Scripted Introspection with Dyninst

Josh Stone <jistone@redhat.com>
Software Engineer, Red Hat
March 26, 2012

TL;DR

Dyninst + systemtap

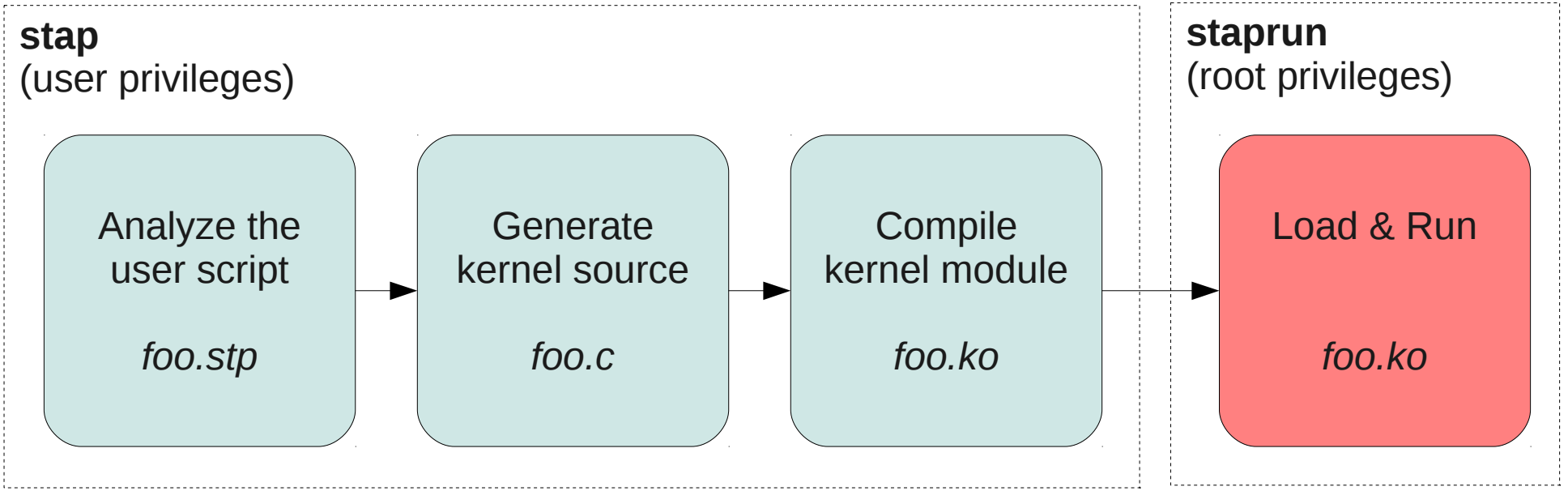
Scripted Introspection with Dyninst

- What is SystemTap?
- How can Dyninst help?
- Where are we now?
- What problems need to be solved?

SystemTap is:

- A Linux command-line tool, driven by scripts
- Instrument and run code at many events
  - Arbitrary kernel locations (kprobes)
  - Arbitrary userspace locations (uprobes)
  - Predefined locations (kernel tracepoints, SDT)
  - Hardware performance counters
  - Timers, syscalls, process lifetimes, etc.
- GPL project with developers from Red Hat, IBM, Hitachi, Oracle, and more
- http://sourceware.org/systemtap

SystemTap operation



Example 1 – top syscalls

```
global syscount

probe syscall.* {
  syscount[name] += 1
}

probe end {
  foreach (count = name in syscount- limit 10)
    printf("%6d  %s\n", count, name)
}
```


Example 2 – scheduler tracing

```
global on_times

probe scheduler.cpu_on {
  if (pid() == target())
    on_times[tid()] = local_clock_ns()
}

probe scheduler.cpu_off {
  if (tid() in on_times) {
    delta = local_clock_ns() - on_times[tid()]
    printf("%3d  %5d  %5d  %10d\n",
           cpu(), pid(), tid(), delta)
    delete on_times[tid()]
  }
}

probe begin {
  printf("%3s  %5s  %5s  %10s\n",
         "CPU", "PID", "TID", "DELTA(ns)")
}
```


SystemTap limitations

- Generating kernel modules
  - Kernel doesn't keep a fixed API
- Running kernel modules
  - Requires privilege to load
    - Root, also groups stapdev and stapusr
  - Mistakes are fatal
    - Mitigated by protected stap language

Enter Dyninst

- No special privilege required for own processes
- Dynamic
  - Run processes directly
  - Attach to live processes
- Runs in-process
  - Not even a ring transition
- Insert arbitrary code
  - e.g. Run a systemtap handler

Microbenchmark

Single-threaded – 10M NOP loops

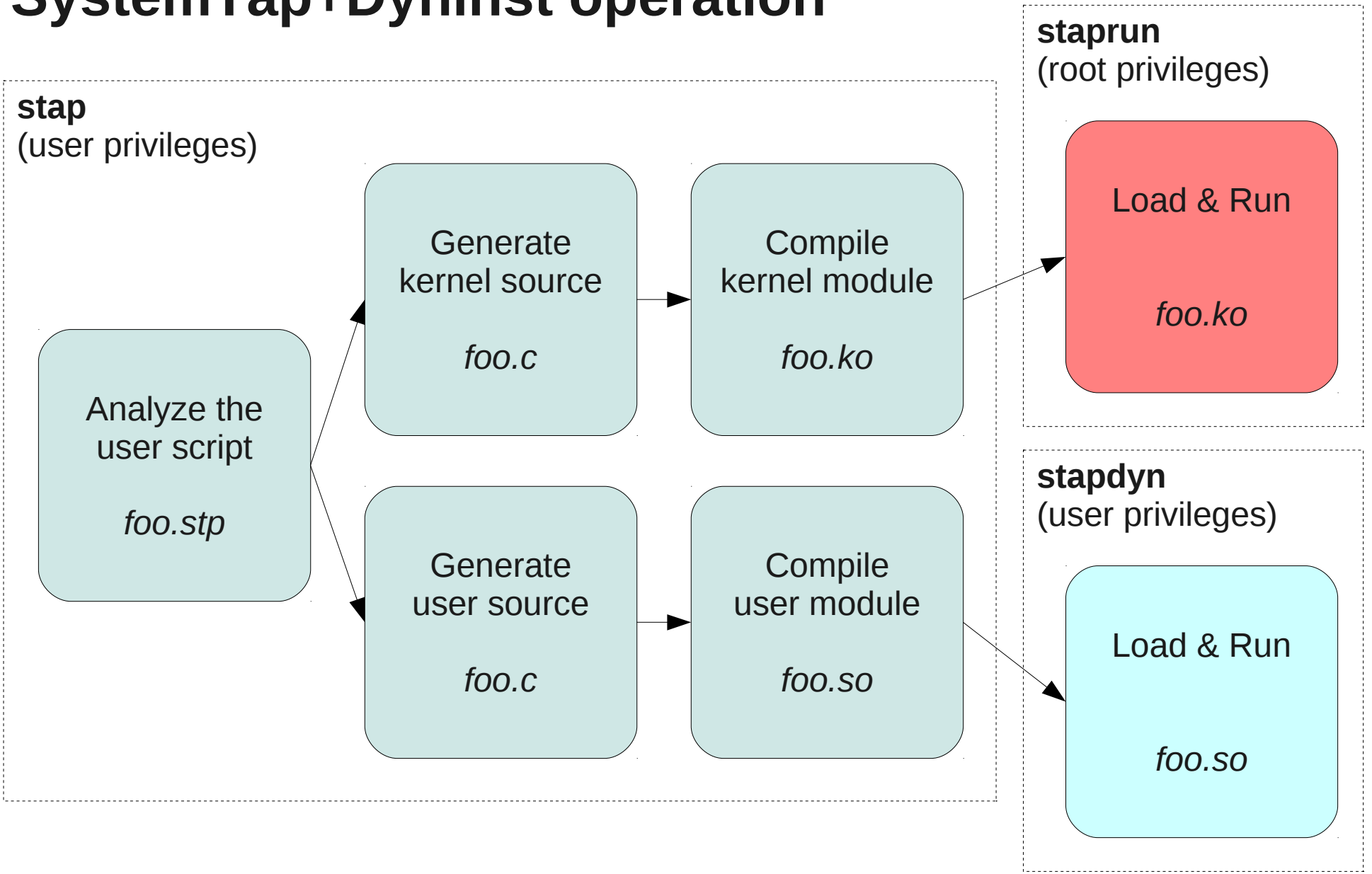
	User (ms)	System (ms)	µs/probe
base	0	0	
uprobes	870	5780	0.67
dyninst	590	0	0.06

Multi-threaded – 8 simultaneous 10M NOP loops

	User (ms)	System (ms)	µs/probe
base	40	0	
uprobes	9060	342680	4.40
dyninst	105290	8470	1.42

* perf: 60% tc_lock_lock, 20% atomic_set

SystemTap+Dyninst operation



SystemTap+Dyninst limitations

- Limited process visibility
  - Only what's accessible by ptrace
  - No practical system-wide monitoring
- Subset of systemtap events
  - YES: process.*, timers
  - NO: kernel.*, perf
- Only a single mutator for any given process
  - Thanks ptrace!

Tracing SDT

- SDT = Statically Defined Tracepoints
- dynsdt: standalone dyninst app
  - Discover SDT in target app and libraries
  - Instrument each point with a printf
  - Run and trace!

- `probe process.mark(“*”) { println($$name) }`

Integration status

- Generating userspace modules
- Connecting probe address to a handler function
- (in-progress) stapdyn loader
- TODO
  - Data transport to mutator
  - Split runtime per-thread
  - Combine state across processes

Current issues

- processCreate fails on #! interpreted scripts
- processCreate fails on ET_DYN executables (-fPIE)
- SymtabAPI fails an assertion vs. ld.so debuginfo
- createInstPointAtAddr gone after 7.0?
- Need more register access
  - Complete pt_regs, if possible
- Need a view of mmap'ed objects
  - Prototype patch to add BPatch_object

Packaging Dyninst for Fedora

- https://bugzilla.redhat.com/799089
- Known issues:
  - Avoid bundling boost and other software with Dyninst
  - Separate build and install steps
  - Honor the DESTDIR for staged installs
  - Use install rather than cp command when installing
  - Use .phony for install make rules
  - Have default for DYNINSTAPI_RT_LIB environment variable if not available

Desires for an open Dyninst community

- Public source repository
  - http://git.dyninst.org/ (not advertised)
- Public bug tracking
  - https://bugs.dyninst.org/ (not advertised)
- Public mailing list
  - dyninst-api@cs.wisc.edu (not advertised, unused?)
- Contributor agreement / guidelines
- IRC channel
- Wiki

Contact

- http://sourceware.org/systemtap
- systemtap@sourceware.org
- Josh Stone <jistone@redhat.com>