Tracing in the QEMU emulator

USER CASE STUDY

Stefan Hajnoczi
Red Hat
Tracing Summit 2014

About me

Member of KVM virtualization team at Red Hat

QEMU tracing maintainer

Happy to use any tool available...

- ftrace, perf, LTTng, SystemTap, DTrace

...to answer questions about:

- What is making everything slow?
- What does this EINVAL errno really mean?

What is QEMU?

QEMU is a hardware emulator, used by:

- KVM, Xen, Linaro

Emulates 17 CPU architectures

- x86, arm, ppc, etc

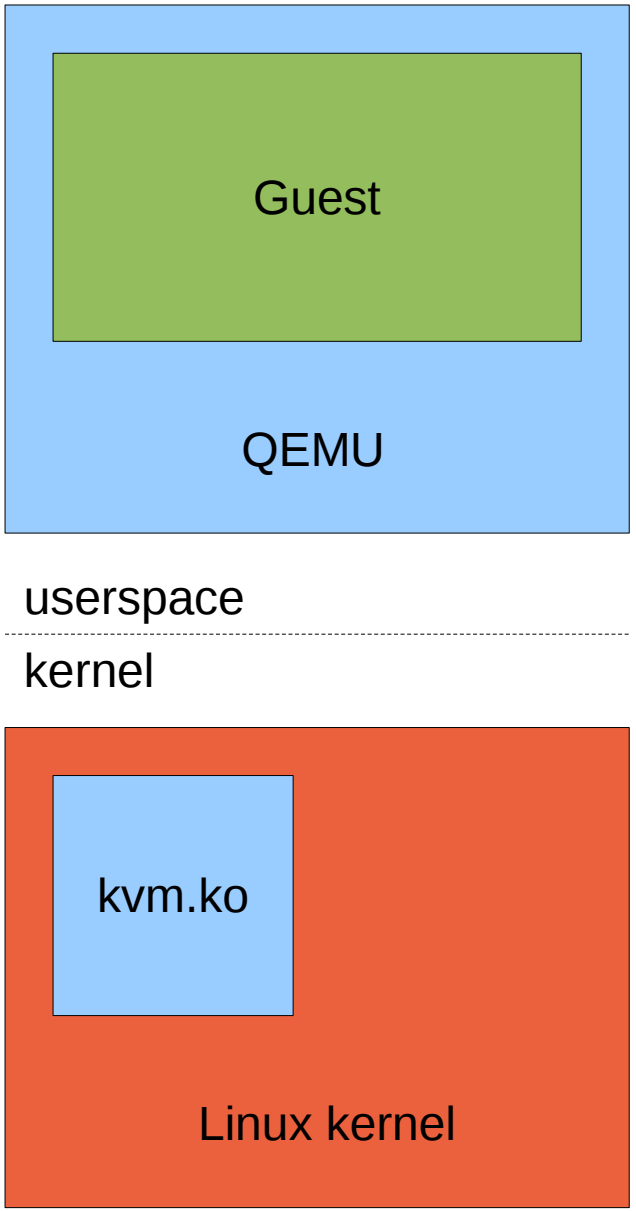
Emulates hardware devices

- net, disk, graphics, etc

Runs on 6+ OS families

- Linux, BSD, Windows, Mac, etc

QEMU architecture



QEMU is a userspace process on the host

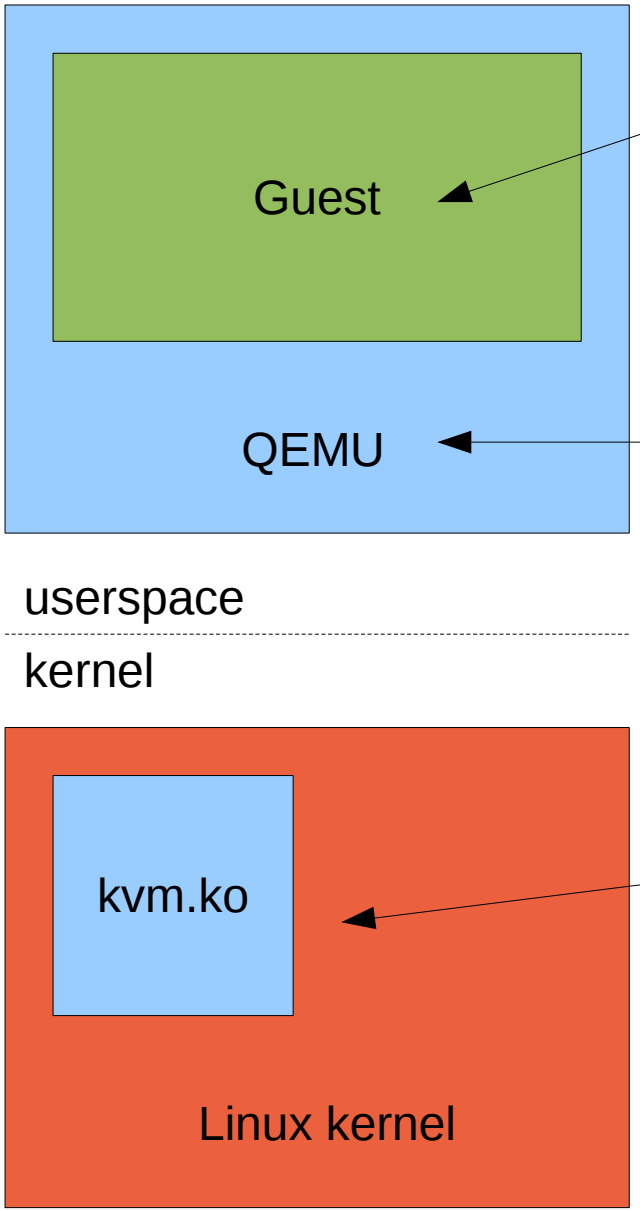
Guest runs as part of QEMU

KVM kernel module switches between host and guest mode

QEMU performs I/O on behalf of guest

Each guest vCPU is a thread on the host when using KVM

How to observe the stack



Use tools inside guest
Guest kernel using perf-kvm(1)

Static probes in QEMU
Dynamic probes using uprobes
gdb, perf, strace, etc

perf, LTTng, SystemTap, ftrace

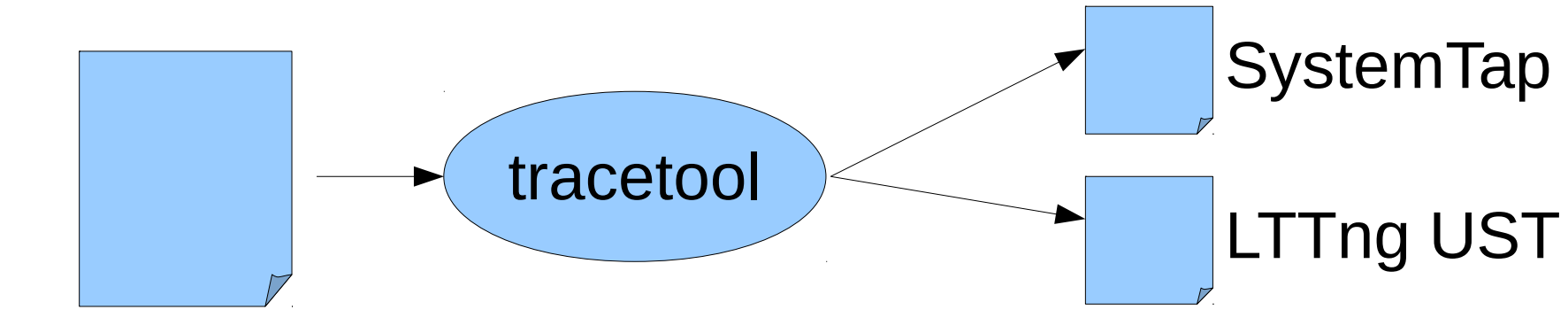
...and top, netstat, mpstat, iostat, etc

Static probes in QEMU

Problem: Support static probes across all host operating systems that QEMU compiles on.

Maybe your application faces the same challenge?

Solution: A code generator that emits tracing code for multiple tracers.



Tracers supported by tracetool

DTrace – generate static probes in .d file

SystemTap – generate tapset containing all probes

LTTng UST – generate TRACEPOINT_EVENT()

simple – QEMU's built-in tracer

stderr – trace_foo(val) -> fprintf(stderr,“foo %#x\n”,val)

nop – compile out all probes (performance paranoid)

Writing analysis scripts using “simple” tracer

```
#!/usr/bin/env python
# Print virtqueue elements not returned to the guest.

import simpletrace

class VirtqueueRequestTracker(simpletrace.Analyzer):
    def __init__(self):
        self.elems = set()

    def virtqueue_pop(self, vq, elem, in_num, out_num):
        self.elems.add(elem)

    def virtqueue_fill(self, vq, elem, length, idx):
        self.elems.remove(elem)

    def end(self):
        for elem in self.elems:
            print hex(elem)

simpletrace.run(VirtqueueRequestTracker())
```


Interested in tracetool?

Python script with tracer backend plugins

GPLv2 or Later license

Part of the QEMU source tree today, but could be spun out if there is demand

TCG tracing

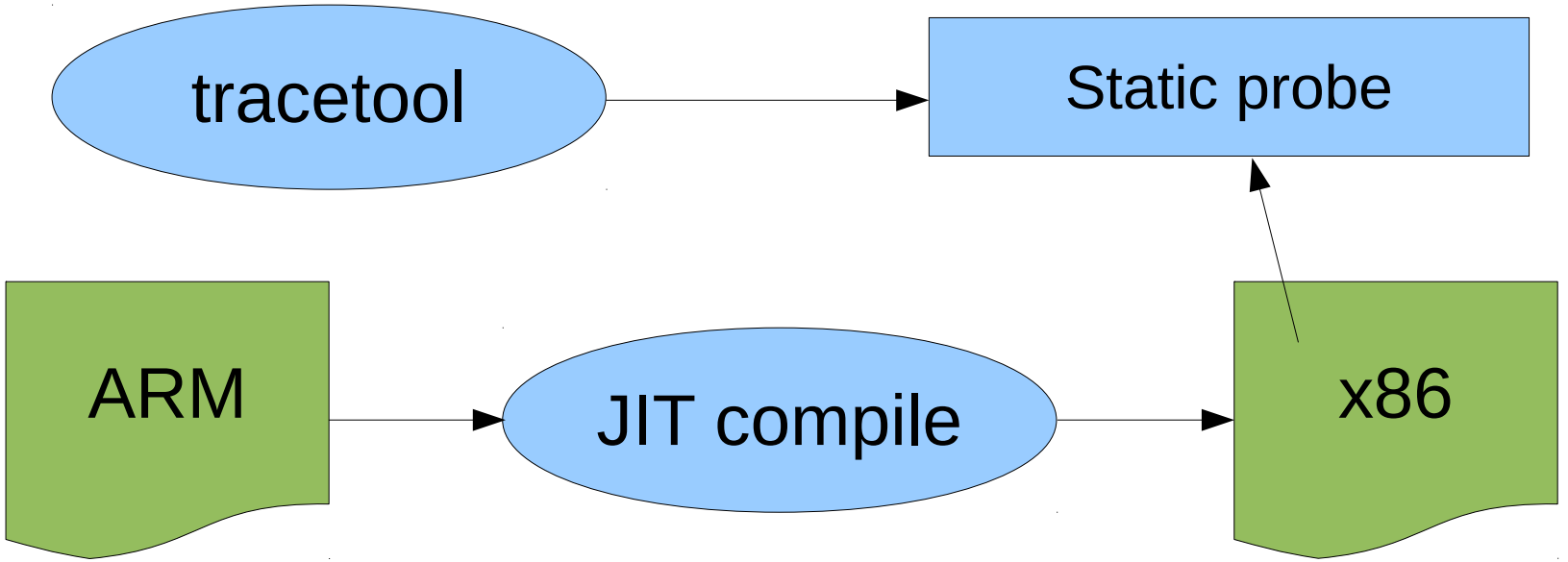
Problem: How to plant probes for Just-in-Time compiled code?

QEMU's TCG translator compiles machine code from the guest architecture to the host architecture

Solution: Generate static probes at compile-time. Let Just-in-Time compiler plant calls to these static probes.

Static probes in run-time generated code

1. Generate static probe at compile-time



2. Plant call to static probe in JIT target code

Questions?

QEMU: http://qemu-project.org/

Email: stefanha@redhat.com

Blog: http://blog.vmsplice.net/

IRC: stefanha on #qemu irc.oftc.net