

Distributed Storage

Evan Danaher

October 9, 2003

Abstract

In many computer labs, there are large numbers of computers with unused drive space. There may also be relatively large quantities of data to be backed up. The goal of this project will be to develop a system for storing data distributed over many computers, with enough redundancy so that data can still be recovered if several of the machines are unavailable (due to inevitable hardware failure). It could be enhanced to provide encryption or some other means of privacy for this distributed data.

1 Title

Distributed Storage

2 Problem Statement

This project will distribute data over computers such that the data can be recovered even if some machines are no longer available.

3 Purpose

In order to efficiently use existing computer hardware and safely store data, a system must be devised for storing that data. This project aims to provide such a system.

4 Scope

The basic project would be to choose a method for redundantly storing the data, implement it, and implement a method for distributing and retrieving the data over multiple computers.

If there is extra time, the project could be extended to include privacy features, distributed computation of the redundancy, or other useful extras.

5 Background

An article, *A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems*, describes the use of a variant of Raid-Solomon for use in certain applications. The idea is that Raid-Solomon can be used relatively simply when you know which data is lost; normally it is used in devices such as CD-ROMS, where it is impossible to know which data is correct and which was mangled. But in "RAID-like Systems," it is known which device failed, so

data recovery is simpler. In particular, I can use this algorithm for recovering data from multiple computers when some computers are dead.

6 Procedure

1. Learn about existing methods for redundant storage. One possibility for this is Reed-Solomon encoding, a common method. Then this should be implemented either using new code, or using an existing library.
2. Once files can be stored redundantly, split the file up among smaller files, such that the original data can be reconstructed for various subsets of the files.
3. Write code to distribute the files onto and retrieve them from different computers.
4. If time is left over, add additional features,

7 Expected Results

Ideally, this will result in a working system that could be applied to various labs, possibly also providing a basis for future improvements.