A Unifying Approach to the Exploitation of File Semantics in Distributed File Systems

Philipp Hahn[*]

Carl von Ossietzky University of Oldenburg
Department of Computer Science
D-26111 Oldenburg, Germany
Phone: [49] (441) 798-2866
Fax: [49] (441) 798-2756
Email: pmhahn@acm.org

Keywords: Distributed File Systems, File Semantics, Customization, Data Consistency, Data Encryption, Data Compression, Data Availability, Data Security

1. Introduction

NFS and CIFS are probably among the currently mostly utilized distributed file systems. Clearly, there is a variety of distributed file systems that has been proposed in literature, but they are often quite specialized with respect to a particular networking environment, usage scenario, or type of files hosted by the distributed file system. As a consequence, they cannot – and in fact: are not – widely used. But specialization is not bad *per se*, since it allows for highly optimized file management. Universality, on the contrary, has the advantage of being generally adaptable, like the two distributed file systems cited before. But universality must pay the prize of being inefficient "on the average." NFS and CIFS, for example, are not suited for supporting disconnected working, like working on certain files while traveling or for "paranoid setups" where work on files must continue even in situations where some consistency criterion cannot be guaranteed anymore. Local caching or replication of whole filesystems might help in such cases. But there are other scenarios where the user does not want to pay the extra price of improved data availability.

2. Basic Idea

We propose an novel approach to the design of distributed file systems that 1) provides a general and universal file abstraction to the user but 2) allows specialized and highly optimized abstractions to be utilized on a per file or per file group basis in an easy manner if circumstances permit or the user dictates. Thus, without further actions from the user side,[1] files can always be handled in a general manner but without exploiting any file-specific properties. This "fall back" behavior allows for easy use and potentially wide dissemination, whereas the means to customize allows for efficiency and/or particular functionality to be offered without the need of migrating the files to other some "single-purpose" distributed file system.

In our framework, we plan to let the user easily exploit file-specific approaches well suited for supporting the following scenarios:

- Some files must be stored encrypted. Other files are stored unencrypted but these ones must be encrypted while transmitting them over an insecure channel.
- Some files must be accessible at any time, even when encountering certain network problems. For other files, access to them must guaranteed according to some more restrictive correctness criterion.
- Some files can be efficiently compressed. Others cannot.
- Some files are accessed many times and should be cached for performance reasons. Others are "read-only-once."
- Some files stay unchanged for quite a long time. Some files are even immutable. Other ones have only a very short live time.
- Some files are accessed concurrently by different users, others not.
- Some files are sequentially read from start to end, others are only partially read or using a particular access pattern.
- Some files contain an uninterpreted byte stream. Other files contain structured data.
- Some files should be versioned. For others, no versioning is required.

A already indicted, todays distributed file systems often simply ignore these dimensions and handle all files the same.

We a currently working on the conceptional design and the implementation of the sketched framework that supports the customization along these dimensions. As part of this work, we investigate in detail, how certain knowledge about file formats and file semantics can be exploited adequately. Furthermore, we identify what building blocks are needed and how must these blocks be combined in order to satisfy the individual user file constraints, for example with respect to data encryption, data security, and data availability?

---

[*] PhD student. Supervised by Oliver Theel, Carl von Ossietzky University of Oldenburg, Germany

[1] A user can also be the distributed file system administrator.