The Storage Performance Analyzer: Measuring, Monitoring, and Modeling of I/O Performance in Virtualized Environments

[Invited Demo Paper]

Qais Noorshams*, Axel Busch*, Samuel Kounev**, Ralf Reussner*
*Karlsruhe Institute of Technology, Germany ([lastname]@kit.edu)
**University of Würzburg, Germany (samuel.kounev@uni-wuerzburg.de)

ABSTRACT

The ever-increasing I/O resource demands pose significant challenges for today's system environments to meet performance requirements. The resource demand effects are even magnified in modern virtualized environments where workloads are consolidated to save hardware and operating costs. Tool-supported analysis approaches can help to understand I/O performance characteristics and avoid I/O performance and interference issues. In this demo paper, we present the Storage Performance Analyzer (SPA) – a tool for automated I/O performance analysis. SPA is equipped with tailored features for virtualized environments allowing to measure, monitor, and model both I/O performance and interference effects in modern environments. SPA is open-source and available for common operating systems.

1. THE SPA APPROACH

The *Storage Performance Analyzer (SPA)* [1] is an approach for the systematic analysis of I/O performance in virtualized environments, which has been successfully applied in our previous work for performance measuring, monitoring, and modeling [2–8]. As illustrated in Figure 1, our SPA framework basically consists of a benchmark harness that coordinates and controls the execution of benchmarks as well as monitors and a tailored analysis library used to process and evaluate the collected measurements.

Measuring. Using integrated I/O benchmarks, SPA coordinates the execution of benchmark runs on possibly multiple targets (e.g, on co-located virtual machines) to obtain measurements of the I/O performance. Currently, we have integrated two benchmarks into our framework, but further benchmarks can be integrated as required. We use the open source *Flexible File System Benchmark*[1] (FFSB) for a fine-grained analysis and the *Filebench* benchmark[2] to emulate mixed application workloads, e.g., a file server workload. After the measurement setup has been configured, the SPA framework first configures the benchmarks, then it executes the target workload, and it finally collects the results.

Monitoring. During the measurement process, the system environment as well as specific targets can be monitored to observe the I/O performance behavior. SPA can activate operating system monitors, such as `blktrace`[3] and `iostat`[4], as well as self-defined monitors, e.g., logging the amount of files the benchmarks produce. The number of executed monitors is not limited and more monitors can be included if needed. If any monitors are activated, SPA starts all monitors before the benchmarks are started. After all benchmarks are finished, SPA stops all monitors and collects the results.

Modeling. The measurement and monitoring results can be processed and analyzed for a variety of purposes, e.g., to identify performance bottlenecks and performance interference effects. The results can also be used for performance modeling. SPA includes analysis libraries enabling fully-automated tuning and modeling using statistical regression techniques.

2. ARCHITECTURE

Components. The benchmarking component, which is implemented in Java, contains a *benchmark controller* that explores the parameter space and coordinates the benchmark runs accordingly. The benchmark controller is connected to the *benchmark driver*, which is an abstraction of the actual benchmark used. In addition, the measurement process can be monitored using a given *monitor driver*. The benchmark controller and the drivers are deployed on a controller machine managing the measurement process. The drivers use an internal *remote execution component* to communicate with the actual benchmark and monitors, which are deployed on the SUT. In our implementation, the remote execution component employs SSH connections, but it could be easily changed to use another connection type. The benchmark controller saves the results using the persistence component.

The performance modeling component is integrated into the open source statistics tool R[5]. The *datastore interface* can load and prepare the data, e.g., by filtering or aggregating data, to evaluate the results. Both the *regression optimization* and the *regression modeling* component can further process this data or use other data specified by the user. The regression optimization component comprises an automated regression parameter tuning process for given training data [3] and uses the *regression techniques* whose

[1] `https://github.com/FFSB-prime`
[2] `https://github.com/Filebench-Revise`

ICPE'15, Jan. 31–Feb. 4, 2015, Austin, Texas, USA.
ACM 978-1-4503-3248-4/15/01.
http://dx.doi.org/10.1145/2668930.2693845.

[3] `http://linux.die.net/man/8/blktrace`
[4] `http://linux.die.net/man/1/iostat`
[5] `http://www.r-project.org/`

implementations are provided by *R* libraries. The regression modeling component automatically creates the models with the considered regression techniques.

Design Decisions. This design has several advantages: i) Using SSH connections, the VMs deployed on the system under test (SUT), on which the experiments are executed, are not required to have additional software installed as, e.g., in the case of using Java Remote Method Invocation (RMI). Such solutions would require an additional abstraction layer on the VMs, which is often difficult to debug in case of unexpected results. ii) Furthermore, this enables the benchmark controller to take control over the synchronization of the measurements from multiple VMs. The benchmark controller starts the benchmarks simultaneously and does not require, e.g., network time protocol (NTP). iii) Measurement results are saved asynchronously in a lightweight SQLite database that easily supports SQL requests and CSV export for versatile data access. Moreover, SQLite is supported natively and deamonless by many operating systems and programs. iv) The analysis library is integrated into the statistics tool *R*. The analysis library can be easily extended for new functionality and all functionality of *R* can be re-used with minimal effort.

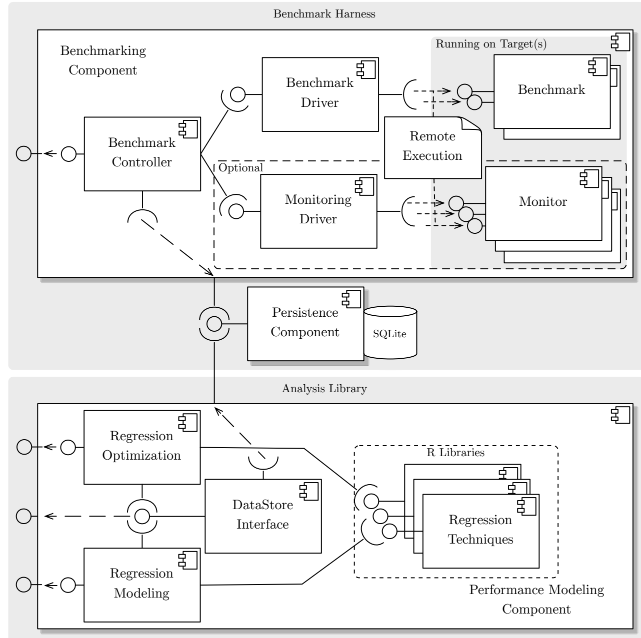

Figure 1: SPA Overview

3. APPLICATION SCENARIOS

SPA enables a wide range of I/O performance analysis and we used it in the following application scenarios:

1. Identifying and evaluating important performance influences of I/O-intensive applications is a prerequisite for systematic performance analysis [2]. Using such information, the monitoring features of SPA can be used to extract performance characteristics of running I/O-intensive applications in virtualized environments [8].
2. SPA can automatically explore and quantitatively evaluate both workload-specific and system-specific performance influences. This information can in turn be used in an automated process for statistical analysis and regression-based performance modeling [3, 5, 6]. These models can be used to analyze the system behavior and predict the performance in different scenarios.
3. With understanding the system behavior and performance impact of I/O-intensive workloads, sophisticated model formalisms, e.g., queueing networks or queueing Petri nets, can be applied [4, 7]. Creating such models requires increased manual effort, but benefit from usually high predictive power and potential for reuse, since once created, the number of required calibration measurements are relatively low.

4. CONCLUSION

In this paper, we presented SPA – a tool for analyzing the I/O performance of storage systems specifically targeting virtualized environments. The main benefits of SPA are the automated exploration of the configuration space (i.e., the possible configuration parameters are specified and all combinations comprised of benchmarks and monitors are executed automatically) and the built-in support for a synchronized execution on multiple targets (for example virtual machines on separate physical hosts), i.e., multiple benchmarks are started at the same time without manual coordination. While the approach is tailored and pre-packaged for benchmarking and monitoring the I/O performance of storage systems in both native and virtualized environments, SPA is not limited to a specific domain and can be extended to integrate other benchmarks and monitoring tools. SPA is freely available and can be downloaded from the project page [1] and the SPEC RG Tool Repository: `http://research.spec.org/tools`.

Acknowledgments This work was supported by the German Research Foundation (DFG) under grant No. RE 1674/5-1 and KO 3445/6-1, and the German Federal Ministry of Economics and Energy (BMWI), grant No. 01MD11005 (PeerEnergyCloud). We especially thank the Informatics Innovation Center (IIC) – `http://www.iic.kit.edu/` – for the support of this work.

5. REFERENCES

[1] The Storage Performance Analyzer (SPA). `http://storageperformanceanalyzer.github.io/SPA/`.
[2] Qais Noorshams, Samuel Kounev, and Ralf Reussner. Experimental Evaluation of the Performance-Influencing Factors of Virtualized Storage Systems. In *EPEW '12*.
[3] Qais Noorshams, Dominik Bruhn, Samuel Kounev, and Ralf Reussner. Predictive Performance Modeling of Virtualized Storage Systems using Optimized Statistical Regression Techniques. In *ICPE '13*.
[4] Qais Noorshams, Kiana Rostami, Samuel Kounev, Petr Tůma, and Ralf Reussner. I/O Performance Modeling of Virtualized Storage Systems. In *MASCOTS '13*.
[5] Qais Noorshams, Axel Busch, Andreas Rentschler, Dominik Bruhn, Samuel Kounev, Petr Tůma, and Ralf Reussner. Automated Modeling of I/O Performance and Interference Effects in Virtualized Storage Systems. In *DCPerf '14*.
[6] Qais Noorshams, Roland Reeb, Andreas Rentschler, Samuel Kounev, and Ralf Reussner. Enriching software architecture models with statistical models for performance prediction in modern storage environments. In *CBSE '14*.
[7] Qais Noorshams, Kiana Rostami, Samuel Kounev, and Ralf Reussner. Modeling of I/O Performance Interference in Virtualized Environments with Queueing Petri Nets. In *MASCOTS '14*.
[8] Axel Busch, Qais Noorshams, Samuel Kounev, Anne Koziolek, Ralf Reussner, and Erich Amrehn. Automated Workload Characterization for I/O Performance Analysis in Virtualized Environments. In *ICPE '15*.