# Vampir

# Performance Monitoring

The analysis of event data with the Vampir tool chain is an effective approach to optimize the performance of parallel applications. Collecting the performance data in a scalable and efficient fashion is a highly challenging task. VampirTrace is our approach to providing a convenient measurement infrastructure for recording fine grained performance events, with a special focus on parallel applications.
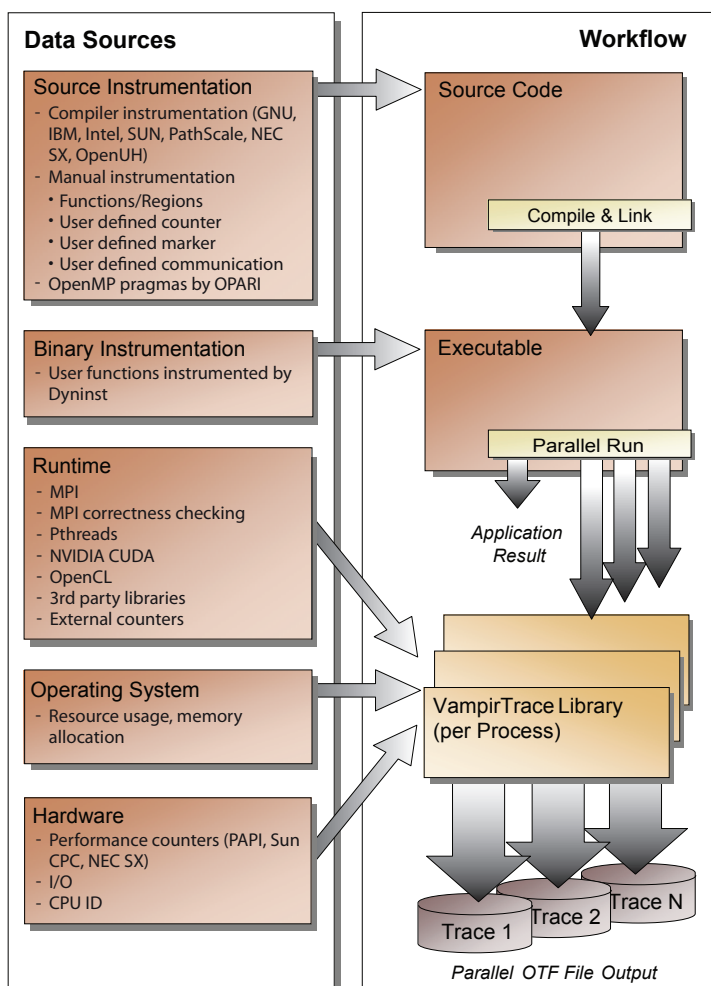
## Workflow

The performance monitoring tool Vampir-Trace collects event data during the execution of an instrumented application and writes trace files, using the parallel Open Trace Format. These trace files contain fine grained performance events of sequential, thread parallel, process parallel, or accelerator-based applications.

**VampirTrace** supports an extensive set of events such as function and library calls, communication events, and hardware counters. To collect this information, VampirTrace supports various instrumentation methods, including source level instrumentation as well as instrumentation at compile/link time.

**VampirTrace** is highly scalable, supporting platforms with more than 100K cores. A powerful filtering feature reduces the amount of information stored in temporary main memory buffers. Each process/thread flushes such a buffer to disk once it is full or the application completes. An additional unification step is used to post-process and synchronize the local traces into a global trace.
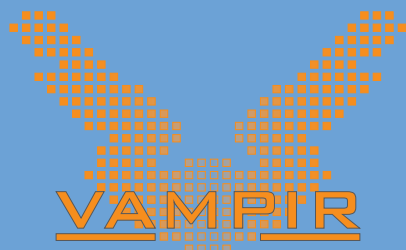
## Availability

**VampirTrace** is provided as Open Source under a BSD license. Supported platforms include IBM BlueGene, Cray XT, Linux, IBM AIX, SGI, Sun, Mac OS X, and NEC SX series. The tool is developed at ZIH, TU Dresden, Germany in cooperation with JSC, Research Center Jülich, Germany and the Innovative Computing Laboratory at University of Tennessee, USA. VampirTrace is part of every installation of the popular Open MPI library.



**Data Sources**

**Source Instrumentation**
- Compiler instrumentation (GNU, IBM, Intel, SUN, PathScale, NEC SX, OpenUH)
- Manual instrumentation
  - Functions/Regions
  - User defined counter
  - User defined marker
  - User defined communication
- OpenMP pragmas by OPARI

**Binary Instrumentation**
- User functions instrumented by Dyninst

**Runtime**
- MPI
- MPI correctness checking
- Pthreads
- NVIDIA CUDA
- OpenCL
- 3rd party libraries
- External counters

**Operating System**
- Resource usage, memory allocation

**Hardware**
- Performance counters (PAPI, Sun CPC, NEC SX)
- I/O
- CPU ID

**Workflow**

Source Code → Compile & Link → Executable → Parallel Run → *Application Result*

VampirTrace Library (per Process)

Trace 1, Trace 2, Trace N

*Parallel OTF File Output*

VampirTrace workflow and performance data sources

## Key Features
- Portable, open source performance monitor
- Extensive set of performance data sources
- Highly scalable, supporting platforms with more than 100K cores
- Supports the widely used Open Trace Format (OTF)
- Integrated in Open MPI

# VAMPIR