# FusionProv: Towards a Provenance-Aware Distributed Filesystem

Chen Shou*, Dongfang Zhao*, Tanu Malik†‡, Ioan Raicu*‡
*Department of Computer Science, Illinois Institute of Technology
†Computation Institute, The University of Chicago
‡Math and Computer Science Division, Argonne National Laboratory

***Abstract*—It has become increasingly important to capture and understand the origins and derivation of data (its provenance). A key issue in evaluating the feasibility of data provenance is its performance, overheads, and scalability. In this paper, we explore the feasibility of a management layer for parallel file systems, in which metadata includes both file operations and provenance metadata. We design and implement a provenance layer within a distributed file system —FusionFS, which implements a distributed file metadata management based on distributed hash tables. Our results show that FusionFS with its own storage layer for provenance capture is able to scale up to 1K nodes on BlueGene/P supercomputer.**

## I. Introduction

Scientific advancement and discovery critically depends upon being able to extract knowledge from extremely large data sets, produced either experimentally or computationally. In experimental fields such as high-energy physics datasets are expected to grow by six orders of magnitude [2]. In computational fields such as fusion science data will be output at 2 gigabytes/second per core or 2 petabytes/second of checkpoint data every 10 minutes [2]. This amounts to an unprecedented I/O rate of 3.5 terabytes/second. To extract knowledge from extremely large datasets in a scalable way, architectural changes to HPC systems are increasingly being proposed—changes that either reduce simulation output data [5, 6] or optimize the current flop to I/O imbalance [1, 3].

A primary architectural change is a change in the design of the storage layer, which is currently segregated from compute resources. Storage is increasingly being placed close to compute nodes in order to help manage large-scale I/O volume and data movement [3, 8, 10, 11], especially for efficient checkpointing at extreme scale [15]. This change in the storage layer has a significant resulting advantage—it enables simulation output data to be stored with the provenance metadata so that analysis can be easily verified, validated as well as retraced over time steps even after the simulation has finished.

While this architectural change is being deemed necessary to provide the much needed scalability advantage of concurrency and throughput, it cannot be achieved without providing an efficient storage layer for conducting metadata operations [9]. The centralized metadata repository in parallel file systems has shown to be inefficient at large scale for conducting metadata operations, growing for instance from tens of milliseconds on a single node (four-cores), to tens of seconds at 16K-core scales [9, 13]. Similarly, auditing and querying of provenance metadata in a centralized fashion has shown poor performance over distributed architectures [7].

In this paper, we explore the feasibility of a general metadata storage and management layer for parallel file systems, in which metadata includes both file operations and provenance metadata with the FusionFS [14] infrastructure. FusionFS provides a POSIX interface and conducts data I/O in a decentralized such that the resources at each node are fully exploited.

## II. Design and Implementation

Figure 1 illustrates how we integrate FusoinFS and ZHT to support distributed provenance capture at the file system level. Provenance is firstly generated in the FUSE layer in FusionFS, and then is cached in the local provenance buffer. And at a certain point (e.g. when the file is closed), the cached provenance will be persisted into ZHT. Users can do query on any node of the system using a ZHT client.

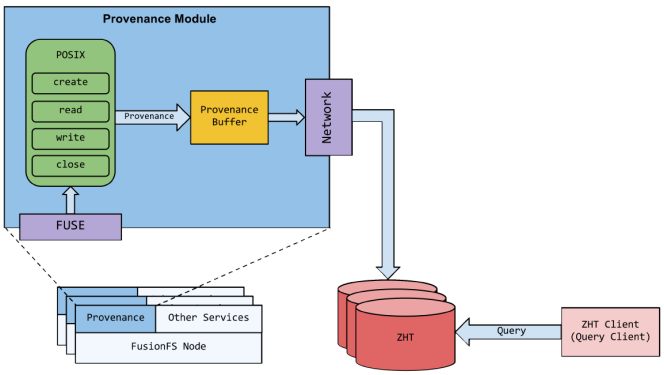


Fig. 1. FusionFS+ZHT architecture overview

Table I shows what is captured for the graph vertex in the distributed provenance store. Basically there are two different vertex types being tracked of: file and process. In other words, we are interested in which file(s) have been touched by which process(es). And we maintain a linked list for the tree topology in ZHT.

TABLE I
Attributes of Graph Vertexes in Distributed Provenance Capture

| Vertex Type | Attributes |
|---|---|
| File | [File path/name] [File version] [File size] |
| Process | [Process host] [Process name] |

We implement a light-weight command-line tool that end users can use to query the provenance, in the following syntax:

```
query vertex [filename] [file version]
  [ancestors -- descendants] [depth]
```

## III. Preliminary Results

We have deployed the distributed provenance-aware file system on 1K-node IBM BlueGene/P supercomputer Intrepid [4]. We also evaluated the system on a 32-node cluster, where each node has two Quad-Core AMD Opteron 2.3GHz processors with 8GB memory. All nodes are interconnected by 1Gbps Ethernet.

We have scaled the distributed provenance system up to 1K-node on IBM BlueGene/P. Figure 2 shows that the provenance overhead is relative small even on 1K nodes (14%). Similarly, we report the query time and overhead on the same workload at large scale (i.e. 1K nodes) in Figure 3, which shows that the overhead at 1K-nodes is about 18%.

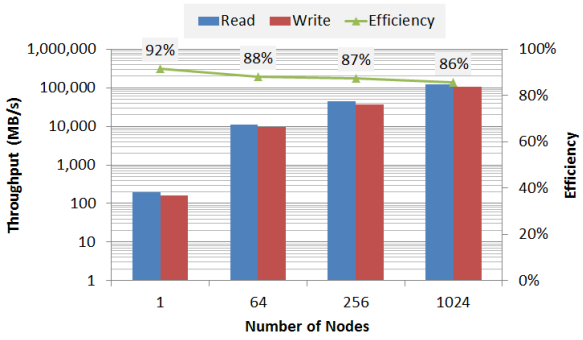


Fig. 2. Throughput on BlueGene/P

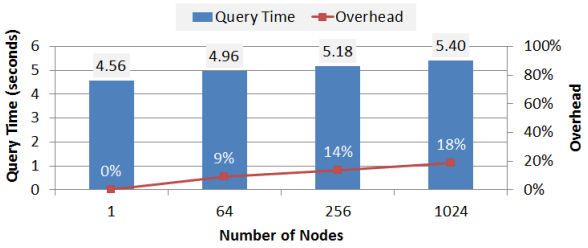


Fig. 3. Query Time on BlueGene/P

## IV. Conclusion and Future Work

FusionFS with its own storage layer for provenance capture is able to scale up to 1K nodes on BlueGene/P supercomputer. As for the future work, we plan to integrate the Swift parallel programming system [17] to deploy real scientific applications [12, 16] on FusionFS+SPADE and FusionProv, as well as continue to scale FusionFS/FusionProv towards petascale levels.

## References


[1] N. Ali, P. Carns, K. Iskra, D. Kimpe, S. Lang, R. Latham, R. Ross, L. Ward, and P. Sadayappan. Scalable i/o forwarding framework for high-performance computing systems. In *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, pages 1–10. IEEE, 2009.

[2] P. A. Freeman, D. L. Crawford, S. Kim, and J. L. Munoz. Cyberinfrastructure for science and engineering: Promises and challenges. *Proceedings of the IEEE*, 93(3):682–691, 2005.

[3] S. Ghemawat, H. Gobioff, and S.-T. Leung. The google file system. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, 2003.

[4] Intrepid. https://www.alcf.anl.gov/resource-guides/intrepid-file-systems.

[5] K.-L. Ma. In situ visualization at extreme scale: Challenges and opportunities. *Computer Graphics and Applications, IEEE*, 29(6):14–19, 2009.

[6] K.-L. Ma, C. Wang, H. Yu, and A. Tikhonova. In-situ processing and visualization for ultrascale simulations. In *Journal of Physics: Conference Series*, volume 78, page 012043. IOP Publishing, 2007.

[7] T. Malik, A. Gehani, D. Tariq, and F. Zaffar. Sketching distributed data provenance. In *Data Provenance and Data Management in eScience*, pages 85–107. 2013.

[8] I. Raicu, I. T. Foster, and P. Beckman. Making a case for distributed file systems at exascale. In *Proceedings of the third international workshop on Large-scale system and application performance*, LSAP '11, 2011.

[9] I. Raicu, Z. Zhang, M. Wilde, I. Foster, P. Beckman, K. Iskra, and B. Clifford. Toward loosely coupled programming on petascale systems. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, 2008.

[10] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, MSST '10, 2010.

[11] I. Stoica, R. Morris, D. Karger, M. F. Kasshoek, and Balakrishnan. Chord: Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications (ACM SIGCOMM)*, 2001.

[12] M. Wilde, I. Raicu, A. Espinosa, Z. Zhang, B. Clifford, M. Hategan, K. Iskra, P. Beckman, and I. Foster. Extreme-scale scripting: Opportunities for large task parallel applications on petascale computers. In *SCIDAC, Journal of Physics: Conference Series 180. DOI*, pages 10–1088, 2009.

[13] Z. Zhang, A. Espinosa, K. Iskra, I. Raicu, I. Foster, and M. Wilde. Design and evaluation of a collective i/o model for loosely- coupled petascale programming. IEEE MTAGS, 2008.

[14] D. Zhao and I. Raicu. Distributed File Systems for Exascale Computing. ACM/IEEE Supercomputing (Doctoral Showcase), 2012.

[15] D. Zhao, D. Zhang, K. Wang, and I. Raicu. Exploring reliability of exascale systems through simulations. In *Proceedings of the High Performance Computing Symposium*, HPC '13, 2013.

[16] Y. Zhao, X. Fei, I. Raicu, and S. Lu. Opportunities and challenges in running scientific workflows on the cloud. In *Proceedings of the 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, CYBERC '11, 2011.

[17] Y. Zhao, M. Hategan, B. Clifford, I. T. Foster, G. von Laszewski, V. Nefedova, I. Raicu, T. Stef-Praun, and M. Wilde. Swift: Fast, reliable, loosely coupled parallel computation. In *IEEE SCW*, pages 199–206, 2007.