Peer-to-Peer Cooperative Backup, December 16th, 2004

MoSAIC, LAAS-CNRS

Introduction

This document summarizes a technical discussion held by part of the MoSAIC team at Toulouse, on December 16th, 2004. Past and current research work in the area of peer-to-peer content storage, peer-to-peer content location as well as applications of these techniques to peer-to-peer cooperative backup were presented and discussed. Slides by Ludovic Courtès are available at *http://www.laas.fr/mosaic/* and related discussions are summarized here.

1. Discussions

The presentation of existing peer-to-peer file sharing and backup systems yielded the following discussions.

1.1. On the Robustness of Content Location Algorithms

Their exists a number of efficient distributed content location algorithms for *overlay networks* [12,5,1] some of which build upon network topologies and associated routing algorithms found in high-performance computing. Since these algorithms rely on peers being well-behaved, they look vulnerable to the participation of *malicious peers* not complying with the algorithm assumptions.

On the other hand, non-deterministic content location algorithms have been designed to suit the needs of anonymous content publishing networks such as Freenet and GNUnet [7]. These protocols have been designed with *censorship resilience* in mind and can tolerate malicious nodes. However, it is unclear how well such schemes perform on large-scale networks.

Fortunately, solutions have been envisioned to make the former type of algorithm more robust, by increasing the number of peers known to each peer and by randomizing the routing algorithm when possible, therefore making it non-deterministic.

1.2. Discovering Similarities Amongst Peers' Data

Pastiche [8] is a peer-to-peer backup system that implements a so-called *lighthouse sweep* mechanism aimed at helping backup peers discover peers whose personal data are *similar* to the data it is willing to back up. The goal is to limit the amount of data being backed up: data shared among several peers is backed up only once.

While Pastiche's authors claim that their system computes an *abstract* (« a small, random subset of their [data chunks'] signatures ») of peers' data in a way that is similar to a hash, it was highlighted in the discussion that hash functions are designed in a way to emphasize differences, not similarities. Indeed, while the authors do not describe the exact algorithm they use for finding redundancy, they refer the reader to an article by A. Z. Broder, *On the resemblance and containment of documents*, 1998.

1.3. Trade-off Between Superfluous Data Redundancy Elimination and Data Locality

As discussed in section 1.2, backup systems try to reduce data chunks redundancy in the overall peer network by preferably grouping peers holding similar data rather than peers with completely different data sets. At the same time, peer-to-peer backup systems try to favor *peers' physical locality* (i.e. in terms of the number of IP hops needed to reach each other) so as to improve the latency when sending/restoring data to/from one's peer. Furthermore, in Pastiche [8], backup peers (or *buddies*) directly communicate (over IP) with each other and only use the overlay network as a means of discovering peers.

This scheme yields to trade-off between *communications efficiency* (good latency and direct communication channels) and *storage efficiency* (elimination of superfluous data redundancy). As noted during the meeting, Pastiche's protocol for discovering peers with similar data may be inefficient for several reasons:

- data similarity in only checked upon peers' introduction into the network;
- more importantly, computing abstracts of *all* the data of a node may not be as efficient as using abstracts of well-chosen subsets of each nodes' contents. A possibility would be to split data into various *themes* where a node (its user) would hold data corresponding to different themes. It would then be more efficient to use a buddy-list for each theme, i.e. the similarities would be maximized.

Some experiments, however, tend to show [8] that this mechanism effectively discovers peers with a high data similarity. However, the authors are considering whole system installations, including the operating system, libraries and executables, etc. This type of data are obviously very similar from one system to another, while the data produced by the user – which is typically the type of data most people value the most – may vary significantly.

Another issue with the peer selection scheme of Pastiche is that it may not be very efficient in terms of the *overall network superfluous redundancy*: a peer may end up backing up a data chunk on its set of backup peers (called *buddies* in Pastiche) even though this chunk has already been backed up by other peers (but not by his buddies). *PeerStore* [10] solves this problem by using a distributed hash table (DHT) in order to locate replicas of data chunks: before backing up a data chunk, each peer looks for other replicas of that block (*wherever they are physically located*) amongst participating nodes and only keeps it if it isn't stored already.

2. Future Directions

This section describes topics that have only been briefly discussed and will require further research.

2.1. Fairness Enforcement Technique in Cooperative Backup

Most peer-to-peer backup systems propose on the one hand a dedicated mechanism to enforce fair contribution (e.g. [9,10]) and on the other hand another mechanism to deal with peers' unavailability (being malicious or not). It was noted that a global *economic model* (such as those found in most modern peer-to-peer networks [3,6]) involved in all interactions amongst peers and taking into account the whole behavior of peers (not only data storage *or* availability) might achieve better overall efficiency.

2.2. Archival and Versioning Systems

Archival and versioning systems were mentioned as a research area particularly relevant to MoSAIC. By *archival* or *versioning systems*, we mean systems that are able to keep track of changes that are made to individual files, file metadata, or directories. This includes regular software configuration management systems (SCM) such as CVS, GNU Arch, etc., as well as file systems archival systems like Plan 9's Venti [11] and versioning file systems [4,2,13]. Such systems offer interesting properties:

- They allow to keep track of modifications and access a specific version of a file. MoSAIC could benefit from such a facility since it may not be able to back up a whole file before it is modified and users would prefer having done a complete backup copy of an older version of a file rather than having only copied piece of successive versions of a file. It is worth noting that simple file locking mechanisms could be used to implement this feature instead of a complete versioning file system , although this would not be as efficient and practical.
- Versioning systems efficiently use storage resources by storing only differences between versions.
- File systems benefit have an intimate knowledge of how files are represented in the underlying block store; integrating versioning capabilities at the file system level allows to benefit from this fine-grained vision of files' contents and to coalesce the contents (blocks) of different versions of a file.
- The file system knows each time a new version is created, although it cannot distinguish which versions are valuable to the user without the user's (or application's) help. It could guarantee that a version being used (e.g. being backed up) will not vanish.

Moreover, as long as storage is available, a user might wish to keep copies of different versions of his important files on his backup peers. From these observations, one can envision the peer-to-peer backup system itself as something that is layered *on top* of a versioning file system.

On the other hand, relying on a versioning file system implies modifying the end-users file system, this could not be an option in wide deployment scenarios. As a consequence, we should probably design MoSAIC in a way that allows it to adapt to a versioning file system when available.

References

[1] ANTONY ROWSTRON, PETER DRUSCHEL – Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems – *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, Heidelberg, Germany, November, 2001, pp. 329-350.

[2] B. CORNELL, P. DINDA, F. BUSTAMANTE – Wayback: A User-level Versioning File System for Linux – *Proceedings of the USENIX Annual Technical Conference, FREENIX Track*, 2004, pp. 19-28.

[3] CHRISTIAN GROTHOFF – An Excess-Based Economic Model for Resource Allocation in Peer-to-Peer Networks – Wirtschaftsinformatik, 3-2003June, 2003.

[4] DOUGLAS S. SANTRY, MICHAEL J, FEELEY, NORMAN C. HUTCHINSON, ALISTAIR C. VEITCH, ROSS W. CARTON, JACOB OFIR – Deciding when to forget in the Elephant file system – *Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP'99)*, December, 1999, pp. 110-123.

[5] ION STOICA, ROBERT MORRIS, DAVID LIBEN-NOWELL, DAVID R. KARGER, M. FRANS KAASHOEK, FRANK DABEK, HARI BALAKRISHNAN – Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications – IEEE / ACM Transactions on Networking, 2005?.

[6] KEVIN LAI, MICHAL FELDMAN, JOHN CHUANG, ION STOICA – Incentives for Cooperation in Peer-to-Peer Networks – *Workshop on Economics of Peer-to-Peer Systems.*

[7] KRISTA BENNETT, CHRISTIAN GROTHOFF – GAP - practical anonymous networking – *Proceedings of the 3rd Workshop on Privacy Enhancing Technologies (PET 2003)*, Dresden, Germany, March, 2003.

[8] L. P. COX, B. D. NOBLE – Pastiche: Making backup cheap and easy – *Fifth USENIX Symposium on Operating Systems Design and Implementation*, Boston, MA, USA, December, 2002.

[9] LANDON P. COX, BRIAN D. NOBLE – Samsara: Honor Among Thieves in Peer-to-Peer Storage – *19th ACM Symposium on Operating Systems Principles*, Bolton Landing, NY, USA, October, 2003.

[10] MARTIN LANDERS, HAN ZHANG, KIAN-LEE TAN – PeerStore: Better Performance by Relaxing in Peer-to-Peer Backup – *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, Zurich, Switzerland, August, 2004.

[11] SEAN QUINLAN, SEAN DORWARD – Venti: A new approach to archival storage – *First USENIX conference on File and Storage Technologies*, Monterey,CA, 2002.

[12] SYLVIA RATNASAMY, PAUL FRANCIS, MARK HANDLEY, RICHARD KARP, SCOTT SHENKER – A Scalable Content-Addressable Network – *Proceedings of ACM SIGCOMM 2001*, UC San Diego, California, USA, August, 2001.

[13] Z.N.J. PETERSON, R.C. BURNS – Ext3cow: The Design, Implementation, and Analysis of Metadata for a Time-Shifting File System – HSSL-2003-03, Hopkins Storage Systems Lab, Department of Computer Science, Johns Hopkins University, USA, 2003.