My Permanent Address: Finding A File After It Has Moved

Dan Muntz, Lisa Liu
Internet Systems and Storage Laboratory
HP Laboratories Palo Alto
HPL-2001-175
July 12th , 2001*

E-mail: dmuntz@hpl.hp.com, lliu@cup.hp.com

file systems, migration, distributed

The DiFFS file service, an ongoing project at HP Labs, supports migration of file system objects (files and directories) at the granularity of individual objects. The original design to support *object migration* relies on *forwarding pointers* stored at the object's previous location to determine the object's new location. There are various difficulties with implementing this scheme, and thus other mechanisms are being considered. One approach to solving this problem is to consider how similar situations are handled in other contexts. When students go off to college, they fill out many forms over the subsequent years asking for their home address. This *permanent address* can be used to communicate with the student during these years when the student is likely to change locations several times. A new mechanism for locating migrated objects is based on this model-objects are assigned permanent addresses. When an object cannot be found at its expected location, i.e., the object has migrated, a message can be sent to the object's permanent address tc retrieve its current location. This permanent address model decouples the forwarding information from the site of the object's previous location, providing greater flexibility for this functionality. It also solves a problem with the proposed method of dealing with volatile file handles in the NFSv4 specification.

* Internal Accession Date Only
Approved for External Publication

My Permanent Address: Finding A File After It Has Moved

Dan Muntz, Lisa Liu

Hewlett-Packard Labs

1501 Page Mill Rd, Palo Alto, CA 94304, USA

dmuntz@hpl.hp.com, lliu@cup.hp.com

1. Abstract

The DiFFS file service, an ongoing project at HP Labs, supports migration of file system objects (files and directories) at the granularity of individual objects. The original design to support *object migration* relies on *forwarding pointers* stored at the object's previous location to determine the object's new location. There are various difficulties with implementing this scheme, and thus other mechanisms are being considered. One approach to solving this problem is to consider how similar situations are handled in other contexts.

When students go off to college, they fill out many forms over the subsequent years asking for their home address. This *permanent address* can be used to communicate with the student during these years when the student is likely to change locations several times. A new mechanism for locating migrated objects is based on this model—objects are assigned permanent addresses. When an object cannot be found at its expected location, i.e., the object has migrated, a message can be sent to the object's permanent address to retrieve its current location. This permanent address model decouples the forwarding information from the site of the object's previous location, providing greater flexibility for this functionality. It also solves a problem with the proposed method of dealing with volatile file handles in the NFSv4 specification.

2. Introduction

Permanent addressing provides an alternative to "forwarding pointers" used to track object migration in the current DiFFS architecture. It also provides a solution for handling volatile file handles in NFSv4.

In this mechanism, each object is assigned a *permanent address*. The permanent address is the location of a server or *service* that can provide the current location of a migrated object. The inspiration for this system is the notion of a student's permanent address, i.e., their pre-college home address.

In NFSv4, when a volatile file handle expires, having the fully qualified path name at the client (as suggested in the NFSv4 protocol specification [1]) is not sufficient to obtain a new file handle for the object. E.g., if the object is `/a/b/c/d`, 'd' has a volatile file handle, and the directory 'b' is renamed to 'x', if a client discovers its file handle for 'd' has expired, a full pathname lookup won't help.

In DiFFS there is a similar problem. When migration occurs, the file handle for the object (usually) changes. This is currently solved in DiFFS in the following manner.

When a volatile file handle is *expired* the server maintains a mapping from the old file handle to the new file handle for some time, T. During this time, the old file handle may not be reused, and any client RPC containing the old file handle gets a response indicating the old file handle has expired along with the new file handle. One problem: clients "holding" (e.g., caching) volatile file handles must *revalidate* them within each time period T. Successful client requests using a volatile file handle are an implicit revalidation of any volatile file handles in the request.

Permanent addressing requires no forwarding pointers, and thus has none of the overhead associated with maintaining and discarding forwarding information [2].

3. How it works

Permanent addressing is described in the context of the DiFFS file service, currently being developed at HP Labs, but may also apply to other distributed file systems where object migration may occur (e.g., in NFSv4 or other future versions of NFS). To support permanent addresses, the DiFFS architecture is augmented with a new class of server (or service) called a *permanent address server*, or PAS. This service could reside on a DiFFS partition server. A PAS should be a highly-available entity with a fixed "location" (i.e., whatever addressing mechanism is used, the address for a PAS should not change). When file system objects are created, they are assigned to a particular PAS. This assignment is merely an association between the object and the address of its PAS (e.g., the address of the PAS could be stored in the object's directory entry). No communication with the PAS is necessary at this point.

The PAS is only involved when files are migrated and when clients attempt to use file handles generated before migration occurred. The PAS, as a service, may be a relatively lightly used resource, which may reside on the same hardware as other HA services that may be introduced for DiFFS (undisclosed).

3.1 From the client's perspective

An object's permanent address is part of the object's file handle, generated by file servers (an alternative implementation could supply the permanent address to clients without making it part of the file handle). In normal operation, the PAS and the permanent address are not used. However, when a client attempts to use a file handle to access a file that has migrated, the client will receive an error. At this point, the client can contact the PAS for the object, to determine its new location, and proceed with the operation.

3.2 From the perspective of an agent triggering object migration

When object migration takes place, a message is sent to the PAS for the object, indicating the object's new location. The PAS stores the new location(s) of migrated objects. The PAS may keep this information for an arbitrarily long time, or may have some heuristic to determine when information may be discarded.

3.3 Permanent addressing with file handle revalidation

One way to garbage collect old forwarding information is to use the existing file handle revalidation mechanism. If the system supports file handle revalidation, there is a weak upper-bound on the time that new address information must be maintained at the PAS to ensure that all outstanding file handles remain valid. Thus, after the time allowed for revalidation has passed for a given object, that object's new address information may be deleted from the PAS (garbage collected). This allows unnecessary forwarding information to be garbage collected at the PAS, potentially decreasing storage utilization and lookup times. Whether the (large) overhead associated with file handle revalidation is acceptable for this garbage collection is an issue for future work.

4. Conclusions

Permanent addressing provides a solution to revalidating expired volatile file handles in NFSv4. The proposed solution for this problem in the NFSv4 specification is not correct.

For the DiFFS architecture, permanent addressing provides an alternative to forwarding pointers and file handle revalidation. However, if revalidation mechanisms are present, they can be used in conjunction with permanent addressing to garbage collect new address information that is no longer needed.

Permanent addressing decouples new address information from the old object location. This allows new address information to be provided as a service that may or may not reside at the original object location. A separate service could be provided on

highly-available hardware, potentially allowing other portions of the system to reside on hardware that is "less reliable."

5. References

1. Reiser, H., *ReiserFS*, 2001. http://www.namesys.com/res_whol.shtml.
2. Mahalingam, M., *et al.*, *Data Migration in a Distributed File Service*, . 2001, Hewlett-Packard Labs, Palo Alto, CA