

# LTTng 2.1 : Advanced Linux tracing for everyone

Yannick Brosseau

LCA 2013

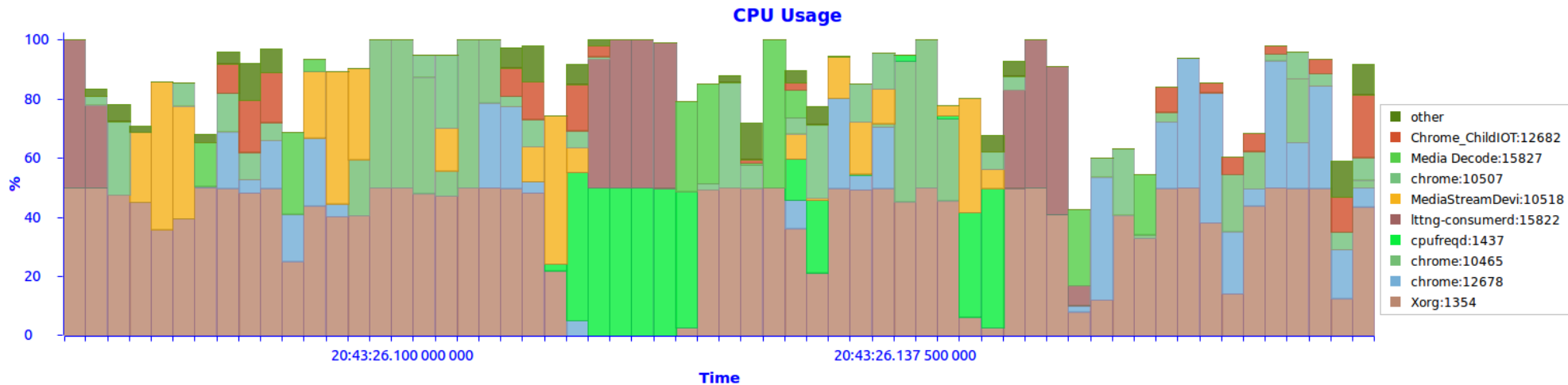


**POLYTECHNIQUE  
MONTREAL**

LE GÉNIE  
EN PREMIÈRE CLASSE



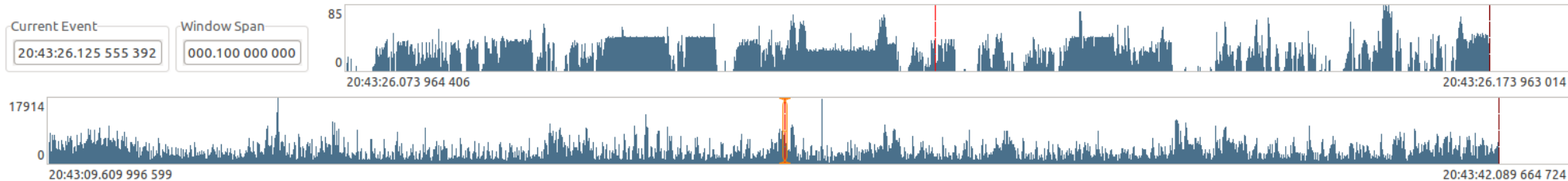
Control Flow Resources Statistics CPU Usage



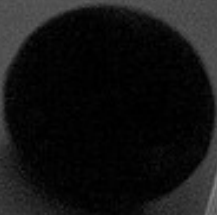
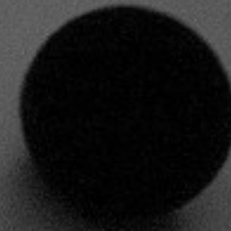
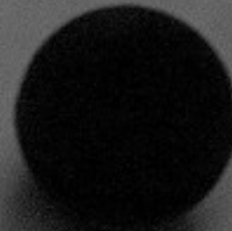
YoutubeHD

Timestamp	Channel	Event Type	Content
<srch>	<srch>	<srch>	<srch>
20:43:26.125 534 23	channel0_0	exit_syscall	ret=0
20:43:26.125 537 72	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8728
20:43:26.125 540 16	channel0_0	exit_syscall	ret=0
20:43:26.125 543 72	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8778
20:43:26.125 545 96	channel0_0	exit_syscall	ret=0
20:43:26.125 553 01	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8758
20:43:26.125 555 39	channel0_0	exit_syscall	ret=0
20:43:26.125 559 72	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8728
20:43:26.125 562 02	channel0_0	exit_syscall	ret=0
20:43:26.125 565 93	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8828
20:43:26.125 568 17	channel0_0	exit_syscall	ret=0
20:43:26.125 570 05	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8728
20:43:26.125 572 43	channel0_0	exit_syscall	ret=0
20:43:26.125 574 52	channel0_0	sys_clock_gettime	which_clock=1, tp=0xbfdb8778

Histogram Properties Bookmarks



**What is  
Tracing?**



**Definition :**

***Tracing is similar to logging: it consists in recording events that happen in a system. However, it usually records much **lower-level** events that occur much **more frequently**.***

**A way to see  
Inside...**



**A way to see  
Inside...**



**At specific  
points**

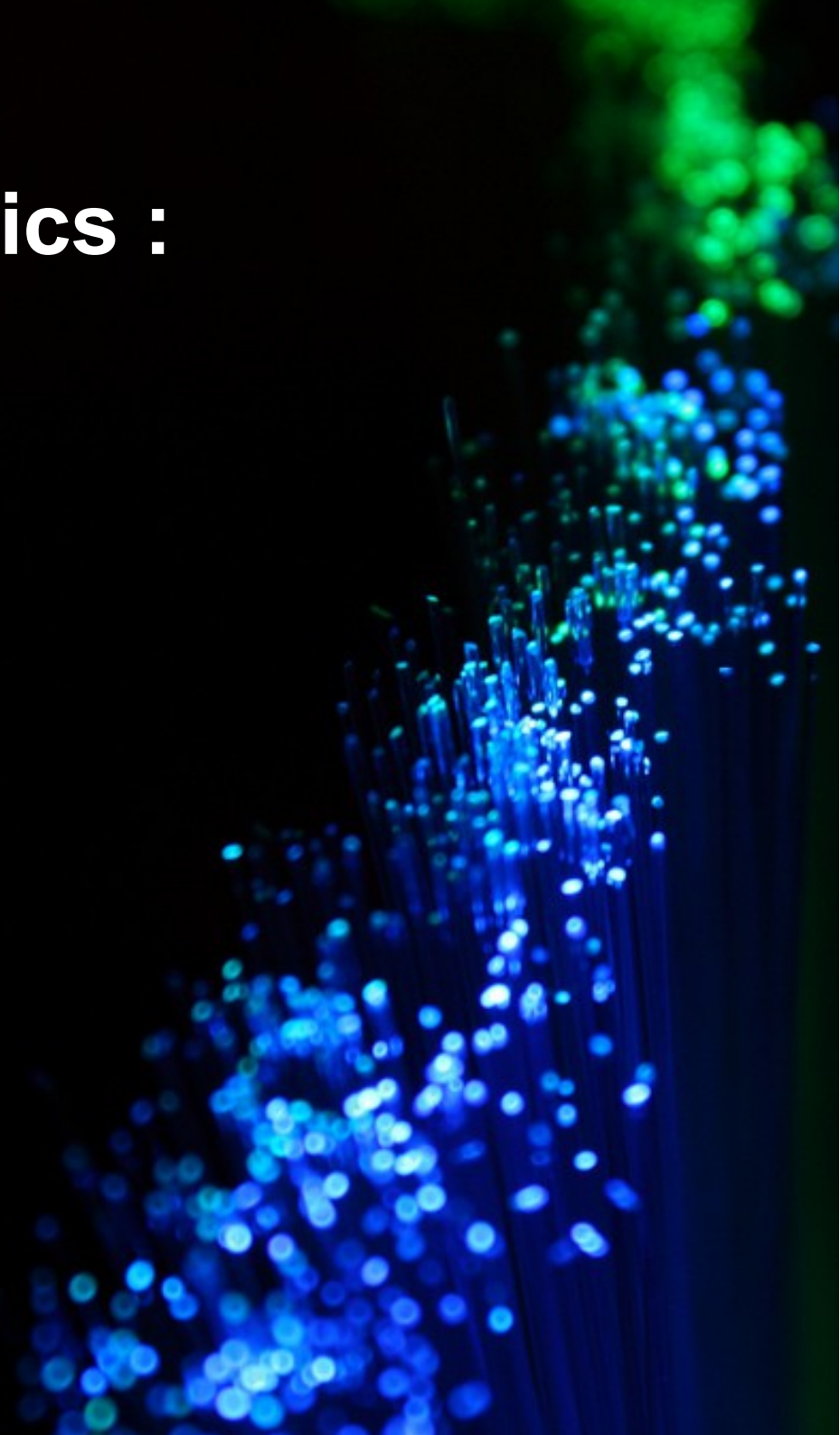


Tracepoints  
can be  
**Static**  
or  
**Dynamic**



## **Tracepoints characteristics :**

- **High speed**
- **Low latency**
- **Timestamp**
- **Payload**



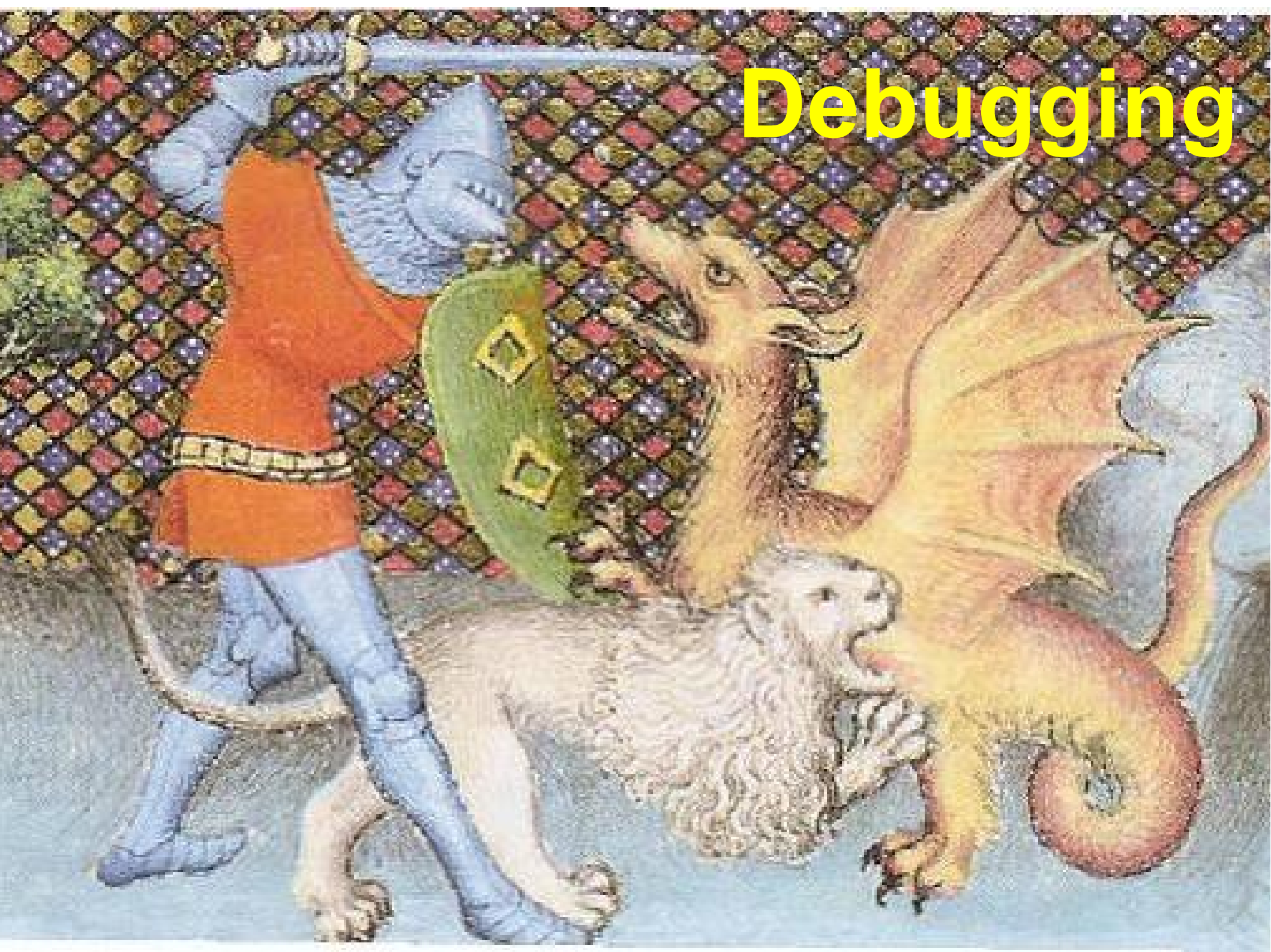


What can it be used for ?

Learning



# Debugging



# Events history





**What  
is  
LTTng  
?**



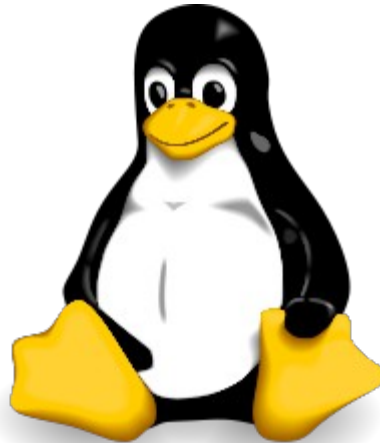
# Tracers Timeline

- 1999 : LTT
- 2005 : LTTng 0.x
- 2005 : Dtrace
- 2005 : SystemTap
- 2008 : Ftrace
- 2009 : Perf
- 2012 : LTTng 2.0
- 2013 : LTTng 2.1





# Userspace and kernel tracers





What's so special about LTTng 2.x?





# Tracing for Everyone?



A close-up, slightly angled view of a red, oval-shaped push-button. The button has a glossy finish and is set against a dark green background. The word "easy" is embossed on the button in a white, lowercase, sans-serif font. The button is mounted on a grey base.

easy

**Installation**

# No kernel patch required

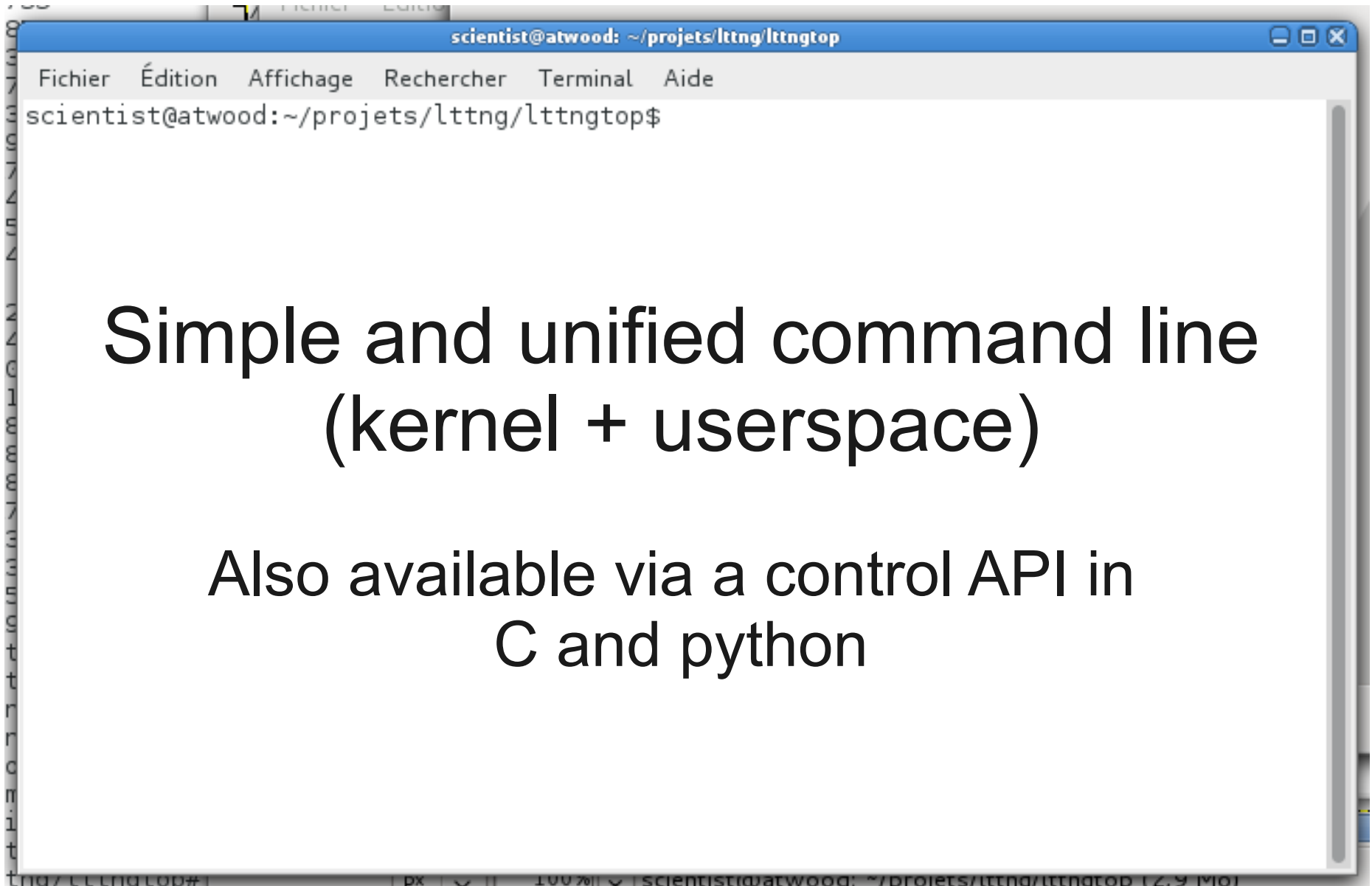
- Kernel tracing
  - Only kernel modules
  - Support kernel from 2.6.38
  - Runs on 2.6.32 with 3 small kernel patches
- Userspace tracing
  - Don't rely on the kernel

# Linux distribution

- Ubuntu
- Debian
- Fedora (without kernel modules)
- Arch
- Suse
- Red Hat (pending)

# Multi-arch and other OS

- x86 and x86-64
- ARM
- PowerPC, Sparc, Mips, Tile, Xeon Phi
  
- Android (Ongoing)
- FreeBSD (UST)
- Cygwin (UST)



Simple and unified command line  
(kernel + userspace)

Also available via a control API in  
C and python

(create, enable, start, stop, view, destroy)

# Ittng create

# Ittng enable-event -k -a

# Ittng enable-event -u -a

# Ittng start

# Ittng stop

# Ittng view

# Ittng destroy



**Tracing by non root users  
(tracing group)**

**Multiple active sessions**



# LTTng 2.0 Low-Overhead Tracing Architecture

Host



Target

**Host-Side User Interfaces**

**Babeltrace (MIT/BSD)**

- Trace converter
- Trace pretty printer
- Allow open source and proprietary plugins

libbabeltrace (MIT/BSD)

**LTTV (GPLv2)**

- Trace display and analysis
- Trace control
- Allow open-source plugins

libbabeltrace (MIT/BSD)

**Eclipse Tracing and Monitoring Framework (EPL)**

- Trace display and analysis
- Trace control
- Allow open source and proprietary plugins

**LTTng Command Line Interface (GPLv2)**

liblttngctl (LGPLv2.1)

**LTTng Session Daemon (GPLv2)**

- Control multiple tracing sessions
- Centralized tracing status management

liburcu (LGPLv2.1)

liblttngctl (LGPLv2.1)

liblttng-ust-ctl (GPLv2)

**Custom Control Software**

- Interface with proprietary cluster management infrastructures

liblttngctl (LGPLv2.1)

**C/C++ Application**

- Tracepoint\*
- Tracepoint Probes\*

liburcu (LGPLv2.1)

liblttng-ust (LGPLv2.1)

**Java/Erlang Application**

- Tracepoint\*

LTTng VM adaptor

- Tracepoint Probes\*

liburcu (LGPLv2.1)

liblttng-ust (LGPLv2.1)

**Linux kernel**

- Tracepoint\*
- Dynamic probes (kprobes)

LTTng modules (GPLv2/LGPLv2.1)

- Tracepoint Probes\*

Posix shared memory and pipe

Posix shared memory and pipe

Memory-mapped buffers or splice, poll, ioctl

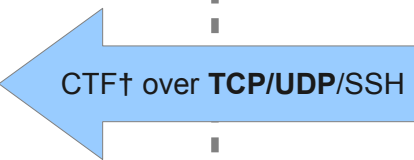
**LTTng Consumer Daemon (GPLv2)**

- Zero-copy data transport or aggregator
- Export raw trace data, statistics and summary data
- Snapshots from in-memory flight recorder mode
- Store all trace data, discard on overrun

liburcu (LGPLv2.1)

liblttng-ust-ctl (GPLv2)

liblttng-consumer (GPLv2)



† **Common Trace Format (CTF)**

- Compact binary format,
- Self-described,
- Handles HW&SW tracing,
- TCP and UDP network streaming,
- Flexible data layouts for expressiveness and highest throughput,
- Layout allows fast seek and processing of very large traces (> 10GB).

\* **Tracepoint and Probes Characteristics**

- Low overhead, no trap, no system call,
- Re-entrant: Signal, thread and NMI-safe,
- Wait-free read-copy update,
- Can be used in real-time systems,
- Use GCC asm goto and Linux kernel static jumps,
- Cycle-level time-stamp,
- Runtime activation of statically and dynamically inserted instrumentation,
- Non-blocking atomic operations,
- Allow tracing of proprietary applications and proprietary control software (LGPLv2.1 license).

- Instrumentation
- Control
- Trace Data
- Libraries

# Multiples data sources...

- Kernel :
  - Static Tracepoints
  - Syscalls
  - Kprobes
  - Function tracer
  - Context (including perf counters)
- Userspace
  - Static Tracepoints
    - With filtering
  - Context



... That produce CTF data

- Common Trace Format, defined by the MultiCore Association

# Kernel tracepoint anatomy

```
TRACE_EVENT(sched_process_fork,  
  
    TP_PROTO(struct task_struct *parent, struct task_struct *child),  
  
    TP_ARGS(parent, child),  
  
    TP_STRUCT__entry(  
        __array(    char, parent_comm,  TASK_COMM_LEN  )  
        __field(   pid_t, parent_pid      )  
        __array(    char, child_comm,   TASK_COMM_LEN  )  
        __field(   pid_t, child_pid      )  
    ),  
  
    TP_fast_assign(  
        memcpy(__entry->parent_comm, parent->comm, TASK_COMM_LEN);  
        __entry->parent_pid   = parent->pid;  
        memcpy(__entry->child_comm, child->comm, TASK_COMM_LEN);  
        __entry->child_pid    = child->pid;  
    ),  
  
    TP_printk("comm=%s pid=%d child_comm=%s child_pid=%d",  
        __entry->parent_comm, __entry->parent_pid,  
        __entry->child_comm, __entry->child_pid)  
);
```

# Simple userspace tracepoint generation

```
TRACEPOINT_EVENT(  
    sample_tracepoint,  Component name  
    message,  Tracepoint name  
    TP_ARGS(char *, text),  
    TP_FIELDS(  
        ctf_string(message, text)  
    )  
)
```

Generated by Ittng-gen-tp :

In C :

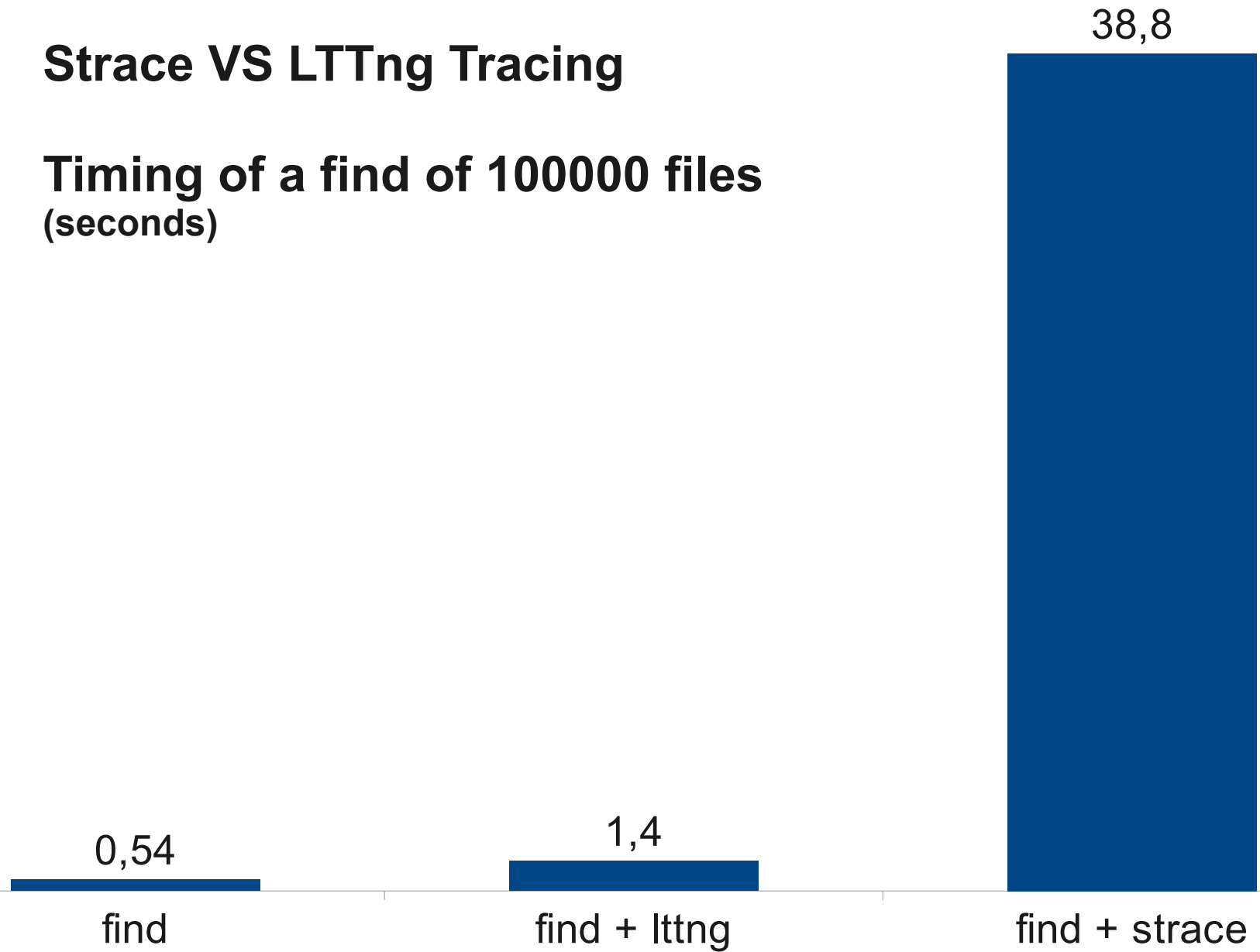
```
#include "sample_tracepoint.h"  
tracepoint(sample_tracepoint, message,  
            "Hello World\n");
```

# FAST



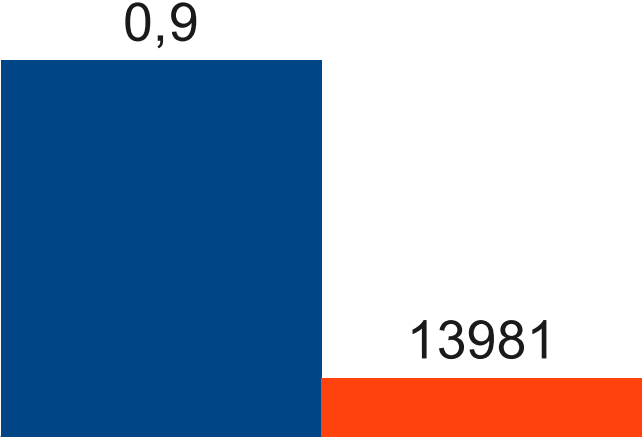
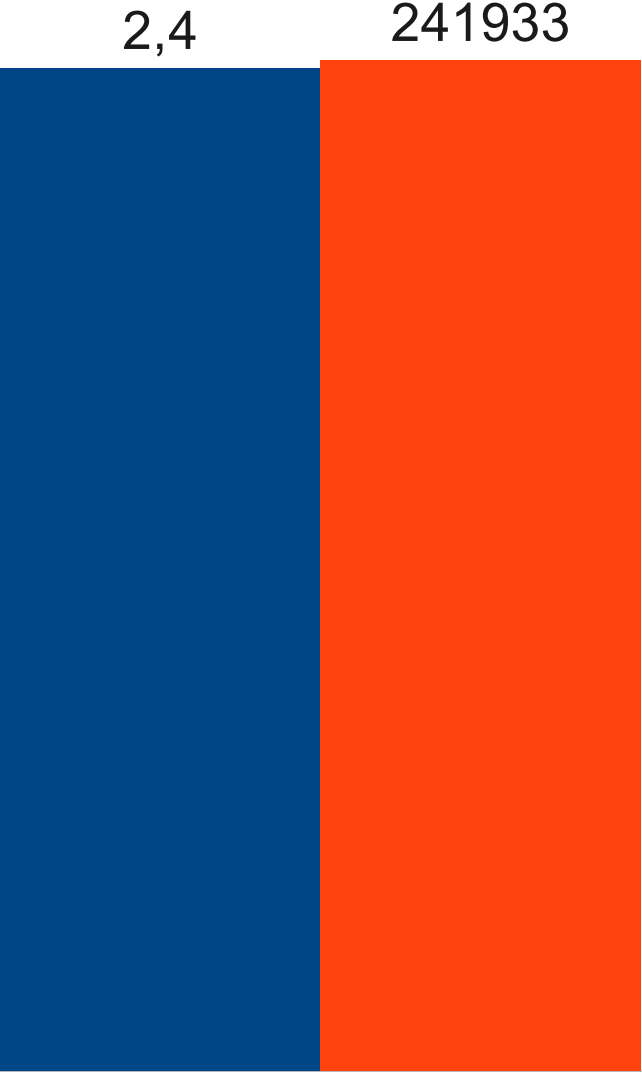
## Strace VS LTTng Tracing

Timing of a find of 100000 files  
(seconds)





# Top vs LTTngTop



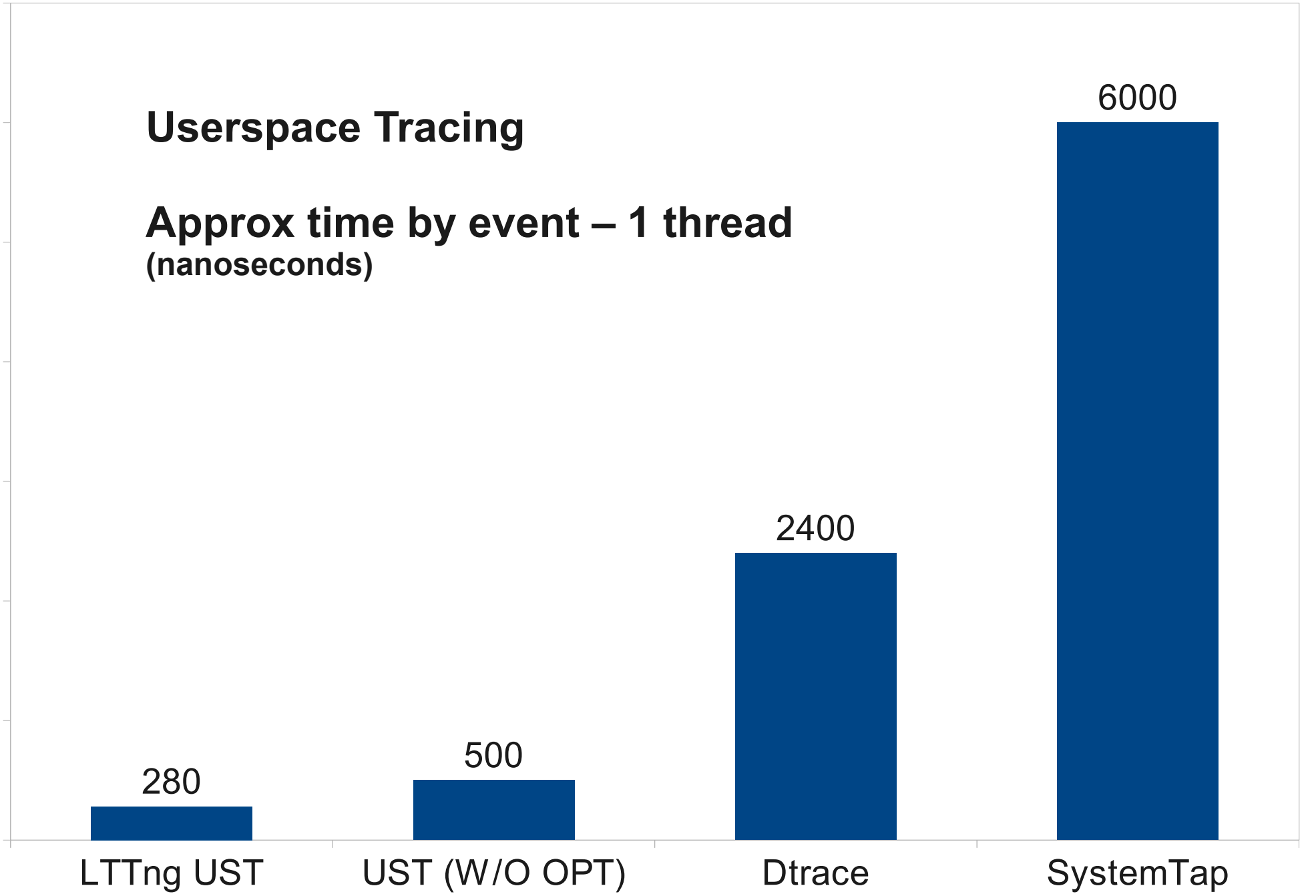
top

ltnngtop

■ Total CPU time  
■ # syscalls

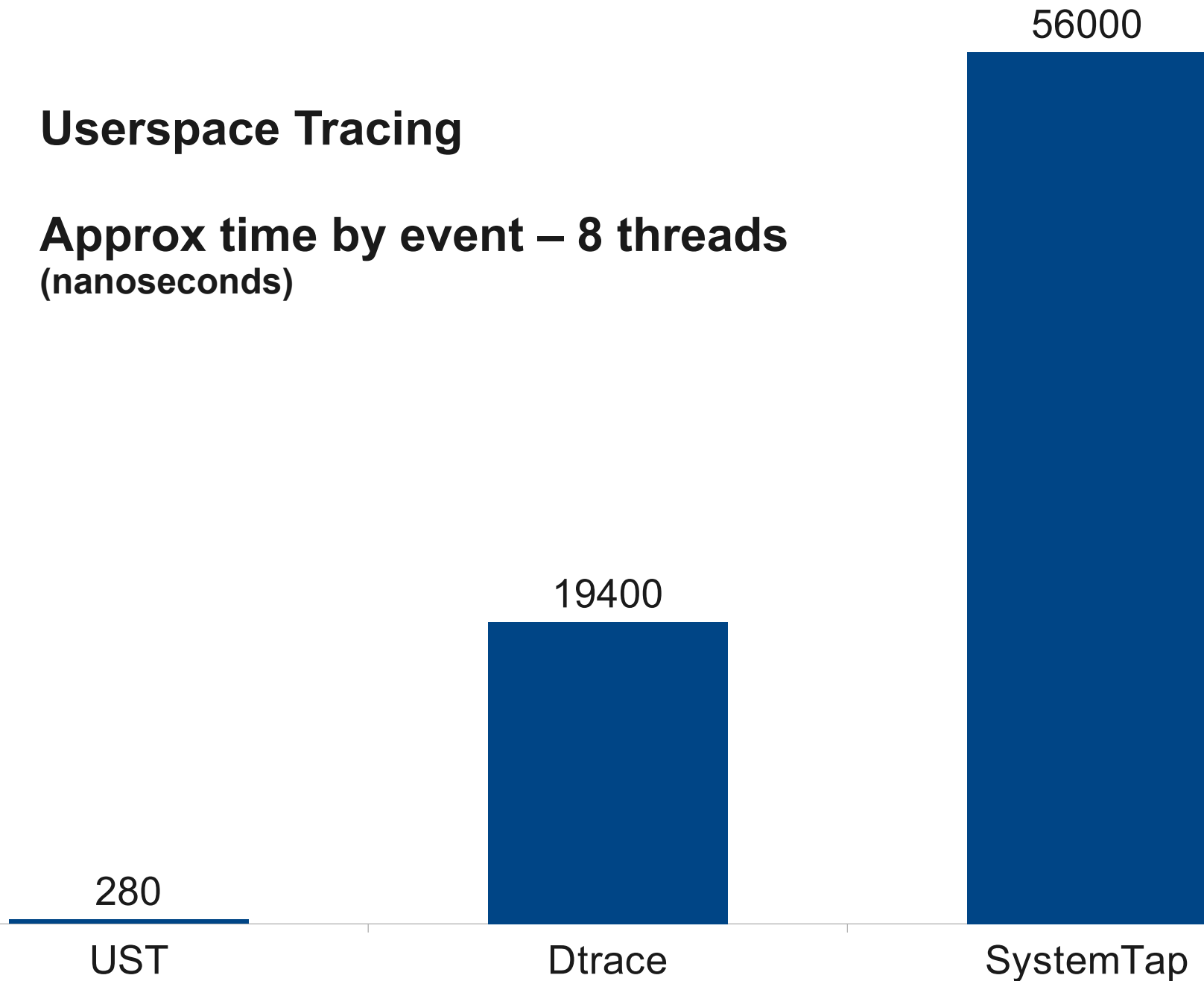
# Userspace Tracing

Approx time by event – 1 thread  
(nanoseconds)



## Userspace Tracing

Approx time by event – 8 threads  
(nanoseconds)





Looking  
at  
the  
trace  
data

# Babeltrace

```
[13:58:29.128909723] (+0.000002475) sys_read: { 0 }, { "firefox-bin", 3363 }, { fd = 5, buf =
count = 16 }
[13:58:29.128911513] (+0.000001790) exit_syscall: { 0 }, { "firefox-bin", 3363 }, { ret = -11
[13:58:29.128919672] (+0.000008159) sys_write: { 0 }, { "firefox-bin", 3363 }, { fd = 5, buf
, count = 8 }
[13:58:29.128921404] (+0.000001732) exit_syscall: { 0 }, { "firefox-bin", 3363 }, { ret = 8 }
[13:58:29.128922884] (+0.000001480) sys_read: { 0 }, { "firefox-bin", 3363 }, { fd = 19, buf
, count = 1 }
[13:58:29.128925765] (+0.000002881) exit_syscall: { 0 }, { "firefox-bin", 3363 }, { ret = 1 }
[13:58:29.128928120] (+0.000002355) sys_write: { 0 }, { "firefox-bin", 3363 }, { fd = 5, buf
, count = 8 }
[13:58:29.128929552] (+0.000001432) exit_syscall: { 0 }, { "firefox-bin", 3363 }, { ret = 8 }
[13:58:29.129020005] (+0.000090453) exit_syscall: { 0 }, { "acpid", 1536 }, { ret = 1 }
[13:58:29.129025587] (+0.000005582) sys_rt_sigprocmask: { 0 }, { "acpid", 1536 }, { how = 0,
oset = 0x0, sigsetsize = 8 }
[13:58:29.129027993] (+0.000002406) exit_syscall: { 0 }, { "acpid", 1536 }, { ret = 0 }
[13:58:29.129030188] (+0.000002195) sys_poll: { 0 }, { "acpid", 1536 }, { ufds = 0x7FFF2A055D
meout_msecs = 0 }
[13:58:29.129032570] (+0.000002382) exit_syscall: { 0 }, { "acpid", 1536 }, { ret = 0 }
[13:58:29.129033929] (+0.000001359) sys_rt_sigprocmask: { 0 }, { "acpid", 1536 }, { how = 1,
oset = 0x0, sigsetsize = 8 }
[13:58:29.129035144] (+0.000001215) exit_syscall: { 0 }, { "acpid", 1536 }, { ret = 0 }
[13:58:29.129037520] (+0.000002376) sys_read: { 0 }, { "acpid", 1536 }, { fd = 4, buf = 0x7FF
= 24 }
.. -
```

Statistics for interval [1330053201794942051, 1330053202795131720[

CPU	4	(max/cpu : 25.00%)	
Processes	N/A	(0, 0)	
Threads	N/A	(0, 0)	
Files	N/A	(0, 0)	N/A kbytes/sec
Network	N/A	(0, 0)	N/A Mbytes/sec

CPU Top

CPU(%)	TGID	PID	NAME
10.00	23844	23844	gnome-shell
5.50	20627	20627	firefox-bin
0.93	23653	23653	Xorg
0.29	4788	4788	epiphany-browser
0.05	11223	11223	kworker/2:2
0.05	11173	11173	kworker/0:0
0.05	11222	11222	kworker/1:1
0.05	10843	10843	kworker/3:1
0.04	14809	14809	hald
0.04	24103	24103	xchat
0.02	31261	31261	synergyc
0.02	20247	20247	emacs
0.02	6251	6251	emacs
0.02	2403	2403	soffice.bin
0.01	25701	25701	emacs
0.01	2719	2719	nmbd
0.01	13085	13085	icedove-bin
0.01	1534	1534	dbus-daemon
0.00	11193	11193	kworker/u:1
0.00	10985	10985	kworker/u:2
0.00	577	577	ips-monitor
0.00	9750	9750	ksoftirqd/3
0.00	17301	17301	kworker/1:2
0.00	23813	23813	gnome-settings-

# ltnngtop

Status

Starting display  
Pause

Linux Trace Toolkit Viewer

File View Tools Plugins Help

Traceset

**Resource**

- Blockdev (22,0)
- Blockdev (3,0)
- CPU0
- IRQ 1 [i8042]
- IRQ 14 [ide0]
- IRQ 15 [ide1]**
- IRQ 18 [uhci\_hcd:usb2]
- IRQ 19 [uhci\_hcd:usb1]

**Process**

Process	Brand	PID	TGID	PPID	CPU
su	UNBRANDED	4182	4182	4151	0
ttctl	UNBRANDED	4183	4183	4182	0
ltd	UNBRANDED	4185	4185	1	0
ltd	UNBRANDED	4186	4185	1	0
<b>/bin/dd</b>	UNBRANDED	4187	4187	4176	0
/bin/su	UNBRANDED	4188	4188	4151	0
/usr/local/bin/ttctl	UNBRANDED	4189	4189	4188	0

Time Frame start: 2254 s 101730325 ns end: 2270 s 869074845 ns Time Interval: 16 s 767344520 ns Current Time: 2258 s 13623345 ns

# Eclipse Linux Tools

The screenshot displays the Eclipse IDE interface for LTTng Linux tracing. The main window is titled "LTTng - Eclipse SDK". The interface is divided into several panes:

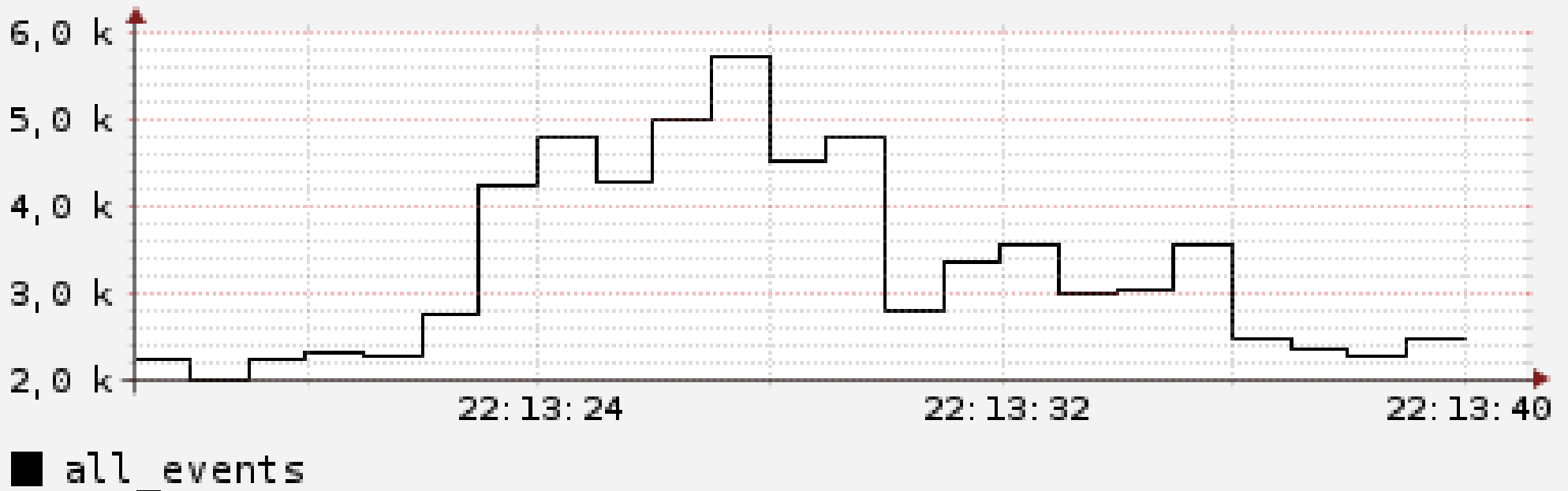
- Project Browser:** Shows a project named "MyLTTngProject" with sub-items for "Experiments [1]" and "Traces [7]". The "Traces" list includes "kernel-trace-10M", "kernel-trace-31M", "kernel-trace-4M", "kernel-trace-70M", "kernel-trace-9M", "trace-15316", and "trace-15471".
- Process List:** A table showing running processes with columns for Process, Brand, PID, TGID, PPID, CPU, Birth sec, Birth nsec, and TRACE. The processes listed are events/0, Xorg, kwin, konsole, gkrellm, and arlinux.
- Control Flow:** A horizontal bar chart showing the execution flow of the selected process (trace-15316) across different time intervals.
- Resources:** A view showing the utilization of system resources like CPU, IRQ, and SOFT\_IRQ over time.
- Events - trace-15316:** A table listing kernel events with columns for Timestamp, Source, Type, Reference, and Content. The events shown are related to kernel scheduling and signal sending.
- Histogram:** A graphical representation of event frequency over time, with a window span and center set for analysis.

Process	Brand	PID	TGID	PPID	CPU	Birth sec	Birth nsec	TRACE
events/0		5	5	2	0	13589	762949776	trace-15316
Xorg		1852	1852	1848	0	13589	763322183	trace-15316
kwin		2207	2207	2205	0	13589	763415321	trace-15316
konsole		2241	2241	1	0	13589	763465194	trace-15316
gkrellm		2259	2259	2174	0	13589	763485178	trace-15316
arlinux		3033	3036	3005	0	13589	763500334	trace-15316

Timestamp	Source	Type	Reference	Content
13589.799792434	Kernel Core	kernel/0/sched_try_wakeup	trace-15316	cpu_id:0,state:1,pid:24682
13589.799800384	Kernel Core	input/0/input_event	trace-15316	value:0,code:28,type:1
13589.799826765	Kernel Core	kernel/0/send_signal	trace-15316	signal:29,pid:1852
13589.799837369	Kernel Core	input/0/input_event	trace-15316	value:0,code:0,type:0
13589.799845650	Kernel Core	kernel/0/send_signal	trace-15316	signal:29,pid:1852



# Ittng-graph



# The Road Ahead

Live tracing

Per user global buffers

Remote tracing

Filtering (kernel)

uprobes

Other language binding

lttv



# What about research?

- Dependency analysis
- Multi level event abstraction
- Time synchronisation
- Integration of hardware tracing
- Tracing the cloud!

**Demo!**

# Contact

- Me :
  - <http://yannickbrosseau.com>
  - [yannick.brosseau@gmail.com](mailto:yannick.brosseau@gmail.com)
  - @greenscientist
- LTTng project :
  - <http://ltnng.org>
  - [ltnng-dev@lists.ltnng.org](mailto:ltnng-dev@lists.ltnng.org)
  - [irc.oftc.net #ltnng](irc://irc.freenode.net/#ltnng)

<http://ltnng.org/download>



# Image references

<https://secure.flickr.com/photos/andreweadie/3746534415>

<https://secure.flickr.com/photos/-bast-/34949798>

<https://secure.flickr.com/photos/jwyg/3746351826>

<https://secure.flickr.com/photos/pedromourapinheiro/4270640863>

<https://secure.flickr.com/photos/puuikibeach/4781504955>

<https://secure.flickr.com/photos/b4b2/3954106061>

<https://secure.flickr.com/photos/edvvc/1972546648>

<https://secure.flickr.com/photos/anirudhkoul/2632880868>

<https://secure.flickr.com/photos/iita-media-library/6031540791>

<https://secure.flickr.com/photos/38471709@N02/5353281594>

<https://commons.wikimedia.org/wiki/File:Yvain-drgaon.jpg>

[https://commons.wikimedia.org/wiki/File:Chevrolet\\_Volt\\_under\\_the\\_hood\\_WAS\\_2011\\_1107.jpg](https://commons.wikimedia.org/wiki/File:Chevrolet_Volt_under_the_hood_WAS_2011_1107.jpg)

<http://www.flickr.com/photos/15708236@N07/2754478731/>

[https://commons.wikimedia.org/wiki/File:Mount\\_Royal\\_Montreal\\_Lookout.jpg](https://commons.wikimedia.org/wiki/File:Mount_Royal_Montreal_Lookout.jpg)