UPGRAID

Usage-based striPe replicatinG RAID

Joseph Naps, Ellen Wagner

August 10, 2007
Project Overview

UPGRAID Partition

RAID Partition
Project Overview
Project Overview
Project Overview
Project Overview
What We Learned
What We Learned

- Kernel Compilation
What We Learned

- Kernel Compilation
- Virtual Machines
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
- RAID
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
- RAID
- Reading poorly documented code
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
- RAID
- Reading poorly documented code
- Properly documenting code
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
- RAID
- Reading poorly documented code
- Properly documenting code
- Working with low-level C code
What We Learned

- Kernel Compilation
- Virtual Machines
- Modules
- RAID
- Reading poorly documented code
- Properly documenting code
- Working with low-level C code
- Block I/Os in Linux
Approach
Approach

- Read Replication
Approach

- Read Replication
- Write Replication
Approach

- Read Replication
- Write Replication
- Read Indirection
Approach

- Read Replication
- Write Replication
- Read Indirection
- Write Indirection
UPGRAID determines if the stripe is eligible for replication. If the stripe is eligible, a read request to the entire stripe is generated. Once that read request completes, a write is generated and put into a queue to await being sent to an UPGRAID partition.
1 UPGRAID determines if the stripe is eligible for replication.
UPGRAID determines if the stripe is eligible for replication.

If the stripe is eligible, a read request to the entire stripe is generated.
1. UPGRAID determines if the stripe is eligible for replication.
2. If the stripe is eligible, a read request to the entire stripe is generated.
3. Once that read request completes, a write is generated and put into a queue to await being sent to an UPGRAID partition.
Approach - Read Replication
Approach - Read Replication
UPGRAID determines if the stripe is eligible for replication. If the stripe is eligible, a read request to the entire stripe is generated. At this point, there are sixteen pages (in the page of a sixty-four KB stripe) with the data from the original stripe. The data from the original write must now be overlaid on top of the data read from the stripe to preserve the modifications from the write. The modified write is sent to a queue to await submission to the proper UPGRAID partition.
UPGRAID determines if the stripe is eligible for replication.
UPGRAID determines if the stripe is eligible for replication.

If the stripe is eligible a read request to the entire stripe is generated.
1. UPGRAID determines if the stripe is eligible for replication.
2. If the stripe is eligible a read request to the entire stripe is generated.
3. At this point there are sixteen pages (in the page of a sixty-four KB stripe) with the data from the original stripe.
UPGRAID determines if the stripe is eligible for replication.

If the stripe is eligible a read request to the entire stripe is generated.

At this point there are sixteen pages (in the page of a sixty-four KB stripe) with the data from the original stripe.

The data from the original write must now be overlaid on top of the data read from the stripe to preserve the modifications from the write.
1. UPGRAID determines if the stripe is eligible for replication.
2. If the stripe is eligible a read request to the entire stripe is generated.
3. At this point there are sixteen pages (in the page of a sixty-four KB stripe) with the data from the original stripe.
4. The data from the original write must now be overlaid on top of the data read from the stripe to preserve the modifications from the write.
5. The modified write is sent to a queue to await submission to the proper UPGRAID partition.
Approach - Write Replication
Approach - Write Replication
Approach - Write Replication
Approach - Write Replication
UPGRAID determines if the request should be sent to the RAID5 partition or UPGRAID partition by looking at the head position of each drive. This drive that has the smallest distance to move is chosen to fulfill the request. The request is then sent to the appropriate disk and the application proceeds upon completion of that read request.
UPGRAID determines if the request should be sent to the RAID5 partition or UPGRAID partition by looking at the head position of each drive. This drive that has the smallest distance to move is chosen to fulfill the request.
UPGRAID determines if the request should be sent to the RAID5 partition or UPGRAID partition by looking at the head position of each drive. This drive that has the smallest distance to move is chosen to fulfill the request.

The request is then sent to the appropriate disk and the application proceeds upon completion of that read request.
Approach - Write Indirection

1. The write request to the RAID5 partition is cloned.
2. This cloned request gets sent to the appropriate location on the UPGRAID partition at the same offset into the stripe as the original write, thereby preserving the mirroring property between the two stripes.
The write request to the RAID5 partition is cloned.
1. The write request to the RAID5 partition is cloned.
2. This cloned request gets sent to the appropriate location on the UPGRAID partition at the same offset into the stripe as the original write, thereby preserving the mirroring property between the two stripes.
Testing Tools

- Integrity Checker
Testing Tools

- Integrity Checker
- Workload Profiler
Integrity Checker

Automated user level application to test reads and writes to specific blocks.

Uses:
- Check and see if data was written to the correct block.
- Make sure that modules are performing correctly.
Integrity Checker

- Automated user level application to test reads and writes to specific blocks.
Integrity Checker

- Automated user level application to test reads and writes to specific blocks.
- Uses:
Integrity Checker

- Automated user level application to test reads and writes to specific blocks.
- Uses:
  - Check and see if data was written to the correct block.
Integrity Checker

- Automated user level application to test reads and writes to specific blocks.
- Uses:
  - Check and see if data was written to the correct block.
  - Make sure that modules are performing correctly.
Integrity Checker

UPGRAID
Joseph Naps,
Ellen Wagner

Proceeds through three testing phases:
Write Phase
Generates a write workload across the entire disk space.
Read Phase
Generates a random read workload across the disk space.
Read and Compare Phase
Reads back in the original write workload and compares the data to ensure there was no data corruption.
Integrity Checker

- Proceeds through three testing phases:
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
    - Generates a write workload across the entire disk space.
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
    - Generates a write workload across the entire disk space.
  - Read Phase
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
    - Generates a write workload across the entire disk space.
  - Read Phase
    - Generates a random read workload across the disk space.
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
    - Generates a write workload across the entire disk space.
  - Read Phase
    - Generates a random read workload across the disk space.
  - Read and Compare Phase
Integrity Checker

- Proceeds through three testing phases:
  - Write Phase
    - Generates a write workload across the entire disk space.
  - Read Phase
    - Generates a random read workload across the disk space.
  - Read and Compare Phase
    - Reads back in the original write workload and compares the data to ensure there was no data corruption.
Workload Profiler

Input Variables
- percent sequential
- fraction writes
- I/O request rate
- average I/O size
- maximum I/O size
- duration of experiment
- seed for the random number generator

Output Variables
- actual duration of experiment
- average I/O time
- standard deviation
- throughput
Generates a workload according to user specifications to test ABLE modules
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
Generates a workload according to user specifications to test ABLE modules

- **Input Variables**
  - percent sequential

- **Output Variables**
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Generates a workload according to user specifications to test ABLE modules

Input Variables
- percent sequential
- fraction writes
- I/O request rate
- average I/O size
- maximum I/O size

Output Variables
- actual duration of experiment
- average I/O time
- standard deviation
- throughput
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment

- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator
- Output Variables
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator
- Output Variables
  - actual duration of experiment
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules
- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator
- Output Variables
  - actual duration of experiment
  - average I/O time
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules

- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator

- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
Workload Profiler

- Generates a workload according to user specifications to test ABLE modules

- Input Variables
  - percent sequential
  - fraction writes
  - I/O request rate
  - average I/O size
  - maximum I/O size
  - duration of experiment
  - seed for the random number generator

- Output Variables
  - actual duration of experiment
  - average I/O time
  - standard deviation
  - throughput
Future Work
Future Work

- Read Heuristic
Future Work

- Read Heuristic
- Testing and debugging of replication, indirection, and popularity code
Future Work

- Read Heuristic
- Testing and debugging of replication, indirection, and popularity code
- Reconstruction
Future Work - Read Heuristic

A similar task is done in the RAID1 code. We have looked into the code and think that it can be ported to UPGRAID with a few modifications.
A similar task is done in the RAID1 code.
A similar task is done in the RAID1 code. We have looked into the code and think that it can be ported to UPGRAID with a few modifications.
Future Work - Testing and Debugging

Currently using autorwbench for the purpose of testing. Once the system is more stable with autorwbench, UPGRAID can be deployed on a file system.
Future Work - Testing and Debugging

- Currently using autorwbench for the purpose of testing UPGRAID
Future Work - Testing and Debugging

- Currently using autorwbench for the purpose of testing UPGRAID
- Once the system is more stable with autorwbench, UPGRAID can be deployed on a file system.
Future Work - Reconstruction

Two main approaches exist:

- Disk-Oriented Reconstruction (DOR)
- Popularity-based Reconstruction (PRO)

An entirely new approach could be developed for UPGRAID.
Future Work - Reconstruction

- Not considered in detail yet
Future Work - Reconstruction

- Not considered in detail yet
- Two main approaches exist
Future Work - Reconstruction

- Not considered in detail yet
- Two main approaches exist
  - Disk-Oriented Reconstruction (DOR)
Future Work - Reconstruction

- Not considered in detail yet
- Two main approaches exist
  - Disk-Oriented Reconstruction (DOR)
  - Popularity-based Reconstruction (PRO)
Future Work - Reconstruction

- Not considered in detail yet
- Two main approaches exist
  - Disk-Oriented Reconstruction (DOR)
  - Popularity-based Reconstruction (PRO)
- An entirely new approach could be developed for UPGRAID
Future Work - Reconstruction via DOR

DOR works by generating a thread for each disk that is responsible for fulfilling requests to that disk for the purpose of rebuilding the data of the failed disk. There is also a master thread that is responsible for coordinating the actions of the disk threads. It is possible that UPGRAID could work directly below the master thread and indirectly rebuild requests for replicated blocks to the replicas stored on UPGRAID partitions.
Future Work - Reconstruction via DOR

- DOR works by generating a thread for each disk that is responsible for fulfilling requests to that disk for the purpose of rebuilding the data of the failed disk.
Future Work - Reconstruction via DOR

- DOR works by generating a thread for each disk that is responsible for fulfilling requests to that disk for the purpose of rebuilding the data of the failed disk.
- There is also a master thread that is responsible for coordinating the actions of the disk threads.
Future Work - Reconstruction via DOR

- DOR works by generating a thread for each disk that is responsible for fulfilling requests to that disk for the purpose of rebuilding the data of the failed disk.
- There is also a master thread that is responsible for coordinating the actions of the disk threads.
- It is possible that UPGRAID could work directly below the master thread and indirect rebuild requests for replicated blocks to the replicas stored on UPGRAID partitions.
Future Work - Reconstruction via PRO

PRO works by dividing the failed disk into “hot zones” and then rebuilding the zones based on the current access rate to that zone. UPGRAID could sit above this process and use replicated stripes to improve this process. This approach would likely be more complex but its popularity based operation seems like a good fit with UPGRAID. It may be good if we defined these “hot zones” to align with the stripes of the RAID5 disk. This would make reconstruction using the replicated stripes easier.
Future Work - Reconstruction via PRO

- PRO works by dividing the failed disk into “hot zones” and then rebuilding the zones based on the current access rate to that zone.
Future Work - Reconstruction via PRO

- PRO works by dividing the failed disk into “hot zones” and then rebuilding the zones based on the current access rate to that zone.
- UPGRAID could sit above this process and use replicated stripes to improve this process.
Future Work - Reconstruction via PRO

- PRO works by dividing the failed disk into “hot zones” and then rebuilding the zones based on the current access rate to that zone.
- UPGRAID could sit above this process and use replicated stripes to improve this process.
- This approach would likely be more complex but its popularity based operation seems like a good fit with UPGRAID.
Future Work - Reconstruction via PRO

- PRO works by dividing the failed disk into “hot zones” and then rebuilding the zones based on the current access rate to that zone.
- UPGRAID could sit above this process and use replicated stripes to improve this process.
- This approach would likely be more complex but its popularity based operation seems like a good fit with UPGRAID.
- It may be good if we defined these “hot zones” to align with the stripes of the RAID5 disk. This would make reconstruction using the replicated stripes easier.
Extremely Preliminary Results
Extremely Preliminary Results

- Tested with autorwbench using one block (512 byte) I/O operations.
Extremely Preliminary Results

- Tested with autorwbench using one block (512 byte) I/O operations.
- 10MB write workload
Extremely Preliminary Results

- Tested with autorwbench using one block (512 byte) I/O operations.
- 10MB write workload
- 100MB read workload
Extremely Preliminary Results

- Tested with autorwbench using one block (512 byte) I/O operations.
- 10MB write workload
- 100MB read workload
- Run in a virtual machine
Extremely Preliminary Results

Random Access Tests

Second for Completion

Test Number
Extremely Preliminary Results

RAID5 Average - 12.187 seconds
Extremely Preliminary Results

Random Access Tests

- RAID5 Average - 12.187 seconds
- UPGRAID Average - 13.1984 seconds
Due to the current instability of the system this data should be taken with a grain of salt.
Questions or Comments?