CPU and DRAM power estimations of software containers

Guillaume Fieni

University of Lille

July 2018

Université de Lille

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

Context & Challenges

- Data-intensive software systems
  - Process continuous flows of data (smartphones, IoT...)
  - Deployed in distributed environments (public/private)
  - Two key concerns: security and energy
- Explores the security/energy trade-offs and optimizations
- Maximize the security while minimizing the power consumption

Related work

- Static power models: BitWatts
- Per-process power estimations
- Use Hardware Performance Counters
- Use of external power meter (IPMI, PowerSpy...)
- Requires an offline calibration phase
- Accurate (2-3% error)
- No DRAM power estimations

SmartWatts

- Self-adaptive power meter
- Provides per-contaienr CPU and DRAM power estimations
- Lightweight
- No external power-meter required

Introduction

Software containers

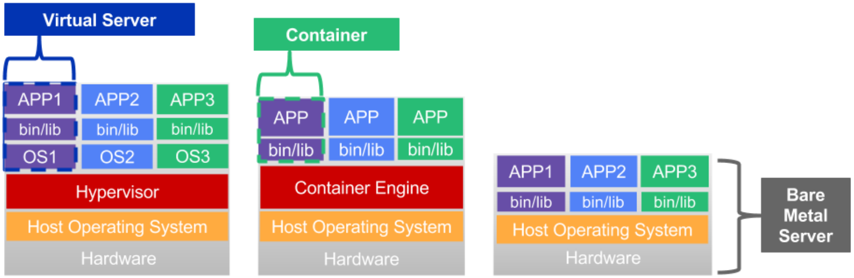


Figure 1: Different types of software containers

SmartWatts

Software containers

- Linux's control groups (cgroups)
- Used by: Docker, Libvirt, Systemd
- perf_event supports per-container monitoring

SmartWatts

Hardware Performance Counters

- Low overhead
- High amount of available events
- Limited amount of simultaneous events
- Virtualized by perf_event
- Need to select the most relevant events (correlation)

SmartWatts

Hardware Performance Counters - Selection

- Heuristic based
- Needed for CPU and DRAM models (different indicators)
- On newest architectures: quick, almost same results
- On oldest architectures: slower, various results

SmartWatts

Running Average Power Limit (RAPL)

- Available on Intel (since Sandy Bridge) and AMD (since Zen)
- Power capping feature
- Provides power estimations for CPU and DRAM
  - Multiple domains: Package, Core, Uncore, DRAM
  - Package wide power estimations

SmartWatts

Power models

- Learning power models at runtime
- Triggers learning of new power model if error $>=$ threshold
- Build a robust power model over the time
- Multivariate linear regression

Hardware optimizations

C-states

- Power optimization (idle states)
- Disable some parts of the CPU when unused
- Reduce greatly the power consumption
- Hardware Performance Counters are not correlated anymore

Hardware optimizations

C-states - IDLE

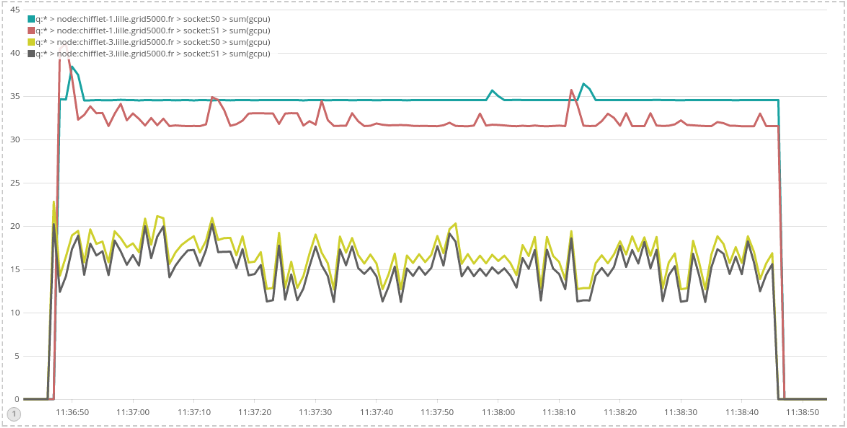


Figure 2: CPU power comsuption with C-states enabled/disabled

Hardware optimizations

P-states

- Performance optimization (operational states)
- Reduce power consumption without impacting performance
- Hardware P-states on $>=$ Skylake micro-architectures
- Boost states
- Model does not predict the correct power consumption

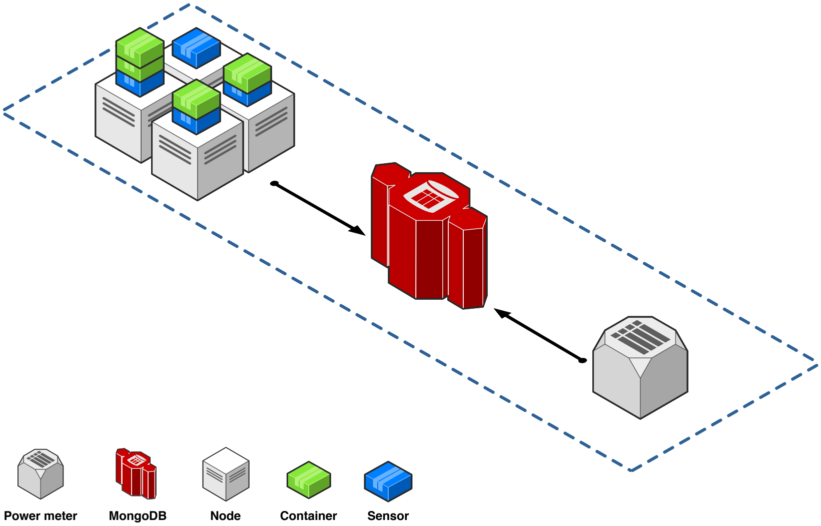
Hardware optimizations

Advanced Vector Extensions (AVX)

- SIMD instructions
- Have designated turbo frequencies
- Affect all cores when using AVX2 or AVX-512
- Model does not predict the correct power consumption

SmartWatts

Architecture



SmartWatts

Evaluation

- Qarnot computing Heater [1]
  - Docker
  - CPU rendering of 3D graphics (Blender)
- Cluster infrastructure
  - Kubernetes
  - Typical HPC workloads (NPB, Linpack)
  - CPU and Memory intensive
  - Deployed on the Grid'5000 testbed infrastructure

[1] `https://www.qarnot.com/`

Future perspectives

- Extend SmartWatts to other architectures (AMD, ARM)
- Power estimations of Intel SGX secure enclaves
- Energy aware scheduling of distributed environments