TECHNISCHE UNIVERSITÄT DRESDEN

Zentrum für Informationsdienste und Hochleistungsrechnen – TU Dresden

Vorstellung Workpackage 3 ADA-FS

ADA-FS Kick-Off

Sebastian Oeste

8. Februar 2016

Room: WIL A315
Address: Zellscher Weg 12
Phone: +49 351 463-32041
E-Mail: sebastian.oeste@tu-dresden.de
XMPP: soeste@tu-dresden.de

ZIH
Center for Information Services & High Performance Computing

Gliederung

1. Arbeitspaket 3.1
   - Topology Discovery
2. Arbeitspaket 3.2
   - Dynamic Resource Usage Tracking
   - Integrated Monitoring
3. Arbeitspaket 3.3
   - Access Monitoring
   - Monitoring Visualization

Topology Discovery and Dynamic Resource Tracking

- Erfassen der Ressourcen, die auf dem HPC-System zur Verfügung stehen.
- Bereitstellen einer *System-Map* aller Compute-Nodes und Speicherkomponenten *(statische Komponente)*.
- Darstellung dynamisch genutzter (geteilter) Ressourcen von parallelen Anwendungen *(dynamische Komponente)*.

Ziel:

Die Differenz aus der *System-Map* und der dynamischer Ressourcen-Auslastung sind die Ressourcen, die tatsächlich zur Verfügung stehen.

System-Map

System-Map

Die System-Map soll einen Überblick über die HPC-Plattform und die zur Verfügung stehenden Ressourcen der Plattform bieten.

- Compute-Nodes mit Anzahl der Cores.
- Verfügbarer Arbeitsspeicher pro Knoten.
- Verfügbarer lokaler Speicher (HDD, SSD, NVRAM).
- Zentrales Dateisystem mit verfügbaren Netzbandbreiten.
- Bandbreite zwischen den Compute-Nodes.

Umsetzung via SLURM

Umfangreiche Plugin-Architektur z. B. :

- Topology-Plugins
- Node-Selection-Plugins

Slurm speichert die Topologie in einer Datei /etc/slurm/topology.conf.
Bietet Informationen zu:

- Anzahl Compute-Nodes
- Anzahl Switches
- Verbindung zwischen Switches und Compute-Nodes

Topology Guide

At some point in the future Slurm code may be provided to gather network topology information directly. Now the network topology information must be included in a topology.conf configuration file. . .

[1]

[1] http://slurm.schedmd.com

hwloc - The portable Hardware Locality

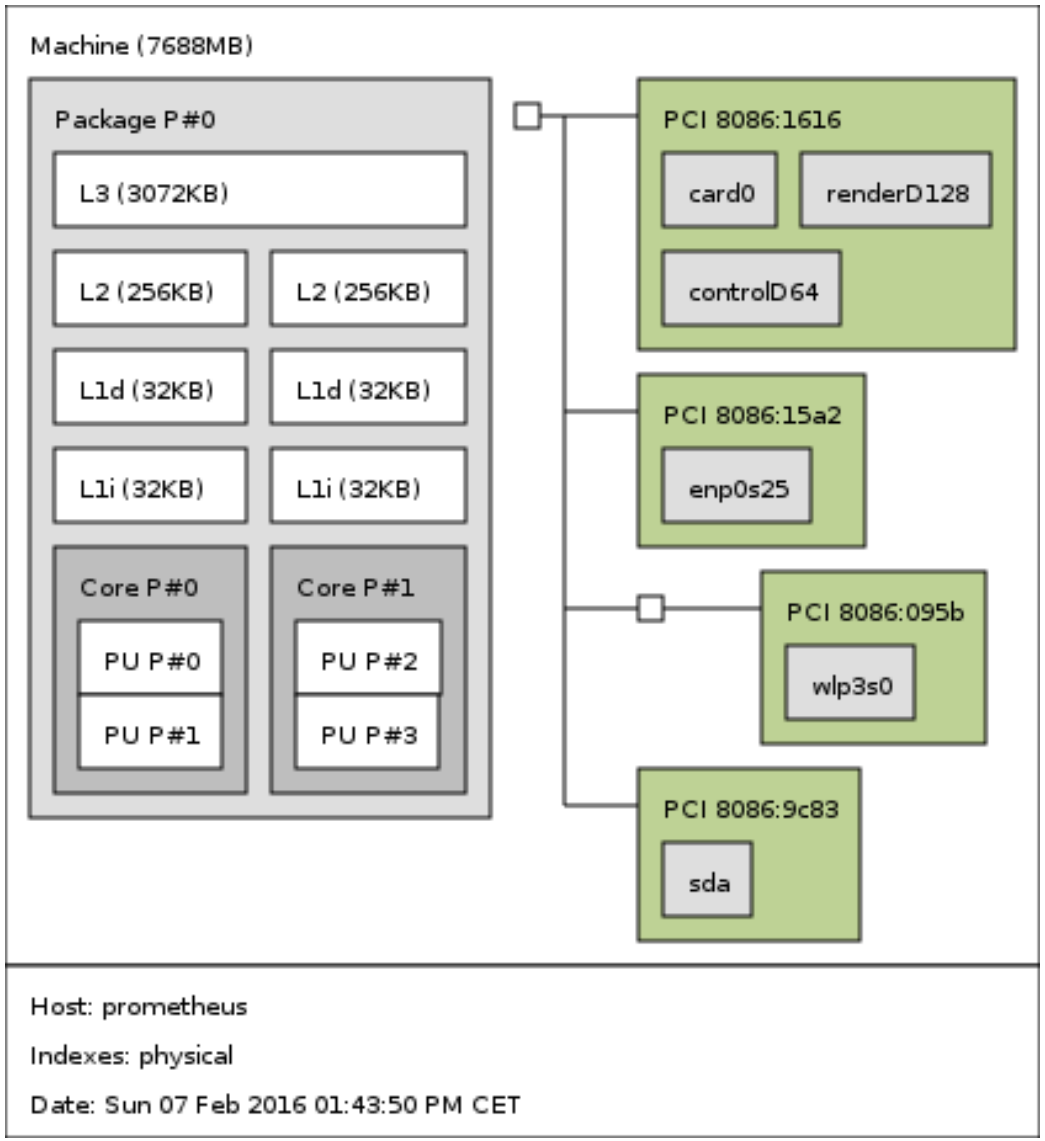
Open-MPI Projekt: Bibliothek und Tools.
Umfasst CLI und C-API.

- Sammelt alle verfügbaren Informationen vom OS.
- Erstellt abstrakten Baum von Objekten z.B.: memory nodes, caches, processors, sockets, processor cores, ...
- Lokalität von I/O devices z.B.: network interfaces, Infiniband HCAs oder GPUs.
- PCI Geräte Erkennung z.B.: OpenCL, CUDA und Xeon Phi Beschleuniger.

In vielen Projekten bereits genutzt.

- PaRSEC
- StarPU
- TORQUE - resource manager
- likwid

hwloc - The portable Hardware Locality

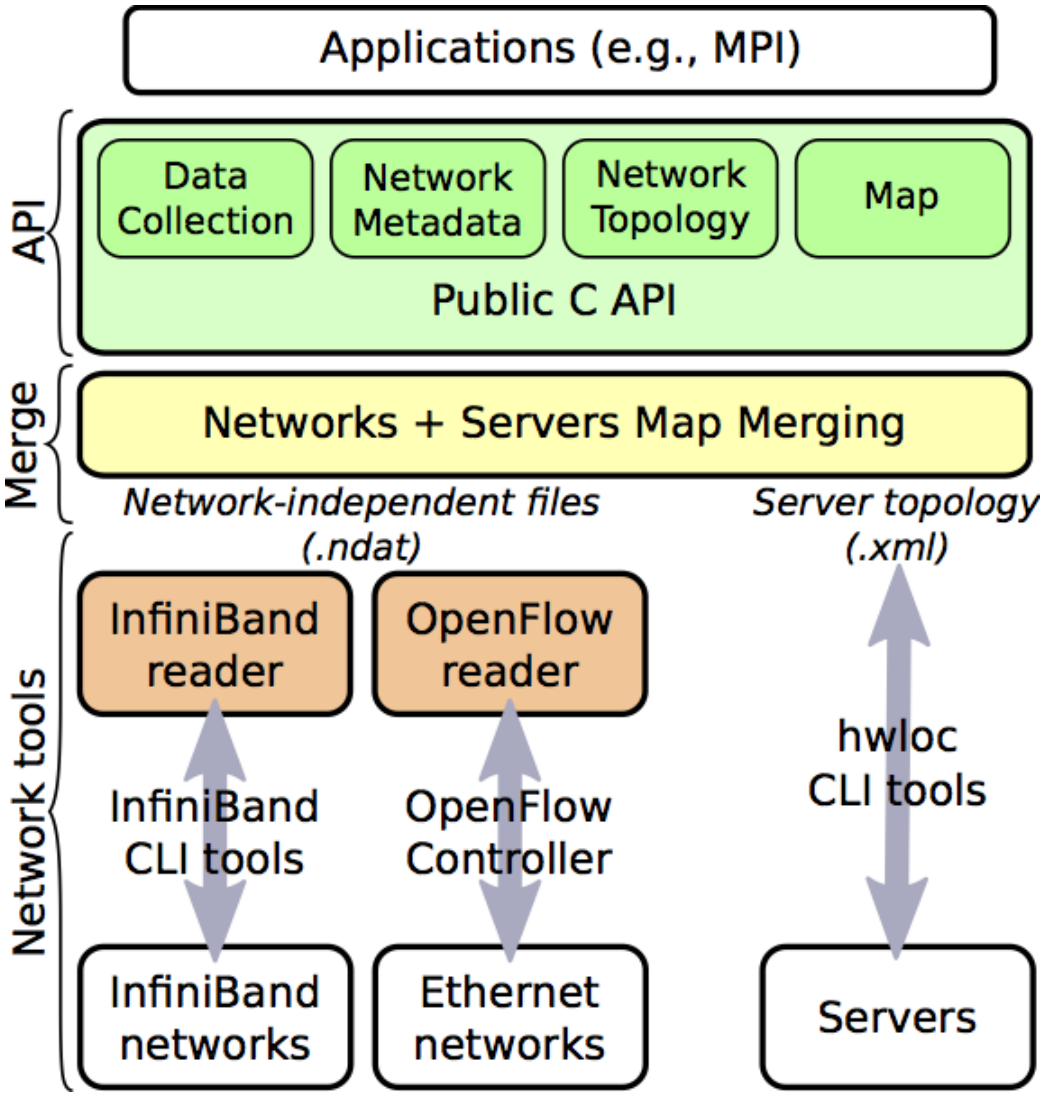


netloc - Network Locality

Open-MPI Projekt.
Ermöglicht Informationen über die Netzwerktopologie von HPC-Clustern zu identifizieren.

- Abstrakte Representation als Graph.
- Unterstützung für Ethernet und Infiniband Netzwerke.
- In Kombination mit *hwloc* umfassende Sicht der Komponenten verschiedener Knoten und dem dazwischen liegendem Netzwerk.

netloc - Network Locality



https://www.open-mpi.org/projects/netloc/netloc-design-full-size.png

Vorschlag

- Entwicklung eines Tools auf Basis von hwloc und netloc zum identifizieren der Topologie.
- Einfaches Interface zum Abfragen der Informationen (einfach integrierbar).
- Unterstützung mehrerer Ausgabeformate z.B. als OTF2 Stystem-Tree.

Dynamische Ressourcen Auslastung

Dynamische Ressourcen Auslastung

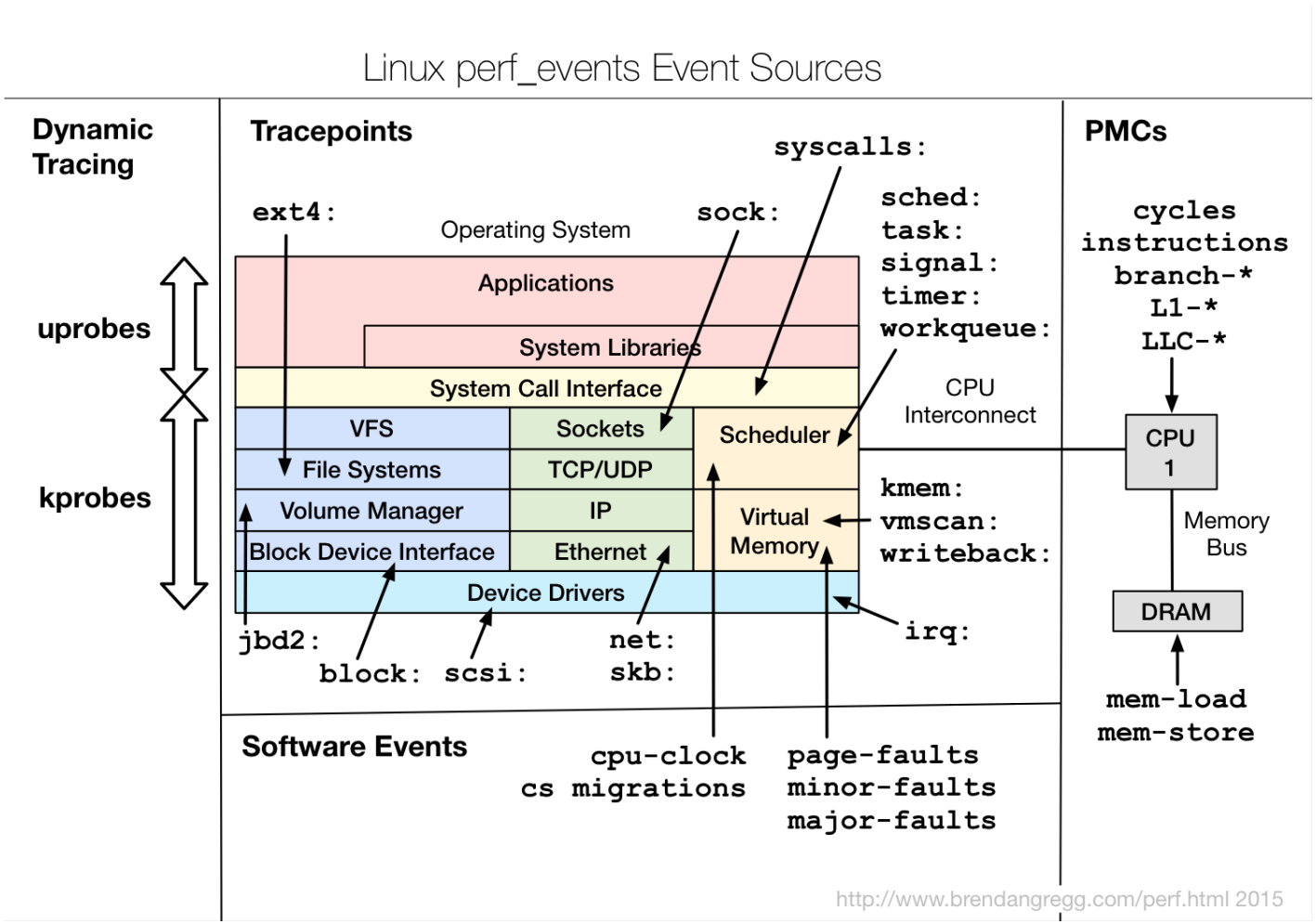
Aktuelle Auslastung der Ressourcen einer Menge von Knoten.
Wie hoch ist die Ressourcen Auslastung einer Anwendung?

- Auslastung von Arbeitsspeicher und lokalen Speicher, pro Knoten oder NUMA-Domain.
- Lastmessung zwischen Zentralen Dateisystem und Compute-Node.
- Auführung, nebenläufig zur Anwendung.
- Der Anwendungscode soll nicht verändert werden.

perf

- Events Subsystem im Linux-Kernel.
- Instrumentiert Hardware Performance Counter, tracepoints, KProbes, UProbes.
- geringer Overhead.

perf



http://www.brendangregg.com/perf_events/perf_events_map.png

Integrated Monitoring of the Ad-hoc File System

- Integration der Monitoring Komponenten in das Ad-hoc Dateisystem.
- Überwachen von I/O Aufrufen an vier Schnittstellen.
  1. Aufrufe von der Anwendung an das Dateisystem.
  2. Aufrufe des Ad-hoc Dateisystems zum unterliegenden Dateisystem.
  3. Anfragen vom *data transfer executor*.
  4. Interner Datentransfer innerhalb des Ad-hoc Dateisystems.
- Aufgezeichnete I/O-Events werden in einer Datenbank gespeichert.

Ziel

Analyse des Ad-hoc Dateisystems. Entwickelte Komponenten bieten Informationen zur Entwicklung und Optimierung des Systems.

Access Monitoring through Applications

- Monitoring von Datenzugriffen
- Wie oft kann eine I/O Anfrage lokal bearbeitet werden?
- Wie oft muss für eine I/O Anfrage kommuniziert werden?
  Wieviel zusätzliche Zeit wird dafür benötigt?
- Welche Dateien oder Dateisystemblöcke, werden von welchen Prozessen oder Knoten zugegriffen?

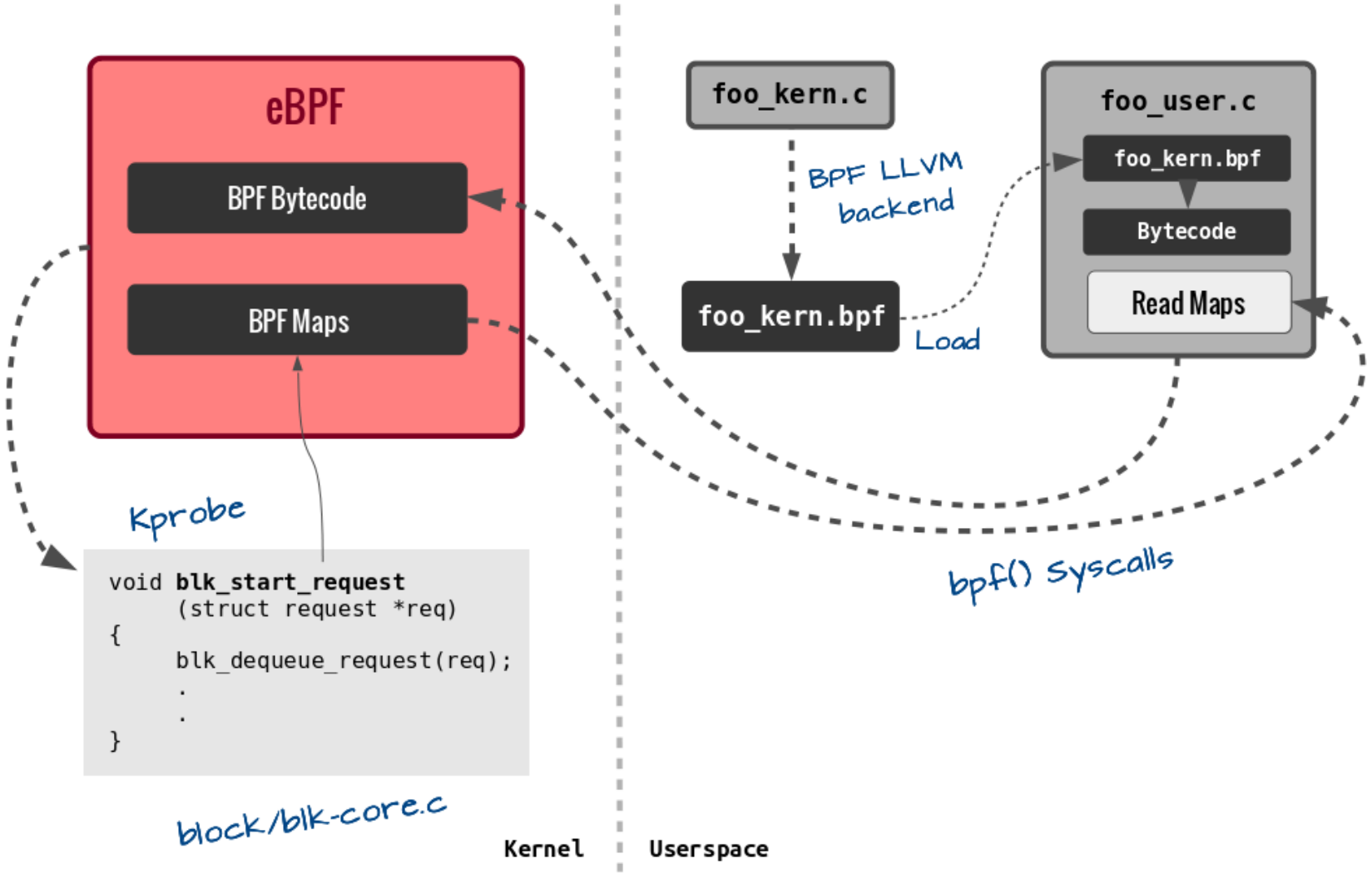
Ziel

Diese Informationen stellen den Input für das "Data-Placement"da. Die Ergebnisse werden "pro Anwendungs-Lauf"gespeichert. Das Format soll einfach aggregierbar sein und in einer Datenbank gespeichert werden.

eBPF - extended Berkley Packet Filter

- virtuelle Maschine innerhalb des Kernels
- BPF Programm wird aus dem User-Space geladen.
- Just-in-Time Compiler konvertiert BPF Code in nativen Bytecode.
- Ausführung hängt an einem KProbe oder UProbe.
- abgefragte Daten werden in den User-Space kopiert
- benötigt aktuellen Kernel

eBPF - extended Berkley Packet Filter



https://suchakra.files.wordpress.com/2015/08/ebpf-session.png

Workflow: Monitoring

1. Eintrag pro Anwendungslauf.
2. Abfragen der Informationen über Monitoring-Schnittstellen.
3. Speichern in der Datenbank.
4. Aggregationen in der Datenbank.

Auswertung

- Tools zur Datenbankabfrage der Monitoring Daten.
- Support für verschiedene Ausgabeformate.
- Visualisierung der Daten.