LTTng 2.0 and LTTngTop Current Status

Mathieu Desnoyers
David Goulet
Julien Desfossez
Yannick Brosseau

December 9, 2011
École Polytechnique, Montreal

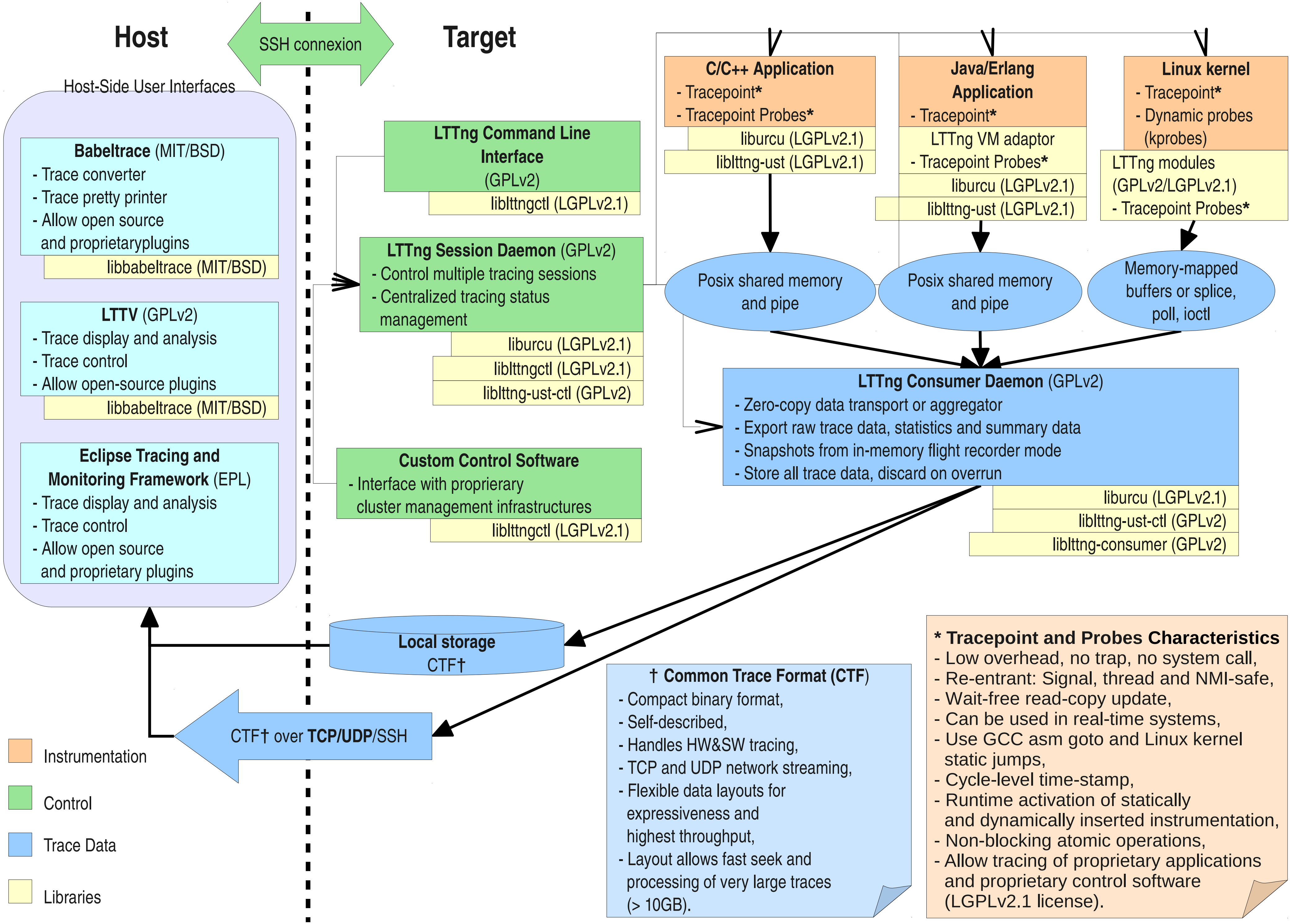
Content

- Main goals reminder
- Architecture
- Current status
- Future work
- Demos

LTTng 2.0: Main goals

- Unified interface for kernel and userspace
- Common Trace Format (CTF) suitable for software and hardware tracers
- Improve the user experience
- Remove the need for kernel patches, to ease the integration into distributions

LTTng 2.0 Low-Overhead Tracing Architecture



Host

SSH connexion

Target

Host-Side User Interfaces

Babeltrace (MIT/BSD)
- Trace converter
- Trace pretty printer
- Allow open source and proprietaryplugins

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 proprierary 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)

Local storage
CTF†

CTF† over **TCP/UDP**/SSH

† 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

Current Status

- Kernel tracing:
  - Context
  - Syscall
  - Kprobes
  - Function tracer

- Userspace:
  - Loglevel
  - Wildcard activation
  - Automatic application registration
  - Automatic tracepoint activation on registration

Current Status (2)

- Viewers:
  - Babeltrace
  - LTTngTop
    - cputop, perftop, Iotop
    - offline, live

- Distributions
  - Daily packages in LTTng Ubuntu PPA

Demos

LTTng 2.0
LTTngTop

Future plans

- New prerelease bundle with UST next week
- Networked consumer
- Trace clock (Kernel infrastructure + CTF description)
- System wide per user buffer for UST
- Data aggregation
- Statedump
- LTTV