iJRASET
International Journal For Research in Applied Science and Engineering Technology



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 3 **Issue:** V **Month of publication:** May 2015

**DOI:**

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

# Development of Comprehensive Parallel I/O Benchmarking Tool for HPC Cluster File Systems

M.Swarnalatha[1], P.Durgadevi[2], T.Chindrella Priyadharshini[3], M.Hemalatha[4]
[1,2,3]*Department of IT, RMK College of Engineering &Technology, Anna University.*
[4]*Team Lead, Infosys, Chennai*

***Abstract— High Performance Computing Clusters uses parallel file systems such as Lustre, GlusterFS for better I/O throughput with multiple clients. These file systems are implemented over high speed interconnects like 10G, Quadrics, Myinet and Infiniband. Parallel scientific applications require high performance I/O support from underlying file systems. The objective of this work is to evaluate and analyze performance of parallel file systems in HPC cluster with different configuration parameters with development of a comprehensive bench-marking tool. The developed tool measures the performance of different data I/O and meta-data operations under various workloads. This paper briefly describes various cluster file systems, parallel I/O approaches and different Message Passing Interface (MPI) implementations supporting MPI-IO and throughput results for different I/O scenarios.***
***Keywords— High Performance Computing (HPC), I/O performance, MPI, Lustre, GlusterFS and Infiniband***

## I. INTRODUCTION

The growing need for processing ever-increasing amounts of data in large-scale scientific and business applications motivates research and development on parallel file systems that can offer scalable performance. Large data-processing requirements typically translate to high I/O throughput demands from parallel file systems comprising hundreds to thousands of disks. When designing a large-scale computing installation for a scientific or industrial application, IT architects face a challenge deciding on the most appropriate parallel file system. The right choice typically depends on the specific characteristics of the application as well as the design assumptions built into the parallel file system. Parallel scientific applications require high-performance I/O support from underlying storage units and file systems. High Performance Computing (HPC) Clusters use parallel file systems such as Lustre, GlusterFS implemented over high speed interconnects like Infiniband to improve the sustained performance. An important element in the comprehensive support solution is to develop tools and techniques for testing the reliability and performance of these scalable file systems. In this paper, development of new benchmarking tool with comprehensive I/O testing functionalities is illustrated and the performance evaluation of parallel file systems with this tool is discussed. The developed tool supports new test features with intuitive user interface and it can be used to evaluate and fine tune the performance of cluster file systems for parallel I/O operations. The interconnect networks and file systems used in HPC cluster are introduced in section[II], section[III] is about MPI-IO and its implementations, section[IV] describes approaches in parallel I/O, section[V] gives overview of existing Benchmark tools, and the need for new tool is discussed in section[VI]. The I/O operations simulated for performance evaluation, configurations of benchmark and the test results are given in the remaining sections.

## II. HIGH PERFORMANCE COMPUTING

HPC uses super computers and cluster computers to solve advanced computation problems. Computer systems approaching the teraflops-region are counted as HPC systems. HPC clusters use high speed interconnect and parallel cluster file systems to achieve better efficiency.

### *A. High Speed Interconnects*

There are various HPC interconnects like Quadrics, Myinet, Gigabit Ethernet and Infiniband. Inifiniband is the most commonly used switched fabric communication link in HPC clusters. The Infiniband architecture specification defines a connection between processor nodes and high performance I/O nodes such as storage devices.

### *B. Parallel File Systems for HPC*

Large-scale scientific computation often requires significant computational power and involves large quantities of data.


©IJRASET 2015: All Rights are Reserved

However, the performance of I/O subsystems within high-performance computing (HPC) clusters has not kept pace with processing and communications capabilities. Inadequate I/O capability can severely degrade overall cluster performance, particularly in the case of multi-teraflop clusters, Parallel File System Manages storage hardware, present single view, stripe files for performance. Parallel file system connects few nodes to the storage known as I/O nodes which serve data to rest of the cluster. It may also include separate metadata server(s). Lustre, GlusterFS, PVFS, GPFS, Panasas are currently used parallel cluster file system supporting huge storage. Our reference cluster uses GlusterFS which is discussed below.

1) *GlusterFS:* GlusterFS is a distributed file system capable of scaling to several petabytes and handling thousands of clients. GlusterFS clusters together storage building blocks that include direct attached storage, JBOD (Just a Bunch of Disks), as well one or more SAN (Storage Area Network) fabrics, aggregating disk and memory resources and managing data in a single global namespace. GlusterFS is based on a stackable user space design and can deliver exceptional performance for diverse workloads. Clients can use one of several protocols, including the GlusterFS Native Client, NFS(Network File System), and CIFS(Common Internet File System), among others, to access the global namespace connected through Gigabit Ethernet (GbE), 10 Gigabit Ethernet (10 GbE), and Infiniband. It provides simultaneous access for multiple clients to a single shared file system, it uses FUSE (File system in User SpacE) to allow cluster nodes to store and retrieve files from a central location.

## III. MESSAGE PASSING INTERFACE

Message Passing Interface is a standard and portable message-passing system designed to function on a wide variety of parallel computers. All implementations of MPI include parallel functions but not parallel I/O functions, which are available only in MPI-2 implementations.

### *A. MPI Implementations*

There are several Linux supported MPI implementations like, MPICH, MVAPICH**,** Intel MPI and OpenMPI. An overview of Intel MPI which is used in our reference machine is given below.

1) *Intel MPI:* Intel MPI Library focuses on making applications perform better on IA based clusters by implementing the high-performance MPI-2 specification on multiple fabrics. The Intel MPI Library dynamically selects the most appropriate fabrics for communication between MPI processes. Thus an MPI program compiled with Intel MPI can run, for example, over both the gigabit Ethernet and Infiniband fabrics, with no re-compilation needed.

### *B. MPI-IO*

MPI-IO is an I/O interface specification for use in MPI applications. The MPI-IO provides high performance, portable, parallel I/O interface to high performance, portable, parallel MPI programs. MPI-IO refers to a set of functions designed to abstract I/O management on distributed systems to MPI, and allow files to be easily accessed in a patterned way using the existing derived data type functionality. MPI-IO features include, parallel read/write, non-contiguous data read/write, Non-blocking I/O, Collective I/O, Portable data representation across platforms**.**

## IV. PARALLEL I/O APPROACHS

Reading and writing to files is often called file input and output, or file I/O for short. Input/output operations such as open, close, read, write and append, all of which deal with standard disk files. Performing multiple I/O operations on a file at the same time is called parallel-IO.

### *A. Independent File Access*

This approach enables each process accessing independent files, method produces numerous files that can get difficult to manage, and it may not be convenient to restart the calculation on a different number of nodes. Figure 1 depicts independent file access approach.



©IJRASET 2015: All Rights are Reserved

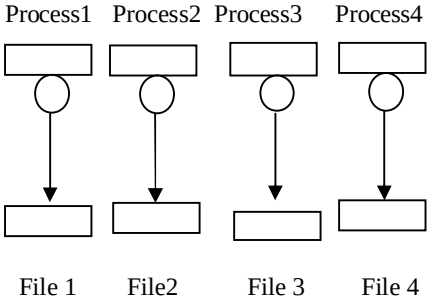



Figure 1. each process accessing Independent files

Advantages:
No communication or coordination necessary between processes. It avoids some file system quirks (e.g. false sharing). And has the potential to utilize the whole available bandwidth.
Disadvantages:
For large process counts, lots of files are created. Data often must be post-processed to recreate canonical dataset. Uncoordinated I/O from all processes may swamp I/O system

*B. All Processes Access One File*

This approach enables multiple processes to access parts of the single shared file. MPI-IO provides a framework for single file access from multiple processors. Figure2 depicts the shared file access by many processes.

*Advantages:*
Only one file (per time step etc.) to manage: fewer files overall. Data can be stored in canonical representation, avoiding post processing and has the potential to utilize the whole available bandwidth
*Disadvantage*:
Uncoordinated I/O from all processes may swamp I/O system.

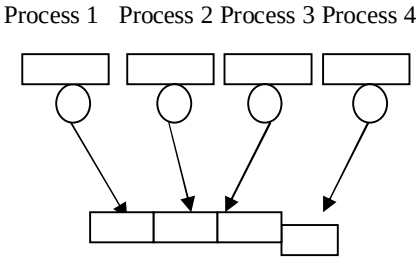



Figure 2. Multiple processes accessing shared file

Shared file access can be implemented in two ways as individual and collective file access, as well and blocking and non-blocking file access.

*1) Independent (non-collective) I/O:* Independent I/O operations specify only what a single process will do. Independent I/O calls do not pass on relationships between I/O on other processes.
*2) Collective I/O:* Collective I/O is coordinated access to storage by a group of processes. Collective I/O functions are called by all processes participating in I/O. It allows I/O layers to know more about access as a whole, more opportunities for optimization in lower software layers for better performance.

## V. EXISTING BENCHMARKING TOOLS

Commonly used benchmarking tools for evaluation of parallel file systems are: IOzone, Dbench, and Bonnie/Bonnie++

*A. IOzone*

IOzone is a benchmark for file read and writes speed. It supports Normal file I/O and POSIX asynchronous I/O i.e. Non-blocking I/O operations executed in parallel. I/O behavior of IOzone includes shared and per-process file access, MPI-IO and



©IJRASET 2015: All Rights are Reserved

POSIX I/O libraries, sequential and random I/O patterns. Throughput calculated in only kilobytes/second. Uses only command line to execute, there is no GUI which supports this tool.

### *B. Dbench*

Dbench takes only one parameter on the command line, which is the number of processes (client) to start. It runs with n parallel processes and delivers only one value as a result. The resulting value is an average throughput of the file system operations and measured in megabytes per second. It uses synchronous blocking I/O for directory and file operations.

### *C. Bonnie++*

Bonnie which is a C language based UNIX file system benchmark, reporting its throughput in Kilobytes per second. Bonnie wouldn't run tests on data sets larger than 2.1 Gigabyte. Bonnie++ is an enhancement of Bonnie. Bonnie++ is a migration from C to C++ and 64 bit software which works with lots of little files that tests bunch of things such as creation or deletion rates that Bonnie doesn't. It performs a number of simple tests of hard drive and file system performance, by sequential I/O and random seeks.

## VI. NEED FOR NEW BENCHMARKING TOOL

Present benchmarks support serial I/O to a large extent and parallel I/O operations in a limited way in HPC I/O performance evaluation. They do not provide options to measure the performance of metadata operations along with data I/O operations and also they do not provide any user friendly graphical interface to configure and submit the workloads. It would be better if the new tool supports custom workloads with different I/O scenarios of multiple clients and intuitive user interface. Therefore there is, need for a comprehensive tool to measure performance of meta-data and I/O operations in sequential and parallel modes with user specific workloads.

## VII. REFERENCE CLUSTER SYSTEM

The developed benchmarking tool has been used to evaluate performance of 128-node HPC cluster system installed with GlusterFS parallel file system. 20 Gbps Infiniband inter-connect is used for inter-process communication and storage access. Each node is based on Dual Intel Xeon processor with 3.13 GHz and 16 GB of RAM running Red Hat Enterprise Linux OS. Reference Cluster system configuration is tabulated in annexure I (TABLE I).

## VIII. I/O PERFORMANCE

The benchmarking tool developed determines the stability, integrity and performance of the specified parallel cluster file systems in the reference cluster. The benchmark generates a variety of file operations and measures their performance under different conditions.

### *A. Data I/O Operations*

The benchmark tests file I/O performance for the following operations with single/multiple clients using MPI-IO.
Write and Re-Write: Writing a new file and that of writing a file that already exist.
Random Write**:** Writing a file with accesses being made to random locations within the file is measured.
Strided Write**:** Writing a file with a strided access behavior is measured.
Read and Re-Read: Reading an existing file and that of reading a file that was recently read.
Random Read: Reading a file with accesses being made to random locations within the file is measured.
Strided Read: Reading a file with a strided access behavior is measured.
Backwards Read: Reading a file backwards.

### *B. Metadata Operations*

To stress the simultaneous metadata operations for files and directories following operations are tested.
Create, Open and Close file(s).
Stat: File or file system status is returned.
Lseek: sets the file offset of open file associated with the file descriptor to the argument offset.

Unlink: remove a symbolic link.



©IJRASET 2015: All Rights are Reserved

Truncate & Expand: Sets the size of a file.

Symlink: Symbolic links to reorder the file system hierarchy.

Directory stat: Returns the Status of working directory.
Readdir: Reads the content of directory.
Mkdir, Rmdir and Chdir operations.

## IX. BENCHMARK CONFIGURATION

### A. *Inputs / Configurable Parameters*

The benchmark takes following configuration parameters as inputs for running different performance tests:
Number of processes/clients
Number of blocks
Block size
Number of iterations
It also supports options for performing I/O operations-strided write, strided read, random write, and random read and backwards read with single shared file or multiple independent files. In addition shared file operation can be either in collective or in non-collective mode. The context diagram for I/O benchmark tool is shown in figure3.

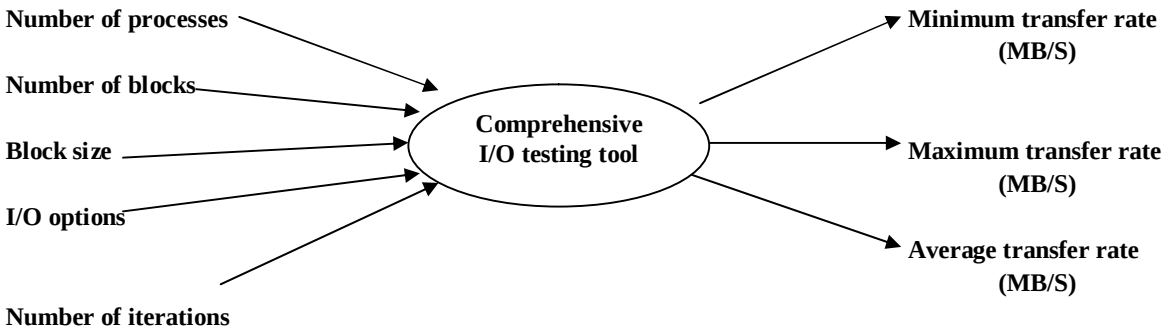



Figure 3. Context diagram for I/O benchmark tool

## X. RESULTS AND CONCLUSIONS

Various test cases for running the benchmark are created with different block sizes(512 KB, 1MB), file sizes (10MB, 100MB, 1GB, 10 GB) and number of clients(1, 2, 4, 8 and 16). The results obtained for these test cases are tabulated in annexure I. With average transfer rate, minimum and maximum transfer rates as outcome of our benchmarking tool, follwing observations are made based on the analysis of performance test results with average transfer rate in consideration. When the aggregate file size increases with multiple clients the I/O throughput also increases. As the number of clients increases from 2 to 16 the READ throughput also increases (Table III), while the WRITE throughput starts decreasing after 4 (Table II). Block size with 512KB gives better performance in both READ and WRITE (Table II & III). RE-READ and RE-WRITE (Table II & III) operations give better performance because of disk caching.

## REFERENCES

[1] LLNL's Parallel I/O Testing Tools and Techniques for ASC Parallel File Systems, *W. E. Loewe, R. M. Hedges, T. T. McLarty, and C. J. Morrone*,2004 IEEE Cluster Computing Conference, San Diego, CA, September 20-23, 2004.
[2] Investigation Of Leading HPC I/O Performance Using A Scientific –Application Derived Benchmark, *julian Borrill, Leonid Oliker, John Shalf, Hongzhang Shan*, Supercomputing, 2007 ACM/IEEE Conference.
[3] Using IOR to Analyze the I/O performance for HPC platforms, *Hongzhang Shan, John Shalf*, CUG Meeting 2007.
[4] Evaluation of A Performance Model of Lustre File System, Tiezhu Zhao, Verdi March, Shoubin Dong, Simon See, The Fifth Annual ChinaGrid Conference.
[5] Using MPI, 2nd Edition, William Gropp, EwingLusk and Anthony Skjellum.
[6] Using MPI-2: Advanced Features of the Message Passing Interface (Scientific and Engineering Computation) by William Gropp, Ewing L. Lusk and Rajeev Thakur (Nov 26, 1999).



©IJRASET 2015: All Rights are Reserved

[7] https://computing.llnl.gov/tutorials/mpi
[8] Bonnie++ [2001] http://www.coker.com.au/bonnie++/.
[9] dbench [2001]. http://freshmeat.net/projects/dbench.
[10] Iozone[2003]http://www.gnu.org/directory/sysadmin/Monitor/Iozone.html

ANNEXURE I

TABLE I. Reference HPC Cluster System Configuration

| NAME | PROCESSOR/NODE | INTERCONNECT | STORAGE | OPERATING SYSTEM | PARALLEL FILE SYSTEM |
|---|---|---|---|---|---|
| 128-noded HPC cluster | Dual Intel Xeon processor (Dual Quad Core) with 3.13 GHz and 16 GB of RAM | 20 Gbps Infiniband | 24 TB aggregated capacity (4 storage nodes each with 6 TB of disk capacity) | Red Hat Enterprise Linux | GlusterFS |

TABLE II. Performance measure for WRITE and RE-WRITE

| I/O operation | Transfer Rate for I/O operations in MB/second | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | *Block size* | *512 KB* | | | | *1 MB* | | | |
| | *Number of clients* / *File size/client* | *2* | *4* | *8* | *16* | *2* | *4* | *8* | *16* |
| WRITE | *1MB* | 30.2 | 25.2 | 100.2 | 61.2 | 71.2 | 82.5 | 80.5 | 60.5 |
| | *10 MB* | 142.5 | 126.7 | 417.5 | 390.5 | 243.8 | 280.5 | 221.6 | 187.2 |
| | *100MB* | 227.8 | 410.2 | 454.1 | 691.9 | 744.3 | 620.4 | 700.9 | 631.5 |
| | *1GB* | 570.7 | 600.4 | 664.0 | 686.2 | 630.2 | 409.4 | 648.9 | 630.5 |
| RE-WRITE | *1MB* | 30.6 | 25.2 | 101.7 | 62.2 | 72.4 | 83.4 | 80.6 | 72.8 |
| | *10MB* | 144.8 | 128.2 | 425.4 | 397.8 | 274.6 | 285.6 | 225.2 | 190.7 |
| | *100MB* | 230.5 | 418.2 | 460.7 | 703.2 | 758.8 | 632.4 | 710.2 | 643.6 |
| | *1GB* | 581.4 | 612.5 | 677.2 | 699.7 | 640.2 | 417.2 | 660.9 | 622.6 |

TABLE III. Performance measure for READ and RE-READ

| I/O operation | Transfer Rate for I/O operations in MB/second | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | *Block size* | *512 KB* | | | | *1 MB* | | | |
| | Number of clients / File size/client | *2* | *4* | *8* | *16* | *2* | *4* | *8* | *16* |
| READ | 1MB | 150.3 | 130.2 | 273.7 | 253.5 | 505.2 | 453.6 | 770.8 | 700.4 |
| | 10 MB | 540.8 | 548.9 | 1056.3 | 1022.1 | 2122.3 | 2033.4 | 2875.1 | 3033.6 |
| | 100MB | 1158.3 | 1125.2 | 2030.4 | 2263.7 | 4121.3 | 3400.2 | 6176.0 | 6050.1 |
| | 1GB | 1541.0 | 1663.5 | 3215.7 | 3204.3 | 6038.8 | 5950.7 | 8208.4 | 7953.5 |
| RE-READ | 1MB | 165.7 | 140.6 | 298.3 | 278.3 | 555.5 | 498.3 | 847.9 | 770.0 |
| | 10MB | 594.4 | 600.8 | 1161.6 | 1024.2 | 2300.2 | 2036.3 | 3162.5 | 3336.3 |
| | 100MB | 1169.6 | 1237.5 | 2233.1 | 2489.3 | 4533.1 | 3740.0 | 6793.6 | 6655.2 |
| | 1GB | 1695.1 | 1829.3 | 3536.5 | 3524.4 | 6624.4 | 6545.7 | 9028.2 | 8750.3 |

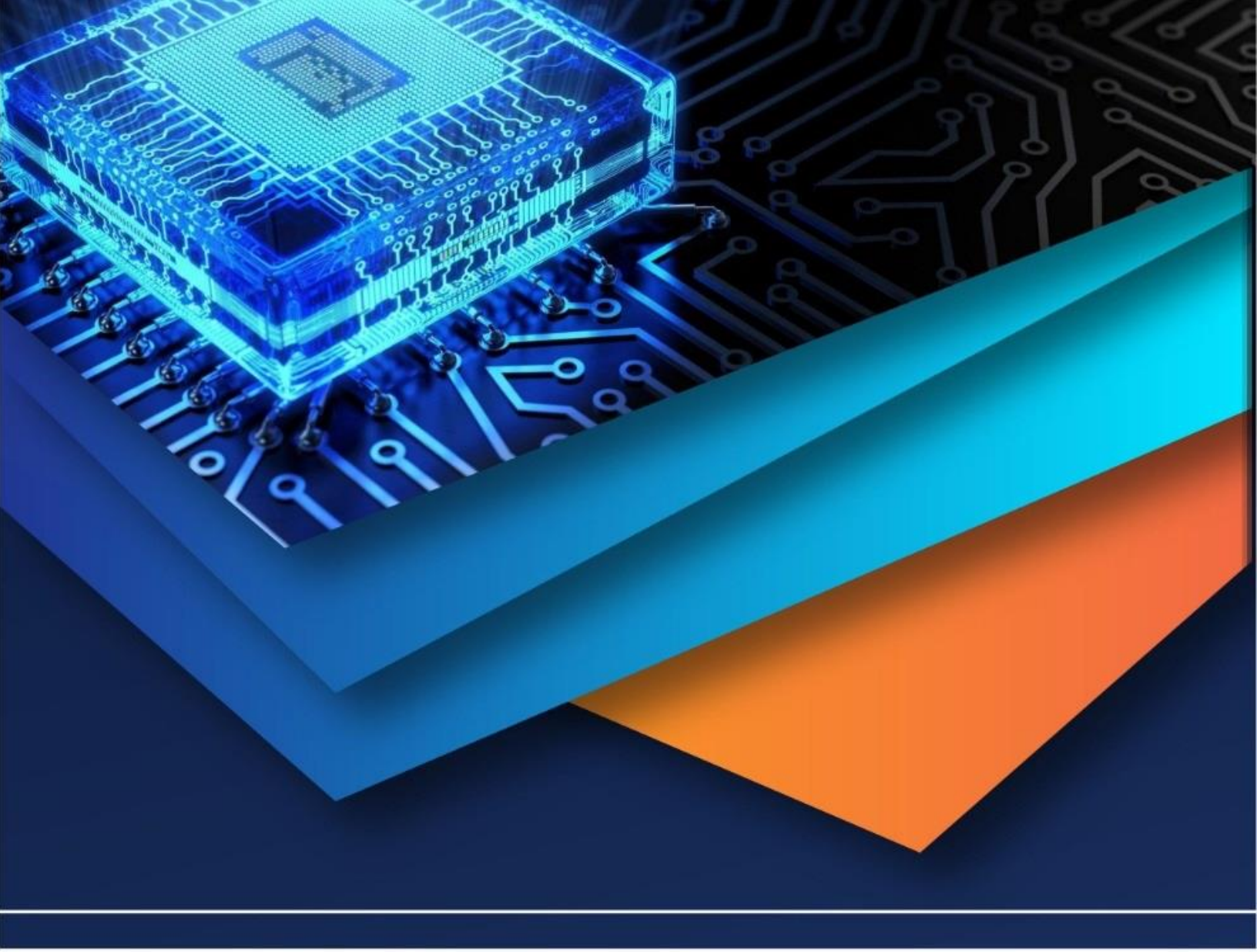
©IJRASET 2015: All Rights are Reserved

Volume 3 Issue V, May 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

TABLE IV. Performance measure for STRIDED WRITE and STRIDED READ

| I/O operation | Transfer Rate for I/O operations in MB/second | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | *Block size* | *512 KB* | | | | *1 MB* | | | |
| | *Number of clients* / *File size/client* | *2* | *4* | *8* | *16* | *2* | *4* | *8* | *16* |
| STRIDED WRITE | *1MB* | 29.9 | 23.4 | 99.5 | 60.8 | 69.9 | 82.3 | 79.9 | 59.9 |
| | *10 MB* | 142.1 | 120.3 | 410.7 | 388.9 | 242.9 | 279.9 | 220.7 | 185.9 |
| | *100MB* | 226.7 | 409.5 | 454.8 | 690.5 | 744.2 | 600.1 | 700.5 | 630.2 |
| | *1GB* | 569.9 | 599.5 | 660.1 | 686.5 | 630.4 | 400.9 | 640.6 | 629.9 |
| STRIDED READ | *1MB* | 149.2 | 129.9 | 272.9 | 25.9 | 505.7 | 452.3 | 769.9 | 699.5 |
| | *10MB* | 540.3 | 547.2 | 1055.9 | 1021.9 | 2121.9 | 2032.9 | 2874.0 | 3033.2 |
| | *100MB* | 1157.9 | 1147.5 | 2029.9 | 2260.9 | 4120.9 | 3399.9 | 6175.2 | 6049.9 |
| | *1GB* | 1540.6 | 1696.2 | 3214.9 | 3203.9 | 6037.9 | 5949.9 | 8208.2 | 7952.9 |

©IJRASET 2015: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)