A Software-Defined Approach for QoS Control in High-Performance Computing Storage Systems

Neda Tavakoli[1], Dong Dai[1] , John Jenkins[2] , Philip Carns[2] , Robert Ross[2] , Yong Chen[1]
[1]Department of Computer Science, Texas Tech University,
[2]Mathemathic and Computer Science Division, Argonne National lab

Abstract

- **Objectives**
  - To meet the Quality-of-Service(QoS) requirements for HPC platforms
  - To propose a flexible and effective storage system using software-defined approach to assure a certain level of resources per application
- **Background**
  - The ability to guarantee a certain level of performance is called QoS support
  - As larger storage systems are using, multiple applications which are sharing the same storage systems, will compete and interfere with each other
  - Application interference dramatically degrades the overall system performance
- **Contribution**
  - We proposed a flexible solution to achieve soft QoS storage guarantees using software-defined approach
  - We proposed borrow model and policies to meet QoS requirements

Proposed Architecture

- We propose to use software-defined technique as Figure 1 shows
- Two key software-defined components are added into HPC storage system to enable a flexible QoS provisioning
  - **Data plane:** It is running on each storage server for IO classification and bandwidth shaping
    - Data plane contains multiple queues, each of which buffers requests from a given application.
    - IO classification is done based on the IO header
    - IO requests with the same IO header belong to the same application and will be put to the same queue (in the data plane
  - **Control plane**: The control plane consists of the several components
    - **Token Rate Generator:** It communicates with the Desired QoS component in data plane to sync the requested bandwidth specification of each application and generates a token rate per application
    - **Virtual Token Buckets**: It learns the token rate from token rate generator to operate
    - **Traffic Shaper:** It communicates with the virtual token buckets to get information to shape the traffic
    - **Policy Enforcer:** It is used to deliver policies to meet the QoS requirements

Architecture & Challenges

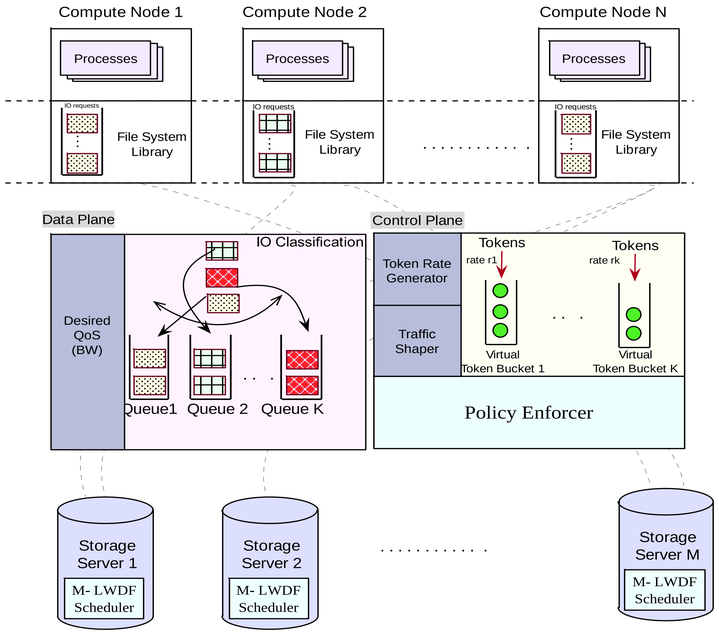


Figure 1. Quality of service architecture using software-defined approach

- **[Challenge] Unbalanced IO Requests**
  - HPC application may request IOs in an unbalanced way. These might cause problems facing the fair sharing of tokens among storage servers
  - Figure 2 shows how an example of this case
- **[Challenge] Physical Limitation**
  - Each storage has a physical bandwidth limitation
  - Reasons of violating physical limitation:
    - IO requests from a single application
    - Concurrent data accesses from multiple applications

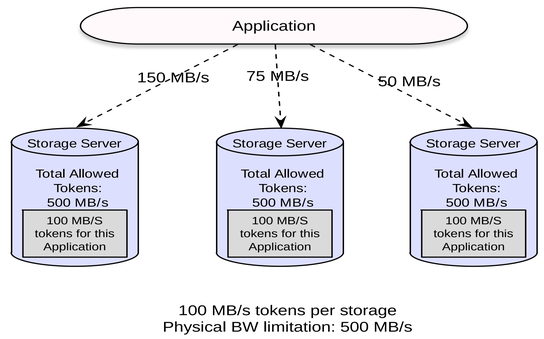


Figure 2. Unbalanced IO request

Proposed Solution

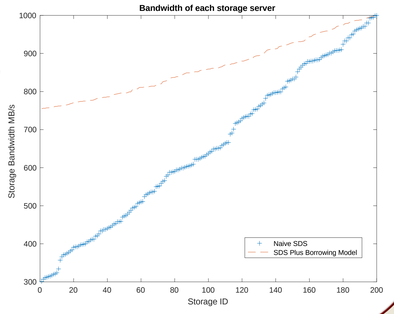
- **Proposed Software-Defined Solution**
  - Based on the software-defined architecture, we propose several key improvements:
    - We introduce borrowing model which allows a queue of one application to borrow tokens from queues of the same application in other storage servers.
    - We extend the original M-LWDF algorithm in conjunction with the borrowing model. The extended M-LWDF is used to guarantee the fairness during degradation
    - We design a set of policies regarding the borrowing model, including whether the borrow can happen and when and how many tokens should be borrowed.

Preliminary Results

- **Borrowing policies**
  - Prohibit an application to borrow tokens:<app-i,borrow=FALSE>
  - Allow an application to borrow tokens: <app-i, borrow=TRUE>
  - Allow an application to borrow tokens if only threshold percent of its required bandwidth is satisfied: <app-i, borrow=TRUE, thres=0.8>
  - Example:

Storage BW limitation: 1000 MB/s

Application	Desired MB/S	Naive SDS MB/S	Plus Borrowing Model
App1	1000	800	1000
App2	800	700	750
App3	1400	1100	1250
Total	3200	2600	3000



References

[1] Thereska, Eno, et al. "IOFlow: a software-defined storage architecture." Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. ACM, 2013.
[2] Andrews, Matthew, et al. "Providing quality of service over a shared wireless link." IEEE Communications magazine 39.2 (2001): 150-154.
[3] Stolyar, Alexander L., and Kavita Ramanan. "Largest weighted delay first scheduling: Large deviations and optimality." Annals of Applied Probability.
[4] Wu, Joel C., and Scott A. Brandt. "The design and implementation of AQuA: an adaptive quality of service aware object-based storage device." Proceedings of the 23rd IEEE/14th NASA Goddard Conference on Mass Storage Systems and Technologies, 2006.