Nahanni
a shared memory interface for KVM

Cam Macdonell
University of Alberta
cam@cs.ualberta.ca
Sharing Memory

Host

Qemu Userspace

Qemu Userspace

Qemu Userspace
Sharing Memory

- Use-cases?
- data stage-in(stage-out)
- pointer-based based data structures
Sharing Memory

- caching
- data sharing
- IPC
Nahanni* Overview

- Nahanni is a mechanism for sharing host memory with VMs running on that host
  - zero-copy access to data
  - interrupt/signalling mechanism
  - guest/guest and host/guest

*also known as "ivshmemp" on the KVM/qemu lists
Qemu maps shared memory into RAM
Exposed as a PCI BAR

Qemu maps shared memory into RAM
mmap to user-level

Exposed as a PCI BAR

Qemu maps shared memory into RAM
Using Nahanni

Start the server

% ivshmem_server -m 512 -p /tmp/nahanni

Add chardev and device to the Qemu command line

-chardev socket,path=/tmp/nahanni,id=nahanni
-device ivshmem,chardev=nahanni,size=512m

OR without interrupts

-device ivshmem,shm=nahanni,size=512m
Guest Interface

- Nahanni uses the UIO driver interface in the guest
- Initialization
  - mmap registers (map region #0)
  - mmap shared memory region (map region #1)
- Synchronization primitives
  - POSIX spinlocks work in shared memory
    - cond. variables/semaphores do not
  - GCC atomic operations work
    - MCS locks
- Barriers
- Interrupts
Implementation (interrupts)

- Interrupts are triggered via writes to the interrupt register

\[
\text{regs[INTERRUPT]} = (\text{dest} \ll 16) | \text{vector};
\]

- Options
  - ioeventfd, irqfd, MSI-X
Implementation (interrupts)

- Interrupts trigger writes to the eventfds from Qemu

```c
uint64_t one = 1;
write(peers[dest].eventfds[vector], &one, 8);
```

- With KVM's ioeventfd we can avoid the Qemu process

```c
kvm_set_ioeventfd_mmio_long(peers[dest].eventfds[i], 
    reg_addr + INTERRUPT, (dest << 16) | vector, 1);
```
Possible Use Cases

- Simulations
  - NASA using shared memory for multiple-VM simulations that run on a custom OS
- Particle simulation (e.g. FLUID)
- Sharing application-level data
- Moving data in Map/Reduce applications
  - Hadoop
  - Pointer-based data in Map/Reduce
    - Phoenix
- Host/guest applications
Performance

• Data staging benchmark (host-guest)
  • Nahanni
    • ring buffer using interrupts
  • Netcat & SCP-HPN
    • over virtio-net/vhost
  • 9p
• Transport mechanism is isolated
  • warm cache on host
    • no disk I/O on read
  • file is copied to /dev/null in guest
    • no disk I/O on write
Performance

Data Stage-in

- NAHANNI
- NETCAT
- SCP-HPN
- 9P

secs vs file size (300mb, 700mb, 2g, 4g)

- NAHANNI: Low for small files, higher for larger files
- NETCAT: Very low for all file sizes
- SCP-HPN: Medium for small files, higher for larger files
- 9P: High for all file sizes, especially 4g
Conclusions and Future Work

- Nahanni is a mechanism for sharing host memory with (possibly) multiple VMs
- Synchronization primitives
  - barrier implementation
  - reliable signalling (in progress)
- Memory Allocator for Nahanni Shared Memory
  - modifying `talloc` allocator that uses memory pools for allocation (in progress)
    - from Samba
- Applications (in progress)
Acknowledgements

Avi Kivity, Anthony Liguori, Alex Graf and the Qemu/KVM development communities for all the feedback

Paul Lu, Jeremy Nickurak, Adam Wolfe Gordon, Xiaodi Ke from the Trellis group at the U of A

Thank you

Cam Macdonell
cam@cs.ualberta.ca

www.gitorious.org/nahanni