung task Backtrace

```
# ksmd
crash> bt 615
PID: 615 TASK: ffff881fa174a940 CPU: 15 COMMAND: "ksmd"
#0 [ffff881fa1087cc0] __schedule at ffffffff818207ee
#1 [ffff881fa1087d10] schedule at ffffffff81820ee5
#2 [ffff881fa1087d28] rwsem_down_read_failed at ffffffff81823d60
#3 [ffff881fa1087d98] call_rwsem_down_read_failed at ffffffff813f8324
#4 [ffff881fa1087df8] ksm_scan_thread at ffffffff811e613d
#5 [ffff881fa1087ec8] kthread at ffffffff810a0528
#6 [ffff881fa1087f50] ret_from_fork at ffffffff8182538f
```

Host Machine - Hung task Backtrace

```
# khugepaged
crash> bt 616
PID: 616 TASK: ffff881fa1749b80 CPU: 11 COMMAND: "khugepaged"
#0 [ffff881fa108bc60] __schedule at ffffffff818207ee
#1 [ffff881fa108bcb0] schedule at ffffffff81820ee5
#2 [ffff881fa108bcc8] rwsem_down_write_failed at ffffffff81823b32
#3 [ffff881fa108bd50] call_rwsem_down_write_failed at ffffffff813f8353
#4 [ffff881fa108bda8] khugepaged at ffffffff811f58ef
#5 [ffff881fa108bec8] kthread at ffffffff810a0528
#6 [ffff881fa108bf50] ret_from_fork at ffffffff8182538f
```

Host Machine - Hung task Backtrace

```
# qemu-system-x86

crash> bt 12555
PID: 12555 TASK: ffff885fa1af6040 CPU: 55 COMMAND: "qemu-system-x86"
#0 [ffff885f9a043a50] __schedule at ffffffff818207ee
#1 [ffff885f9a043aa0] schedule at ffffffff81820ee5
#2 [ffff885f9a043ab8] rwsem_down_read_failed at ffffffff81823d60
#3 [ffff885f9a043b28] call_rwsem_down_read_failed at ffffffff813f8324
#4 [ffff885f9a043b88] kvm_host_page_size at ffffffffc02cfbae [kvm]
#5 [ffff885f9a043ba8] mapping_level at ffffffffc02ead1f [kvm]
#6 [ffff885f9a043bd8] tdp_page_fault at ffffffffc02f0b8a [kvm]
#7 [ffff885f9a043c50] kvm_mmu_page_fault at ffffffffc02ea794 [kvm]
#8 [ffff885f9a043c80] handle_ept_violation at ffffffffc01acda3 [kvm_intel]
#9 [ffff885f9a043cb8] vmx_handle_exit at ffffffffc01afdab [kvm_intel]
#10 [ffff885f9a043d48] vcpu_enter_guest at ffffffffc02e026d [kvm]
#11 [ffff885f9a043dc0] kvm_arch_vcpu_ioctl_run at ffffffffc02e698f [kvm]
#12 [ffff885f9a043e08] kvm_vcpu_ioctl at ffffffffc02ce09d [kvm]
#13 [ffff885f9a043ea0] do_vfs_ioctl at ffffffff81220bef
#14 [ffff885f9a043f10] sys_ioctl at ffffffff81220e59
```

Host Machine - Hung task Backtrace

We can see that the previous three tasks are waiting on the **mmap_sem**. The most interesting part is the backtrace of **numad**:

The disassembly analysis of numad call stack

```
crash> bt 2950
#1 [ffff885f8fb4fb78] smp_call_function_many
#2 [ffff885f8fb4fbc0] native_flush_tlb_others
#3 [ffff885f8fb4fc08] flush_tlb_page
#4 [ffff885f8fb4fc30] ptep_clear_flush
#5 [ffff885f8fb4fc60] try_to_unmap_one
#6 [ffff885f8fb4fcd0] rmap_walk_ksm
#7 [ffff885f8fb4fd28] rmap_walk
#8 [ffff885f8fb4fd80] try_to_unmap
#9 [ffff885f8fb4fdc8] migrate_pages
#10 [ffff885f8fb4fe80] do_migrate_pages
```

KSM merge list extraction

I've tried to **disassemble** the code and finally find the stable_node->hlist is as long as 2306920 entries(**Around 9.2GB memory merged into one page**).

rmap_item list(stable_node->hlist):

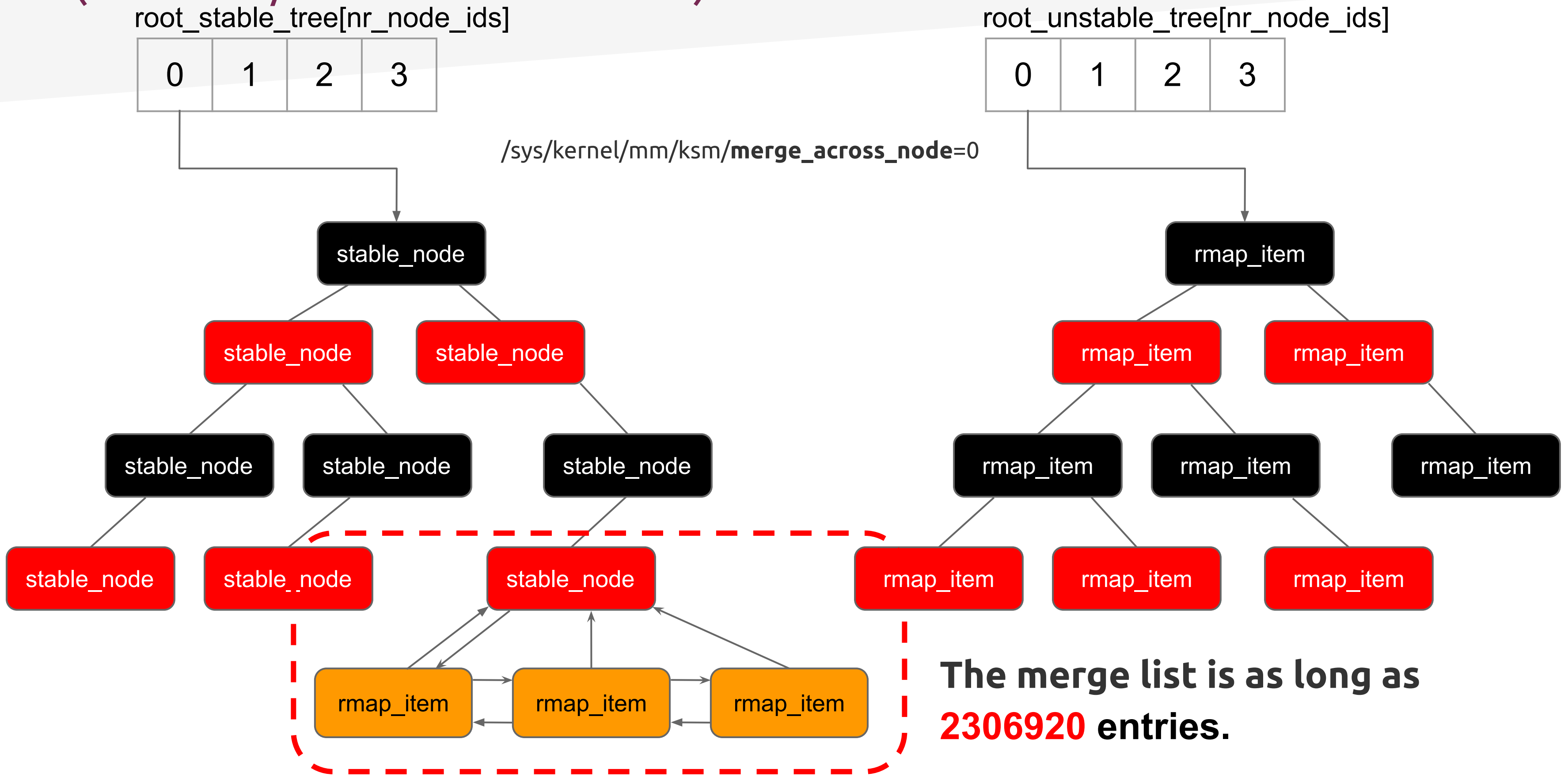
stable_node: 0xffff881f836ba000 stable_node->hlist->first = 0xffff883f3e5746b0

```
struct hlist_head {
    [0] struct hlist_node *first;
}
```

```
struct hlist_node {
[0] struct hlist_node *next;
[8] struct hlist_node **pprev;
}
```

```
crash> list hlist_node.next 0xffff883f3e5746b0 > rmap_item.lst
$ wc -l rmap_item.lst
2306920 rmap_item.lst
```

Introduction to the KSM Stable Tree (Stable/Unstable tree)

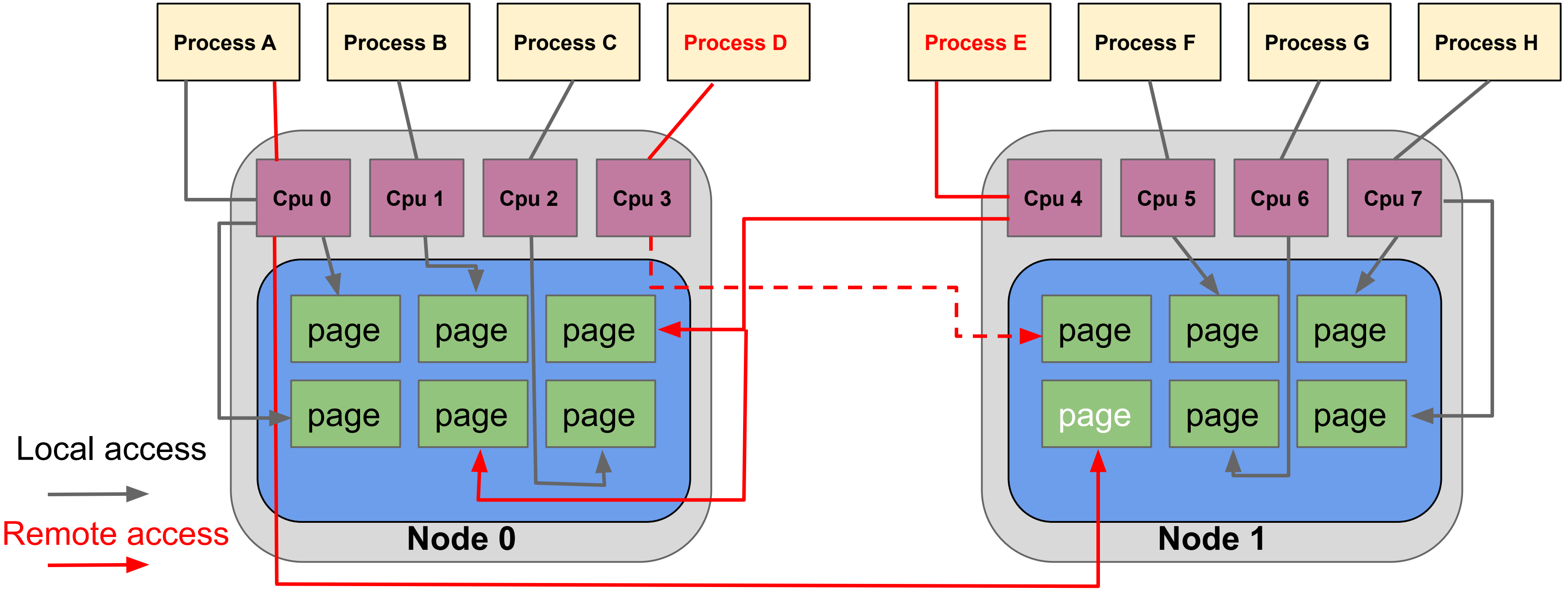


The merge list is as long as 2306920 entries.

Automatic NUMA balancing

According to the memory access latency, it would be better to migrate Process D to node 1 and Process E to node 0. The remote access page by Process A can be migrated to node 0. However, it would also need to consider the CPU loading before migrating the processes.

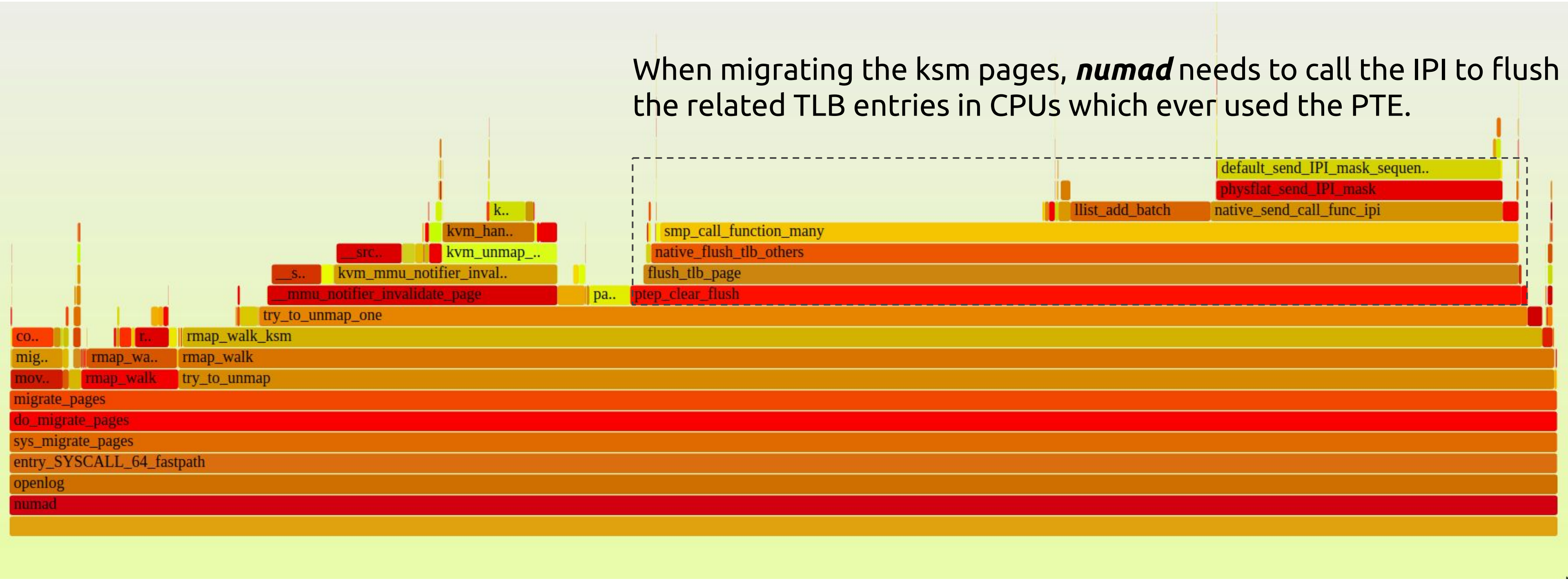
Local/Remote access



FlameGraph of the performance problem

https://kernel.ubuntu.com/~gavinguo/sf00131845/numa-131845.svg

When migrating the ksm pages, ***numad*** needs to call the IPI to flush the related TLB entries in CPUs which ever used the PTE.



Solution

Re: [PATCH 1/1] ksm: introduce ksm_max_page_sharing per page deduplication limit
https://www.spinics.net/lists/linux-mm/msg125880.html

80b18dfa53bb ksm: optimize refile of stable_node_dup at the head of the chain

8dc5ffcd5a74 ksm: swap the two output parameters of chain/chain_prune

0ba1d0f7c41c ksm: cleanup stable_node chain collapse case

b4fecc67cc56 ksm: fix use after free with merge_across_nodes = 0

2c653d0ee2ae ksm: introduce ksm_max_page_sharing per page deduplication limit