IKAROS Parallel File System client utility

Extended Abstract†

E.Plakida
Informatics & Telecommunications,
University of Athens, Greece
e.plakida@di.uoa.gr

K.Triantaphyllou
Informatics & Telecommunications,
University of Athens, Greece
k.triantaphyllou@di.uoa.gr

C. Filippidis
Informatics & Telecommunications,
University of Athens, Greece
cfjs@outlook.com

Y. Cotronis
Informatics & Telecommunications,
University of Athens, Greece
cotronis@di.uoa.gr

ABSTRACT

I/O has become a bottleneck in application performance while the substantial amount of concurrency can cause a critical contention issue for I/O system. In this study we develop an IKAROS multi-threaded client utility in C. IKAROS is a dynamically coordinated I/O system being used for addressing some of the limitations that current parallel file systems and storage architectures are facing with large-scale systems. The fundamental idea is to coordinate I/O accesses according to the topology/profile of the infrastructure, the load metrics, and the I/O demands of each application. Our goal, in this study, is to compare the performance of the client implemented in C, against the existing Python version.

CCS CONCEPTS

• **Software and its engineering → File systems management**; Layered systems • **Networks** → Network range

KEYWORDS

Data management; storage systems; parallel file systems; high performance computing;

1 IKAROS Parallel File System

Large-scale scientific computations tend to stretch the limits of computational power, thus parallel computing is generally recognized as the only viable solution to high performance computing problems. I/O has become a bottleneck in application performance as processor speed skyrockets, leaving storage hardware and software struggling to keep up. Parallel file systems have been developed in order to allow applications to make optimum usage of the available processor parallelism. The most important factors affecting performance are the number of parallel processes participating in the transfers, the size of the individual transfers and of course the access patterns. Current storage architectures, have several performance limitations when used with large-scale systems, because [4]:

1.Bandwidth does not scale economically to large-scale systems.
2.I/O traffic on the high speed network can be affected by other unrelated jobs.
3.I/O traffic on each storage server can also be affected by other unrelated jobs.
IKAROS achieves better performance by creating, on the fly, a cluster of dedicated (4:1) or semi-dedicated (4:2) storage facility for each client, where the I/O traffic cannot be affected by other client requests. In this way we manage to improve I/O performance by 33% using only the 1/3 of the available hard disks.

In this study we develop a multi-threaded IKAROS client in C (the existing version is in Python) in order to trace potential trade-offs in performance while implementing the IKAROS logic. IKAROS client holds an extremely important role in IKAROS architecture and this study is crucial for developing further the IKAROS system.
We are developing the C client on top of the libcurl [5] module, in three different approaches.**A:** libcurl easy API (synchronous) + Fork system call, **B:** libcurl easy API (synchronous) + Threads system call, and **C:** libcurl Multi API. The Multi interface offers multiple simultaneous transfers in the same thread without making it complicated for the application. Our preliminary results using the libcurl easy interface show no significant differences, in performance, between the C and the Python version.

REFERENCES

[1] Christos Filippidis, Panayiotis Tsanakas,Yiannis Cotronis, IKAROS: a Scalable I/O Framework for High-Performance Computing Systems,The Journal of Systems& Software (2016),Volume 118, pp. 277-287, DOI:http://dx.doi.org/10.1016/j.jss.2016.05.027
[2] Filippidis C., Cotronis Y.,Markou C.,(2013) IKAROS: an HTTP-based distributed File System, for low consumption and low specification devices, Journal of Grid Computing springer.
[3] Filippidis, C., Cotronis, Y., & Markou, C. (1/2016). Using IKAROS as a data transfer and management utility within the KM3NeT computing model, Very Large Volume Neutrino Telscope workshop (VLVnT 2015).
[4] Dongarra, J., Beckman, P. et al. The International Exascale Software Roadmap," , Volume 25, Number 1, 2011, International Journal of High Performance Computer Applications, ISSN 1094-3420. Exascale Nearby Storage, Cray Position paper.
[5] https://curl.haxx.se/libcurl/