MongoDB I/O Access Patterns are under the Microscope
Abdulqawi Saif, Lucas Nussbaum, Ye-Qiong Song

To cite this version:
Abdulqawi Saif, Lucas Nussbaum, Ye-Qiong Song. MongoDB I/O Access Patterns are under the Microscope. Annual PhD students conference IAEM Lorraine 2017, Oct 2017, Nancy, France. 2017. hal-01793531

HAL Id: hal-01793531
https://inria.hal.science/hal-01793531
Submitted on 16 May 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
I. Context

- MongoDB is a document-based NoSQL database
- NoSQL databases store semi-structured and/or unstructured data
- We need to create indexes to optimize querying data
- Based on certain industry reports, MongoDB takes unjustified time to create indexes over a pre-stored data
- Benchmarks report high-level results in general, they also use synthetic data (≠ in-production data)

II. Objectives

- Investigation on creating indexes by MongoDB
- Introducing new experimental methods that could go beyond benchmarking high-level results

III. Experimental settings

- A data set of 20 million doc. is created; about 71 GB
- Experiments are performed on Grid’5000 testbed
- Ubuntu 14.04, Linux 4.9.13
- Tests are performed on one HDD per machine
- MongoDB V3.4, replicas are disabled, hash sharding, sharding on _id
- MongoDB storage engine (WT) stores each collection on a separate file

IV. Performance results

- It is a recent dynamic tracing technique in Linux
- It could connect to all Linux data sources
- Very negligible overhead (4 ns per syscall)
- Fits with systems in production

An eBPF tool is built to evaluate the I/O access patterns:

- A generic tool, could be used by other systems
- It traces all I/O requests
- It reports offsets, request latencies & data size
- Its results could be filtered by accessed files

VI. I/O access patterns using eBPF

- How to get the main raison behind these results?

V. extended Berkeley Packet Filter (eBPF)

- A performance study on MongoDB I/O access pattern is done
- A generic eBPF tool for testing I/O access patterns is developed
- A new experimental method to go beyond benchmarking results is introduced

VII. Conclusion

Acknowledgment

This work is partially funded by the Xilopix SAS, a French startup which is behind Xaphir search engine.