System-level critical path analysis

Progress Report Meeting
December 11th 2013

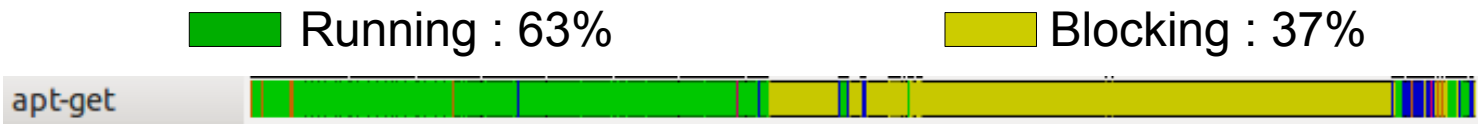
Francis Giraldeau
francis.giraldeau@polymtl.ca

Under the direction of Michel Dagenais
DORSAL Lab, École Polytechnique de Montréal

General objective

“ *Provide trace analysis tools to understand the overall performance of a distributed application.* ”

apt-get install tree

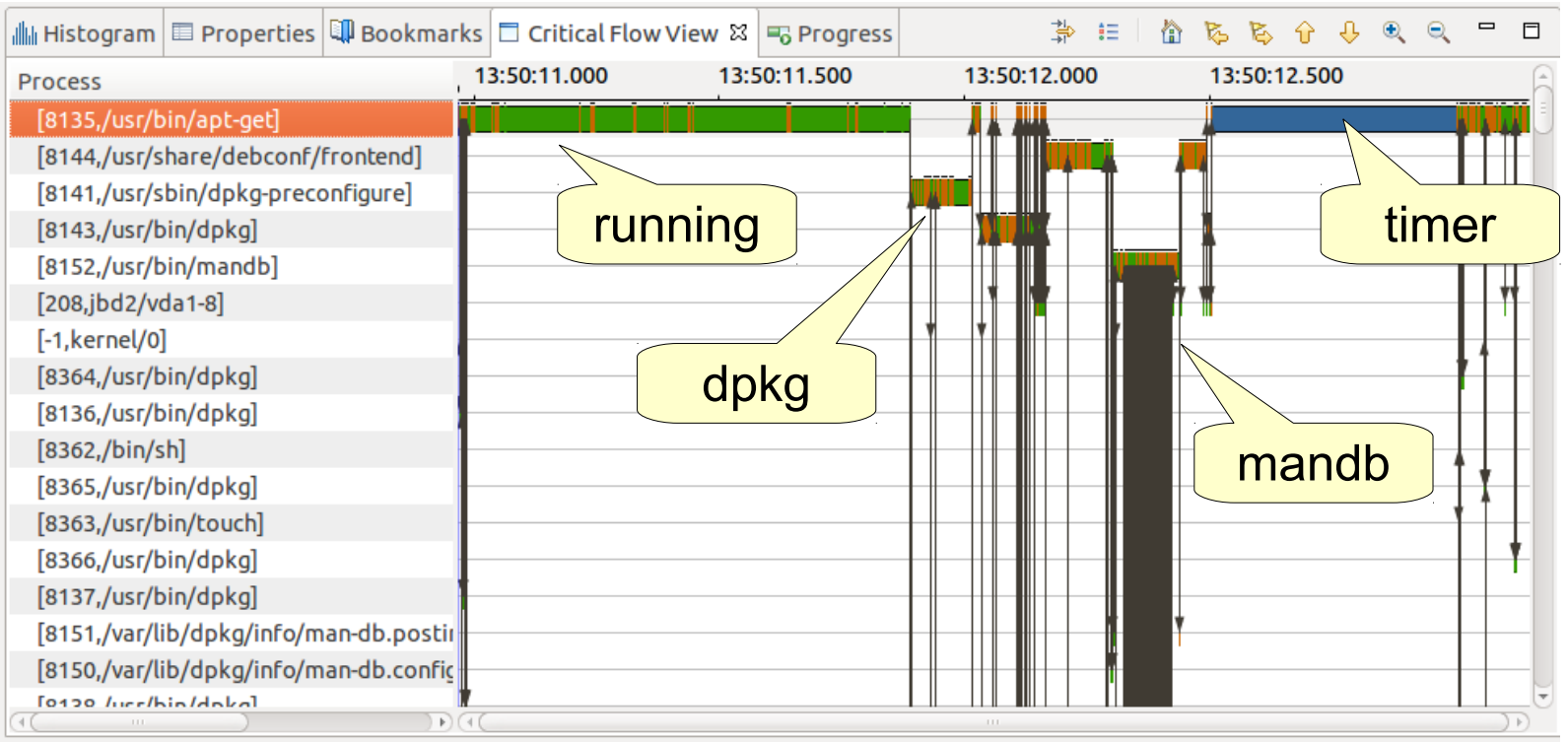


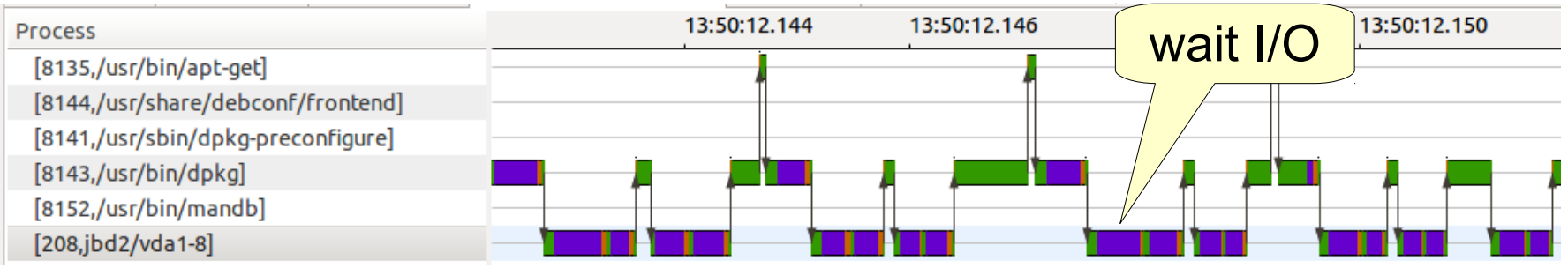
Incl.	Self	Called	Function	Location
100.00	0.00	(0)	0x0000000000001650	ld-2.17.so
99.87	0.00	1	0x000000000040a5a8	apt-get
99.87	0.00	1	(below main)	libc-2.17.so: libc-start.c
99.87	0.00	1	0x000000000040a0b0	apt-get
99.72	0.00	1	CommandLine::DispatchArg(CommandLin…	libapt-pkg.so.4.12.0
99.72	0.13	1	0x0000000000416ca0	apt-get
72.20	0.00	1	0x00000000004251e0	apt-get
63.28	0.00	1	pkgCacheFile::Open(OpProgress*, bool)	libapt-pkg.so.4.12.0
62.65	0.00	3	pkgCacheFile::BuildDepCache(OpProgress*)	libapt-pkg.so.4.12.0
62.65	0.51	1	pkgDepCache::Init(OpProgress*)	libapt-pkg.so.4.12.0
53.46	2.22	1	pkgDepCache::Update(OpProgress*)	libapt-pkg.so.4.12.0
41.23	3.66	609 598	pkgDepCache::DependencyState(pkgCac…	libapt-pkg.so.4.12.0
37.57	10.16	1 828 794	pkgDepCache::CheckDep(pkgCache::DepI…	libapt-pkg.so.4.12.0
26.49	2.24	1 130 829	debVersioningSystem::CheckDep(char con…	libapt-pkg.so.4.12.0

Callgrind output

What the app is waiting for?

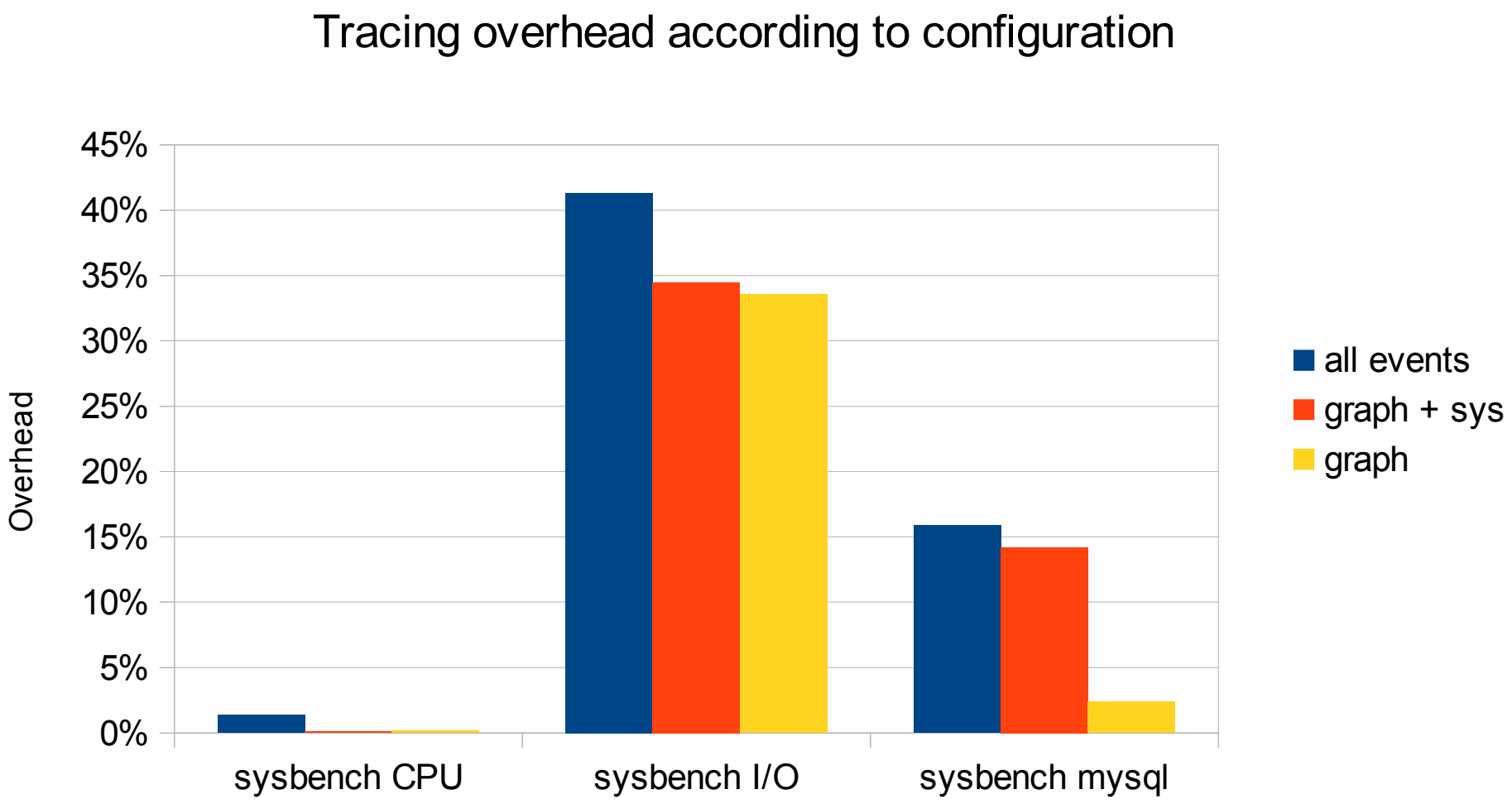
Critical Flow View

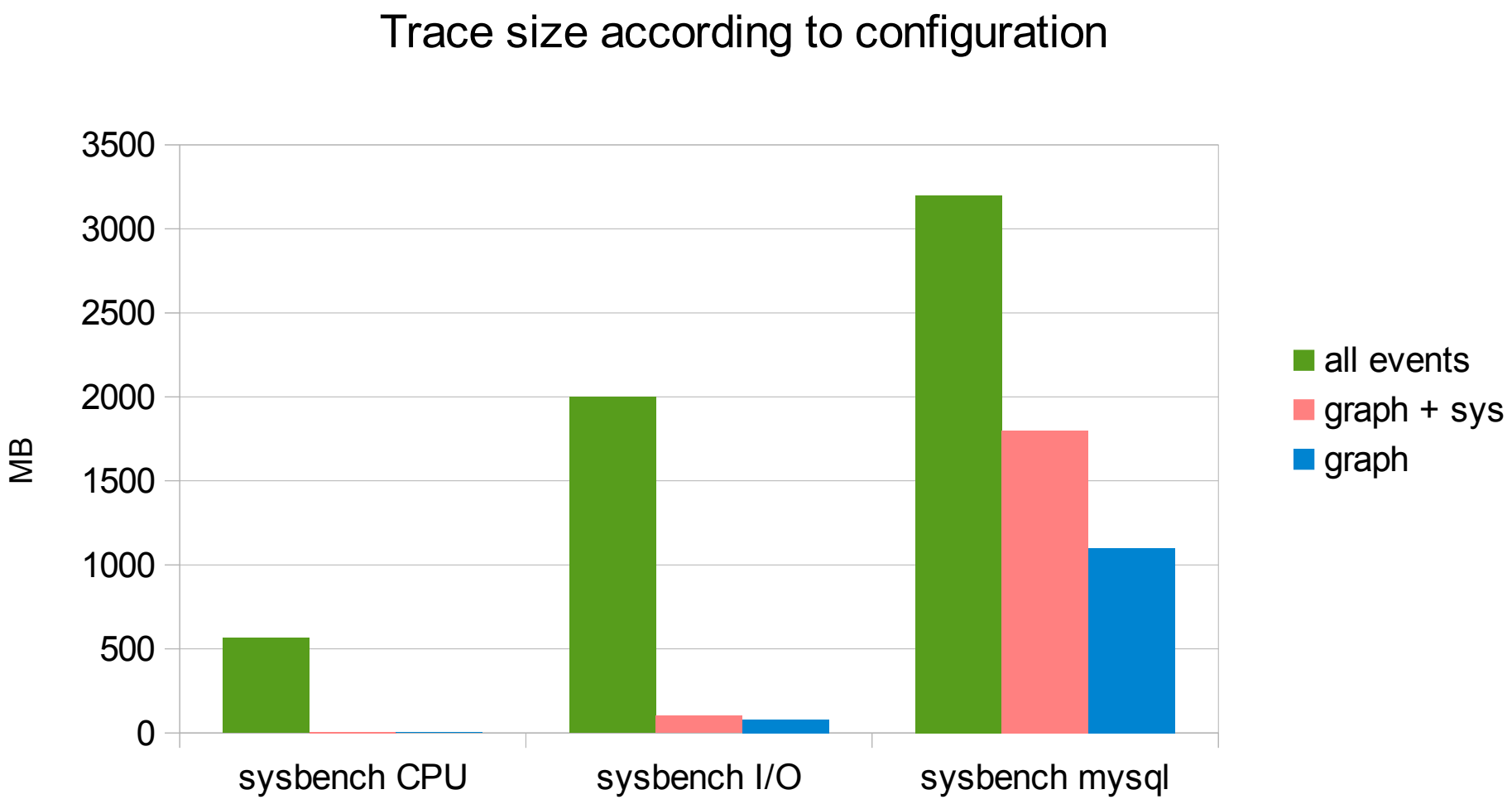


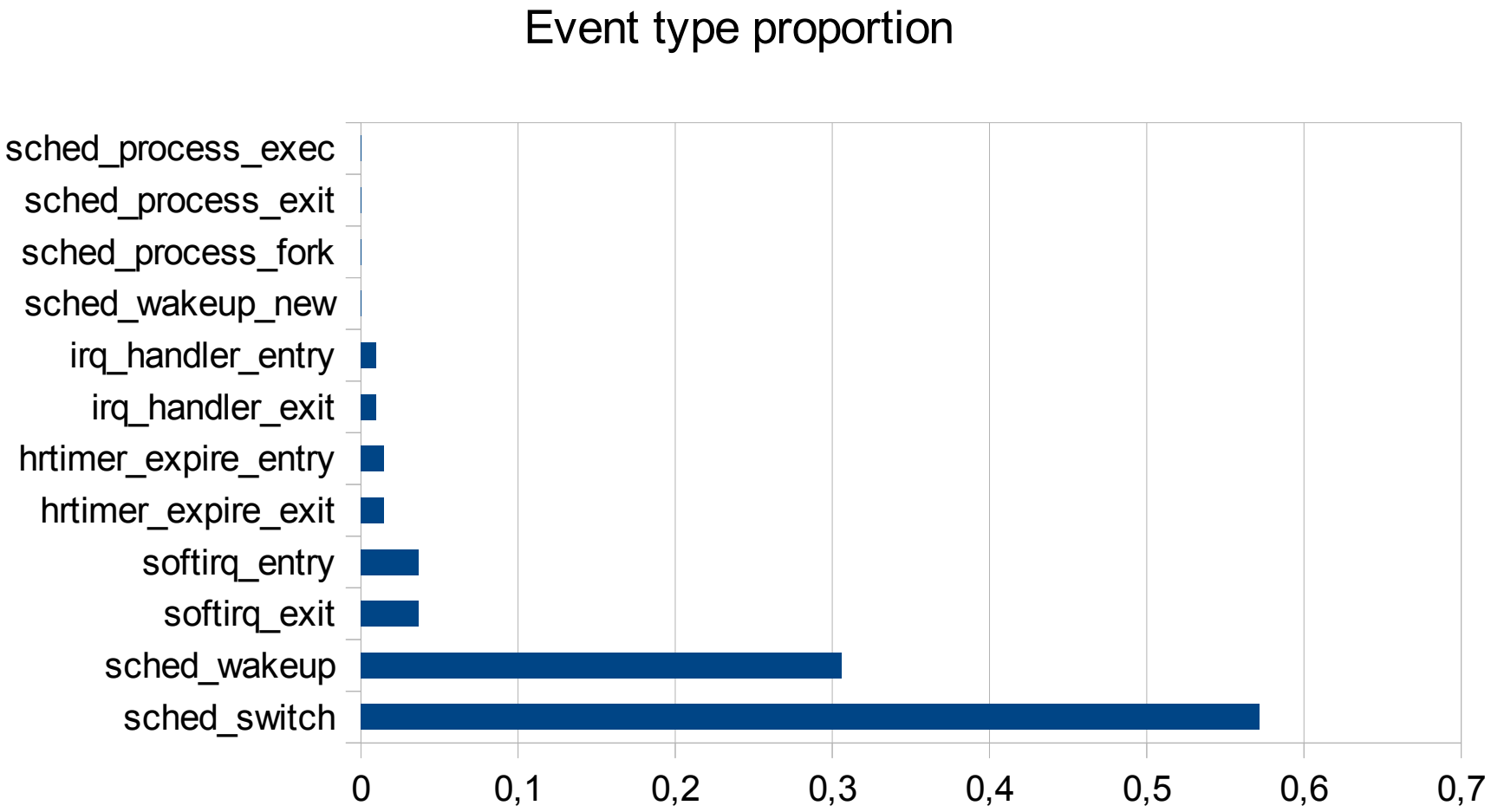


Performance impact

- Sysbench experiments: CPU, I/O, mysql
- lttng 2.3.0 - Dominus Vobiscum
- Ubuntu 13.04 – kernel 3.8.0-34-generic
- i7-3770 8GB RAM
- Single hard drive 7200 RPM (trace+load)
- No event loss achieved using ionice and renice


Tracing overhead according to configuration
Overhead
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
sysbench CPU
sysbench I/O
sysbench mysql
all events
graph + sys
graph


Trace size according to configuration
MB
3500
3000
2500
2000
1500
1000
500
0
sysbench CPU
sysbench I/O
sysbench mysql
all events
graph + sys
graph



Average event size: 36 bytes

Reduce overhead strategies

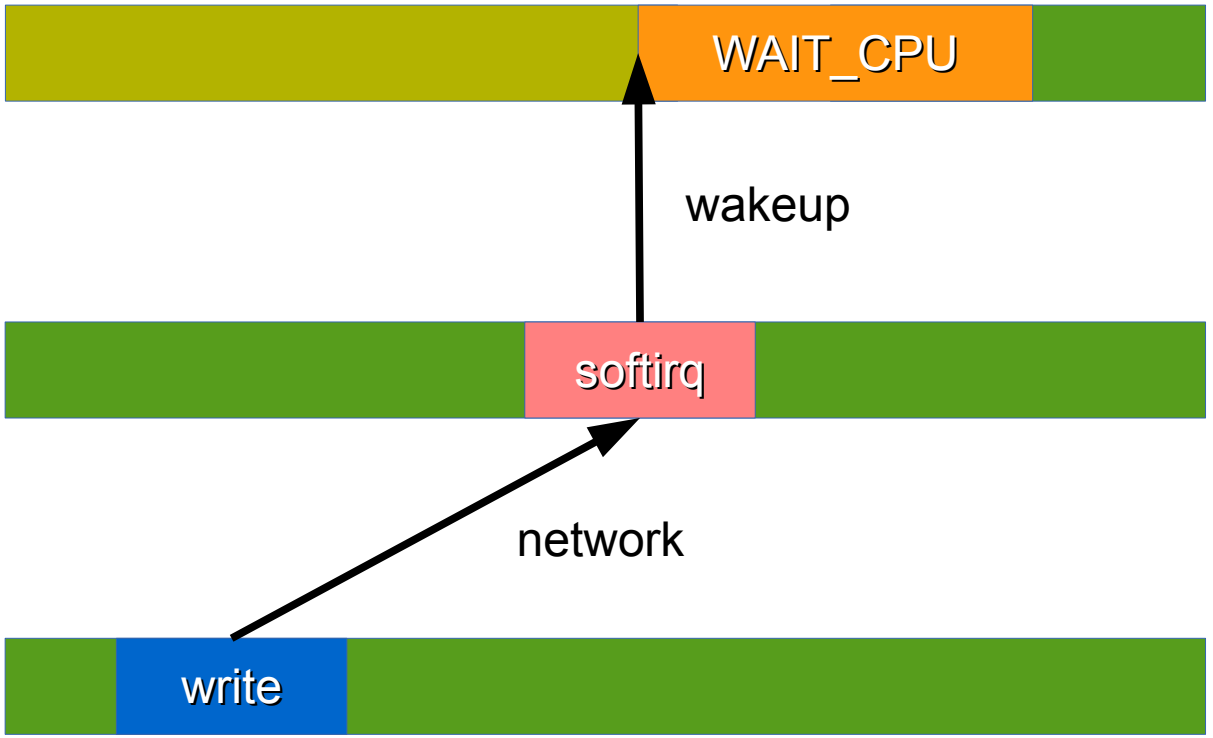
- Target required events with conditions
- Reduce event size
  - Define new TP with minimal fields
  - Record only syscall entry ID, no args
- Interrupt context instead of entry/exit

Critical path recovery of distributed apps

Recovering dependencies over TCP

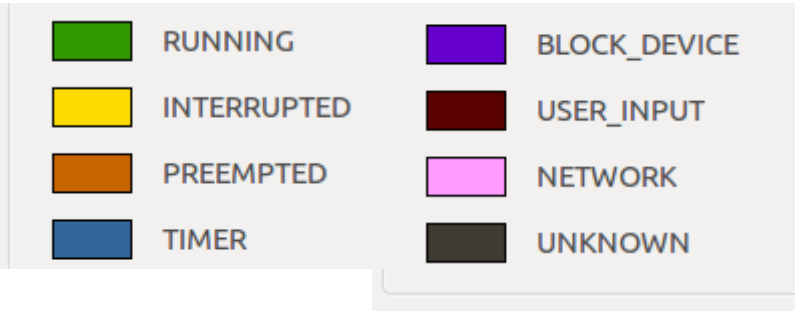
- Recording TCP headers
- Match packets
- Link related nodes in the graph
- Critical path computation: no change!

Principle of operation

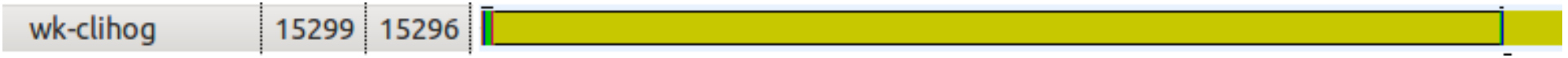


RPC Server

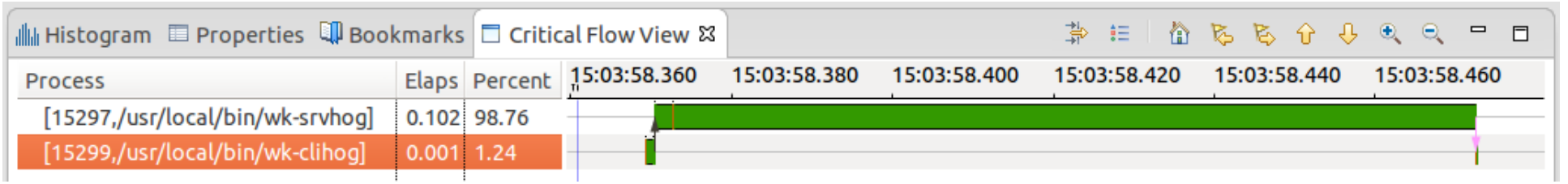
Commands: hog or sleep



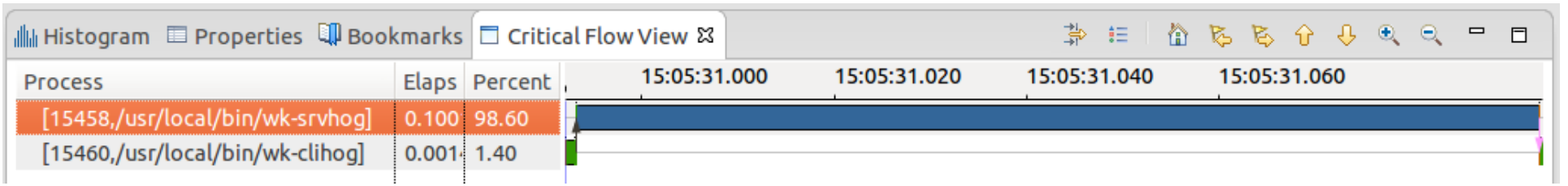
Default control flow view



Critical Flow View : request hog()

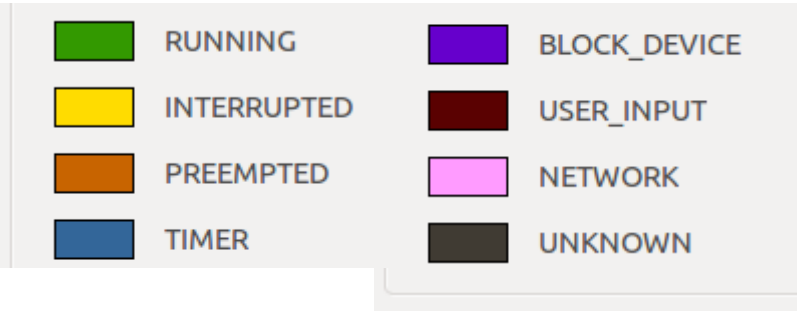


Critical Flow View : request sleep()

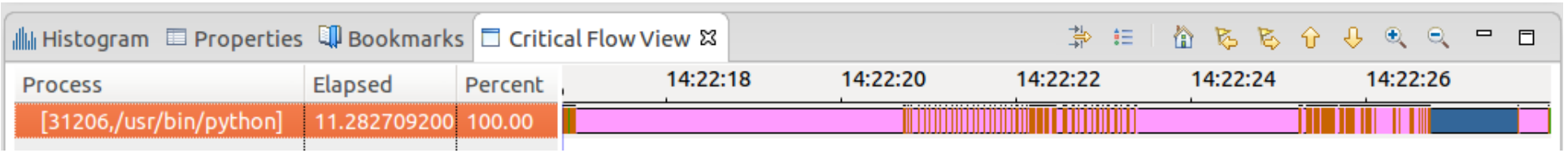


Python Django Unit Test

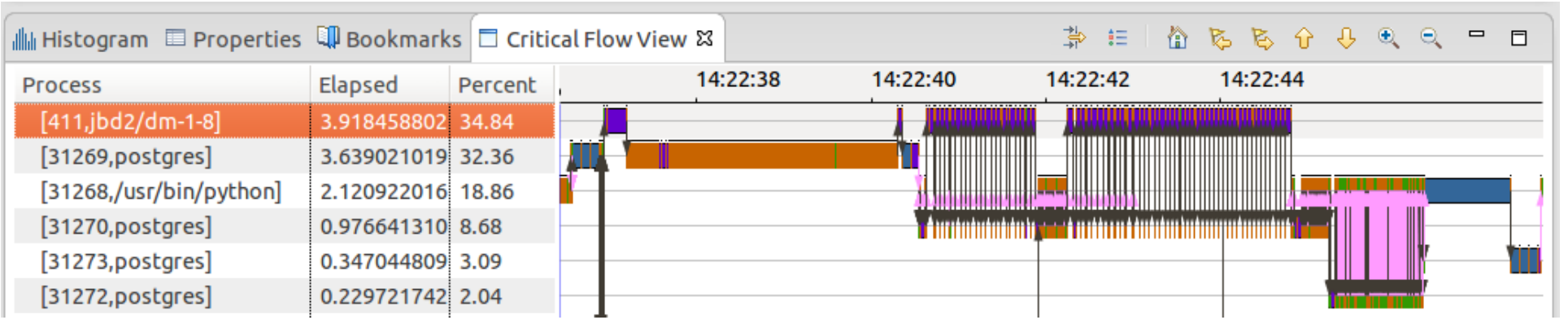
using postgresql database

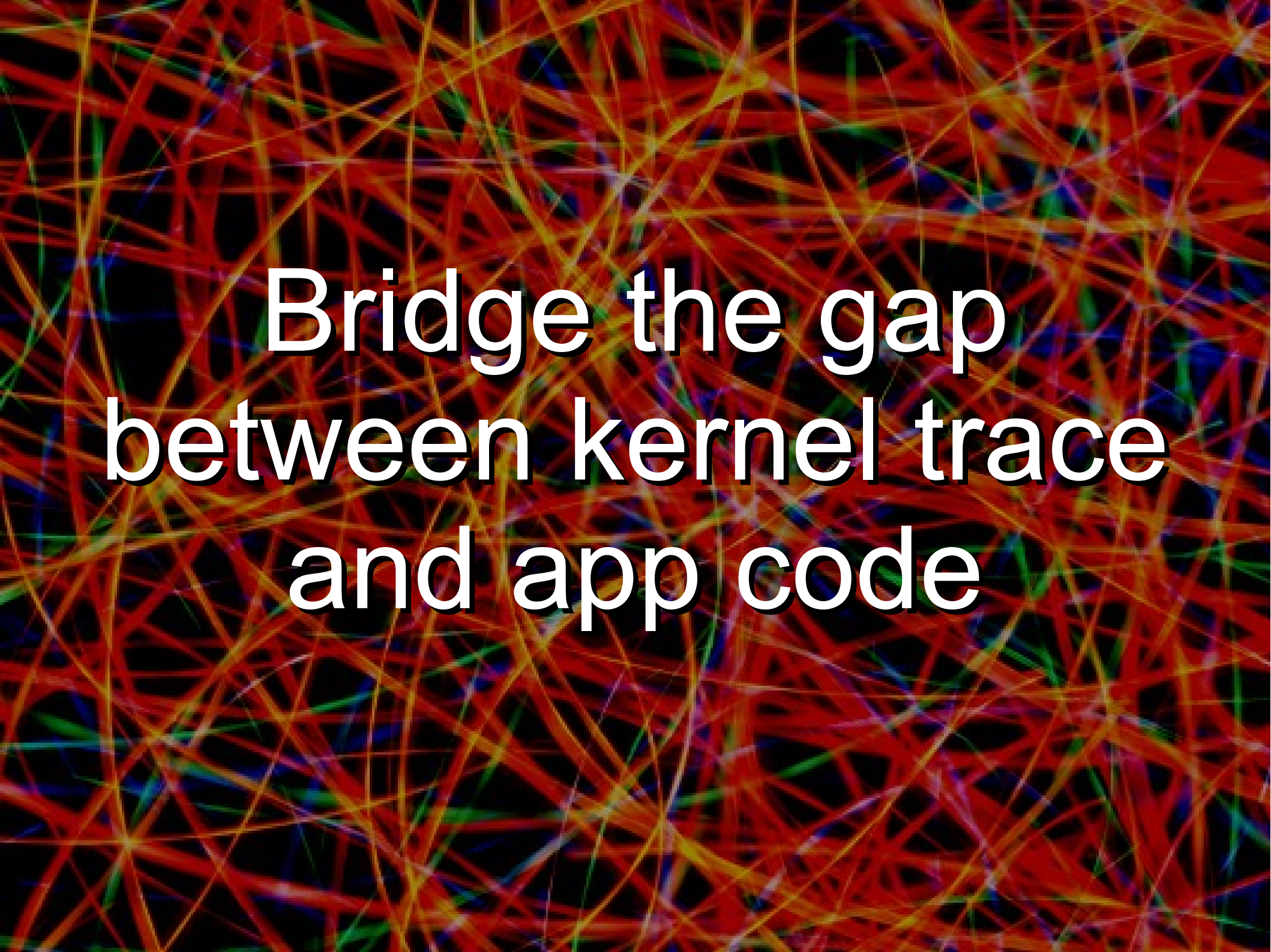


Critical Flow View with default instrumentation



Critical Flow View with TCP Packet Matching




Bridge the gap
between kernel trace
and app code

Locate system calls in the code


```
0 7ffff78fb740 __write
1 7ffff78892f3 _IO_file_write
2 7ffff78891d2 _IO_file_seek
3 7ffff788a905 _IO_do_write
4 7ffff7889b71 _IO_file_xsputn
5 7ffff7859044 _IO_vfprintf
6 7ffff78630a9 _IO_printf
7 400942       main
8 7ffff7830ea5 __libc_start_main
9 400809       _start
```

Recording ELF call stack

- Instruction pointer: mostly in libc
- Frame pointers: fast, chained list of callers
  - -fomit-frame-pointers: silly optimization on x86
- Scrape the stack: record everything that looks like a return address, yields false positive
- Unwind: recover registers state from the stack for each frame (using eh_frame)

WAMS: where are my syscalls?

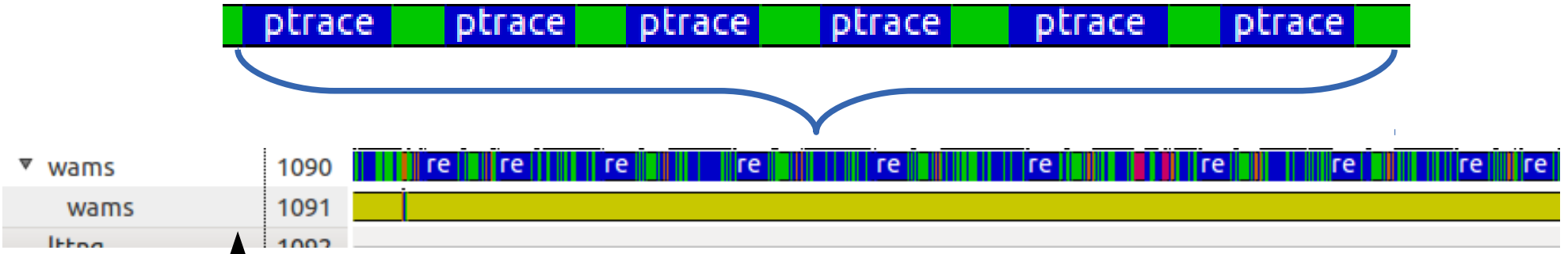
simple implementation of online unwind with ptrace

```
$ wams sleep 1
...
35
ip = 7ffff7ad28c0 nanosleep
ip = 403de7
ip = 403c7a
ip = 4016fa
ip = 7ffff7a32ea5 __libc_start_main
ip = 4017c9
```

Strace-plus (strace and unwind):
https://code.google.com/p/strace-plus/

Wams source code:
https://github.com/giraldeau/wams

Unwind over ptrace overhead



sys_mmap64() : 12us

- Nr calls to ptrace(PTRACE_PEEKDATA) : 105
- One frame processing: 274us
- 14 frames: ~4ms

Each system call adds **milliseconds** overhead

Perf callchain

Record registers + stack + mmap, offline unwind
Can be performed on sys_enter, sched_switch and sched_wakeup

```
$ sudo perf record -g --call-graph dwarf \
    -a -R -r 1 -m 4096 -f -c 1 \
    -e sched:sched_wakeup \
    -e sched:sched_switch -- $CMD
```

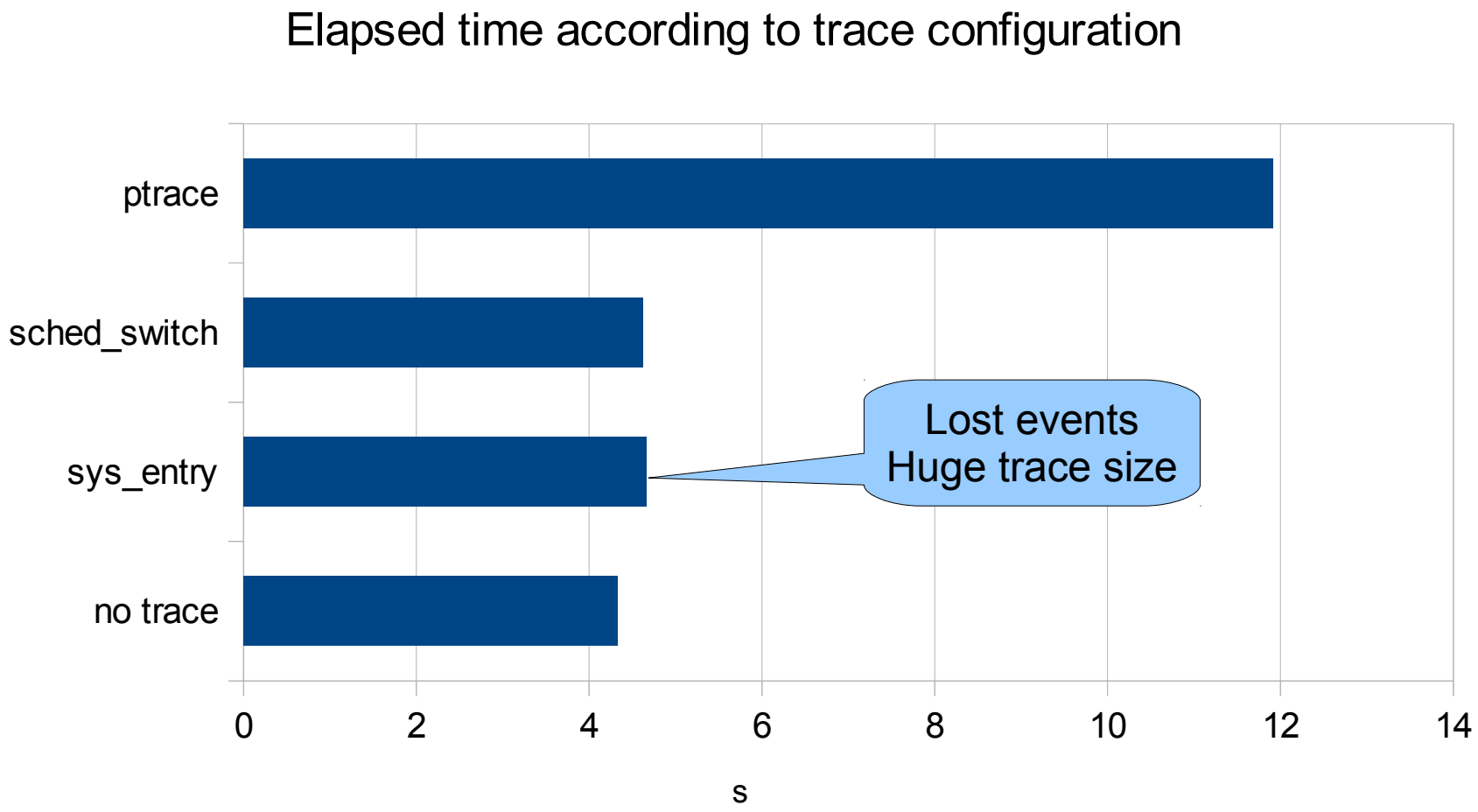
```
sample time 105358012591826 cpu 2 tid 11546 cmd mandb

ffffffff8108ca72 ttwu_do_wakeup
ffffffff8108f044 try_to_wake_up
ffffffff8108f242 default_wake_function
ffffffff81086655 __wake_up_common
ffffffff81089eb8 __wake_up
ffffffff8140e35e tty_wakeup
ffffffff8141a263 pty_write
ffffffff8141328d n_tty_write
ffffffff81410209 tty_write
ffffffff8119411c vfs_write
ffffffff81194462 sys_write
ffffffff816d37dd system_call_fastpath

7ffff72c4740     __GI___libc_write
7ffff72e8961     __printf_chk
403a1a           main
7ffff71f9ea5     __libc_start_main
403b09           _start
```

Kernel code

App code


Elapsed time according to trace configuration
ptrace
sched_switch
sys_entry
no trace
Lost events
Huge trace size
0
2
4
6
8
10
12
14
s

Future work

- Multi-host support
- On-line computation
- Reduce call chain recovery overhead
- Fix sched_wakeup IPI (linux >= 3.10)

```
.....  0: ffffffffffffff80 __mod_pnp_device_table
.....  1: ffffffff81092692 ttwu_do_wakeup
.....  2: ffffffff8109278d ttwu_do_activate.constprop.78
.....  3: ffffffff810927e7 sched_ttwu_pending
.....  4: ffffffff81092d17 scheduler_ipi
.....  5: ffffffff8103e3aa smp_reschedule_interrupt
.....  6: ffffffff816f6add reschedule_interrupt
.....  7: ffffffff8101b23f default_idle
.....  8: ffffffff8101bb06 arch_cpu_idle
.....  9: ffffffff810b54de cpu_startup_entry
..... 10: ffffffff8103eee7 start_secondary
 ... thread: swapper:0
```

Thanks to Professor Michel Dagenais and our partners EfficiOS and Ericsson.

Special thanks to Geneviève Bastien for her excellent work on Luna Dorsal.

Software:

http://secretaire.dorsal.polymtl.ca/~fgiraldeau/workload-kit/

http://secretaire.dorsal.polymtl.ca/~fgiraldeau/traceset/

https://github.com/giraldeau