Chapter 7

EXPERIMENTAL ANALYSIS FOR CRYPTANALYSIS DATA STORAGE

7.1 Introduction

The purpose of this chapter is to explain the Copyback of SMART feature being introduced into MegaRAID (Redundant Array of Inexpensive Disk) SAS (Serial Attached SCSI) products and provide the detailed design.

Example: The drive monitors the number of ECC (Error Checking and Correction) errors and based on the analysis data it can give predictive failure if the ECC error threshold exceeds internal to the drive [17,18]

The following topics are covers in this chapter:

- Description
- Design Goals
- Assumptions
- Restrictions and Limitations
- Redundant Array of Independent Disk (RAID)
- Objectives of the Project
- Literature Survey on SMART Copyback

Description

Copyback is a feature wherein a data from an ONLINE PD is copied to a destination Hotspare(HSP)/Unconfigured-Good PD. After successful completion, the destination Physical Disk(PD) becomes ONLINE (part of the array) while the source PD becomes Hotspare(HSP)/Unconfigured.

Copyback can be initiated in any of the following ways:
Automatically by FW (Firmware) for revertible Hotspare(HSP): Copyback is automatically initiated on replacement of the failed PD whose arm position has now been taken by a Hotspare(HSP). The source of copyback would be the now ONLINE HSP drive and the destination, the newly replaced disk. After copyback completes, the ONLINE HSP is reverted back to HSP and the replaced drive becomes ONLINE.

**Figure 7.1:** Copyback data from a HSP drive to replaced failed drive

Automatically by FW for SMART errors: If a configured PD has SMART errors and there exists a HSP that can replace this drive, then copyback is started from the SMART failure drive to a HSP.

**Figure 7.2:** Copyback data from SMART failure drive to a HSP
SMART error PD to the HSP. After copyback the source PD is marked Unconfigured Bad.

Manually by applications: The applications need to send a direct command with the source and destination PDs.

### 7.2 Terminology

The hard disk drive vendors build a logic into the hard drives to make drives smart so that the user gets warning signal as a “predictive failure” whenever the drive is about to go bad for some cause. The drives built with this kind of logic are called SMART hard disk drives.

Example: The drive monitors the number of ECC (Error Checking and Correction) errors and based on the analysis data it can give predictive failure if the ECC error threshold exceeds internal to the drive.

- **Hotspare:**

  Hotspare is a drive that is not part of any array, but can be commissioned to replace any failed disk of redundant array. The hotspare becomes part of the array once the rebuild is completed on it.

- **Rebuild:**

  Rebuild is process of reconstructing data on either a failed disk or a newly inserted disk or a hotspare in a RAID subsystem. The method to reconstruct depends on the RAID level. There are RAID Levels 1, 2, 3, 4, 5, 6, 10, and etc according to their requirements. In case of RAID1 the mirror copy is read from the other drive; for RAID5/RAID6, the peer drives are read and XORed to reconstruct the data on the rebuilding drive.

  Copyback is a process for copying data from a given source disk to the destination disk. This operation is much faster than rebuild since it is a direct disk-to-disk copy unlike the rebuild where the data is read from peer drives and reconstructed.
Firmware (FW):

Firmware (FW) is a computer program that is embedded in Controller that operates the controller.

Physical Disk (PD):

A physical disk state is a property indicating the status of the disk. The physical drive states are described below:

Physical Drive States Description as follows:

**Online**: Online A physical disk that can be accessed by the RAID controller and is part of the virtual disk.

**Offline**: A physical disk is offline when it is part of a virtual disk but its data is not accessible to the virtual disk.

**Unconfigured Good**: A physical drive that is functioning normally but is not configured as a part of a Virtual Disk (VD) or as a hot spare.

**Hot Spare (HSP)**: A physical drive that is powered up and ready for use as a spare in case an online drive fails.

Design Goals:

- Copyback will use the rebuild rate which can be configured by the user to control host IO verses the rebuild/copyback operation.

- User can enable or disable copyback in the controller properties. There will be no MFC settings for copyback. Smarter copy back can also be controlled via controller properties apart from “revertible/DCMD” copyback.

- PD Allowed operations (startCopyback/stopCopyback) will have bits set by the firmware for the applications to expose copyback operations in the GUI.

- The firmware supports revertible hotspares at a controller level meaning if CtrlProp.copybackDisabled is FALSE, then all HSP’s are assumed revertible. There is no control on a per HSP PD basis. Upon successful completion
(automatic/manual) of copyback from the source ONLINE drive (previously
HSP) to another drive (destination), the ONLINE hotspare will be reverted
back to HSP.

- Drives experiencing SMART errors will be a candidate for copyback source
to an available best fit hot spare, if controller property SMARTerEnabled is
TRUE. During SMARTer copyback, if any array becomes non-optimal (partially
degraded/degraded) and there are no other HSPs other than

- No other background operations except rebuild are allowed on an array when
copy back is active to avoid thrashing. However for rebuild on the same array,
copyback will be aborted. Automatic copyback (revertible or SMARTer) will
be restarted after rebuild is completed on the array.

- Log events to indicate copyback start, abort, progress and successful
completion.

- Recovery of data for the destination drive due to errors on reads for the source
drive will be handled similar to the host read commands.

- If the logical drive becomes non-optimal (partially degraded or degraded) due
to source copyback drive failure, copyback will be aborted and rebuild will
start on the destination copyback drive from PD0.

- Copyback takes a higher priority over restoreHotspareOnInsertion (if both are
enabled).
Assumptions

The design expects at least 512MB DRAM to be present on the controller to support copyback since copyback uses cache lines and other resources. It doesn’t mean that if you have lesser DRAM copyback will not run, but the results of having lesser resource tend to have effect on performance.

Restrictions and Limitations

- Copyback can be initiated only if no other background operations are in progress on that array.
- For revertible HSPs, when the failed drive is replaced, copyback will initiate from the now ONLINE HSP to newly replaced drive. However, if the failed drive is replaced while the system is offline, copyback will not be automatically started on bootup.
- FW limits the number of entries in HSP History to 64. The assumption is this limit would never exceed in reality.

Redundant Array of Independent Disks (RAID)

Redundant Array of Independent Disks (RAID) — is a technology that employs the simultaneous use of two or more hard disk drives to achieve greater levels of performance, reliability, and/or larger data sizes. When several physical disks are set up to use RAID technology, they are said to be in a RAID array [66]. This array distributes data across several disks. RAID drive groups improve data storage reliability and fault tolerance. In this case the data loss resulting from a drive failure can be prevented by reconstructing missing data from the remaining drives. RAID [67] has gained popularity because it improves I/O performance and increases storage subsystem reliability.

The RAID controller supports the following RAID levels:

- RAID level 0,
- RAID level 1,
- RAID level 5,
- RAID level 6,
- RAID level 10,
- RAID level 50 and
- RAID level 60.

**RAID 0**: (striped disks) distributes data across several disks in a way that gives improved speed and full capacity. In this case fault tolerance is no there. Therefore data on all disks will be lost if any one disk fails.

**RAID 1**: (mirrored disks) could be described as a backup solution, using two (possibly more) disks that each store the same data so that data is not lost as long as one hard disk survives. The maximum capacity of the array is just the capacity of a single hard disk.

**RAID 5**: (striped disks with parity) combines three or more disks in a way that protects data against loss of any one disk; the storage capacity of the array is reduced by one disk.

**RAID 6**: In this case striped set with dual (two) distributed Parity. So it provides fault tolerance from maximum two drive failures; array continues to operate with up to two failed drives. This makes larger RAID groups. This becomes increasingly important because large-capacity drives lengthen the time needed to recover from the failure of a particular single drive. So that the single parity RAID levels are vulnerable to data loss until the failed drive is rebuilt: the larger the drive, the longer the rebuild will take. Dual parity gives time to rebuild the array without the data being at risk if one drive, but no more, fails before the rebuild is complete.

**RAID 10**: A combination of RAID 0 and RAID 1, consists of striped data across mirrored spans. It provides high data throughput and complete data redundancy but uses a larger number of spans. RAID 10 is configured by spanning two contiguous RAID 1 virtual drives, up to the maximum number of supported devices for the controller. RAID 10 allows a maximum of eight spans. You must use an even number of drives in each RAID virtual drive in the span. The RAID 1 virtual drives must have the same stripe size.
RAID 50: a combination of RAID 0 and RAID 5, uses distributed parity and disk striping and works best with data that requires high reliability, high request rates, high data transfers, and medium-to-large capacity. RAID 60, a combination of RAID 0 and RAID 6, uses distributed parity, with two independent parity blocks per stripe in each RAID set, and disk striping. A RAID 60 virtual drive can survive the loss of two drives in each of the RAID 6 sets without losing data. It works best with data that requires high reliability, high request rates, high data transfers, and medium-to-large capacity.

A hot spare is an extra, unused drive that is part of the disk subsystem. It is usually in standby mode, ready for service if a drive fails. Hot spares permit you to replace failed drives without system shutdown or user intervention.

When a drive in a RAID drive group fails, you can rebuild the drive by recreating the data that was stored on the drive before it failed. The RAID controller recreates the data using the data stored on the other drives in the drive group. Rebuilding can be done only in drive groups with data redundancy, which includes RAID 1, 5, 6, 10, 50, and 60 drive groups. The RAID controller uses hot spares to rebuild failed drives automatically and transparently, at user-defined rebuild rates. If a hot spare is available, the rebuild can start automatically when a drive fails. If a hot spare is not available, the failed drive must be replaced with a new drive so that the data on the failed drive can be rebuilt.

RAID Functions

Virtual drives are drive groups or spanned drive groups that are available to the operating system. The storage space in a virtual drive is spread across all of the drives in the drive group. The drives must be organized into virtual drives in a drive group and they must be able to support the RAID level that you select. Below are some common RAID functions:

- Creating hot spare drives
- Configuring drive groups and virtual drives
- Initializing one or more virtual drives
- Accessing controllers, virtual drives, and drives individually
• Rebuilding failed drives
• Verifying that the redundancy data in virtual drives using RAID level 1, 5, 6, 10, 50, or 60 is correct
• Reconstructing virtual drives after changing RAID levels or adding a drive to a drive group.

Controller card that would normally be used for interfacing between the system and the RAID array. Firmware (FW) is a computer program that is embedded in Controller that operates the Controller.

➢ Coding Guidelines

Coding Guidelines for Physical Disk (PD):

Physical Disk (PD) file:

Provides information about the physical disk drives connected to the enclosure and adapter slot. This includes information such as the enclosure number, slot number, device ID, sequence number, drive type, size (if a physical drive), foreign state, firmware state, and inquiry data.

For Example: IDE drives, SATA drives, and SAS devices and so on. As the following information regarding Physical Disks as illustrated here.

➢ PD_INFO.c file

```c
typedef enum PD_STATE {
  PDS_UNCONFIGURED_GOOD = 0x00, // unconfigured - drive is good
  PDS_UNCONFIGURED_BAD = 0x01, // unconfigured - drive has failed
  PDS_SPARSE = 0x02, // drive is a spare
  PDS_OFFLINE = 0x10, // configured - data is invalid
  PDS_FAILED = 0x11, // configured - drive has failed
  PDS_REBUILD = 0x14, // configured - drive is rebuilding
  PDS_ONLINE = 0x16, // configured - drive is online
  PDS_COPYBACK = 0x20 // drive is in transition
} PD_STATE;
```

The PD_COPYBACK is the state of the PD getting copied (destination PD), the state of the source PD is not touched.

The below data structure is maintained per physical drive. This structure will be a union with PD_INFO.rebuild since most of the member variables are common across rebuild and copyback and both the operations cannot be active at the same time. Here the lowestActiveBlock/nextActiveBlock is the last/next blocks that were copied on to
the destination PD whose state is PDS_COPYBACK. Note that source PD state(PDS_ONLINE) will not change. copybackPairPd contains the destination deviceID for the source copyback PD and vice versa.

#define MAX_REBUILD_CMDS_PER_PD 8

typedef struct _PD_INFO
{
    .........
    struct {
        CMD_BLOCK    block;              // command blocking structure - used for auto rebuild

        uint64      lowestActiveBlock;  // lowest LBA of active rebuild op; equals nextActiveBlock when ops active

        uint64      nextActiveBlock;    // next block address to be rebuilt
        uint        previousMyTimerCount;
        ushort      state;              // rebuild state
        ushort      cpState;            // copyback state - rebuild and copyback states need to be exclusive
        uchar       ld;
        uchar       span;
        uchar       arm;
        uchar       pendingCmds;

        CMD_TYPE    activeCmdIds[MAX_REBUILD_CMDS_PER_PD];
        uint        copybackPairPd;
        uchar       pad_2[16];      // pad to 32-byte boundary (0x1c0)
    } copyback;
}

➢ Testing

Software testing is the activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. Testing is an integral part in software development. It is broadly deployed in every phase in the software development cycle. Typically, more than 50% percent of the development time is spent in testing.
Testing objectives include

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding a yet undiscovered error. A successful test is one that uncovers a yet undiscovered error.

➢ **Unit Test Cases:**

The following table shows the details of the every test case with description, input, expected output and actual result appeared.

➢ **Unit Test Cases:**

The following table shows the details of the every test case with description, input, expected output and actual result appeared.

➢ **Test Case: UTC-01**

The test case UTC-01 given below is the Creation of RAID1 using Two Physical Disks (PDs). It will take the input from the user and creates the RAID1 and generates a log file. As the below Table 7.1 illustrate the Test Case for virtual disk creation success test.

**Table 7.1: Test Case for Virtual Disk Creation Success Test**

<table>
<thead>
<tr>
<th>SI No. of Test Case</th>
<th>UTC -01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of test</td>
<td>Virtual Disk Creation Success test</td>
</tr>
<tr>
<td>Feature being tested</td>
<td>Creating a RAID1 using Two Physical Disks (PD)</td>
</tr>
<tr>
<td>Sample Input</td>
<td>Select Physical Disks (PDs)</td>
</tr>
<tr>
<td>Expected Output</td>
<td>Action Performed and its event log filed</td>
</tr>
<tr>
<td>Actual output</td>
<td>As Expected</td>
</tr>
<tr>
<td>Remarks</td>
<td>Test Successful</td>
</tr>
</tbody>
</table>
Heuristic Search Procedures for CDEC Techniques

Test Case: UTC-02

The test case UTC-02 given below is the Creation of RAID5 using Three Physical Disks (PD). It will take the input from the user and creates the RAID5 and generates a log file. As the below Table 7.2 illustrate the Test Case for virtual disk creation success test.

Table 7.2: Test Case for Virtual Disk Creation Success Test

<table>
<thead>
<tr>
<th>SI No. of Test Case</th>
<th>UTC -02</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of test</td>
<td>Virtual Disk Creation Success test</td>
</tr>
<tr>
<td>Feature being tested</td>
<td>Creating a RAID5 using Three Physical Disks (PD)</td>
</tr>
<tr>
<td>Sample Input</td>
<td>Select Physical Disks (PDs)</td>
</tr>
<tr>
<td>Expected Output</td>
<td>Action Performed and its log file</td>
</tr>
<tr>
<td>Actual output</td>
<td>As Expected</td>
</tr>
<tr>
<td>Remarks</td>
<td>Test Successful</td>
</tr>
</tbody>
</table>

➢ Test case Manual Copyback

This test case is illustrated in Table 7.3. This test case is used for Manual Copyback module to test the functionality of the function which writes data from Source Physical Disk (PD) to Destination Physical Disk (PD). Check Physical Disk (PD) Allowed Operations.

➢ Expected Result :

startCopyback should be TRUE for all ONLINE PDs
stopCopyback should be TRUE for all PDS_COPYBACK PDs

- Start copyback manually from ONLINE to Unconf. Good/HSP.

➢ Expected Result :

After copyback completes, the source would become Unconf. Good or Unconf. Bad(if it has predictive failure). The destination would become ONLINE.

Note that copyback can never be started to a destination HSP which already has a predictive failure.

- Stop copyback on PDS_COPYBACK PDs.
Note: Copyback Resume from Power Cycle: If the system is turned off while copyback is in progress, on a reboot copyback should resume (not restart) from where it left off. However, if the copyback source PD is removed while the system is off, rebuild should resume.

Table 7.3: Manual Copyback – failure test

<table>
<thead>
<tr>
<th>Serial No. of test case</th>
<th>UTC-06</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the Test</td>
<td>Manual Copyback – failure test</td>
</tr>
<tr>
<td>Item/Feature Tested</td>
<td>Check Physical Disk (PD) Allowed Operations.</td>
</tr>
<tr>
<td>Sample Input</td>
<td>Manual Copyback module to test the functionality of the function which writes data from Source Physical Disk (PD) to Destination Physical Disk (PD). Check Physical Disk (PD) Allowed Operations.</td>
</tr>
</tbody>
</table>
| Expected output        | startCopyback should be TRUE for all ONLINE PDs stopCopyback should be TRUE for all PDS_COPYBACK PDs  
                           • Start copyback manually from ONLINE to Unconf. Good/HSP. |
| Actual output          | After copyback completes, the source would become Unconf. Good or Unconf. Bad (if it has predictive failure). The destination would become ONLINE. Note that copyback can never be started to a destination HSP which already has a predictive failure.  
                           • Stop copyback on PDS_COPYBACK PDs. |
| Remarks                | The Hotspare (HSP) size should be more than an array size or equal to an Array size. As this condition is failed Because destination HSP size is less than an Array size |

7.3 Concluding Remarks

This Chapter has demonstrated the efficacy of SMART Copyback, and an outcome is publication “Self-Monitoring Analysis and Reporting Technology (SMART) Copyback”, at ICIP 2011, pp 463-469, ©Springer-Verlag Berlin Heidelberg 2011.
The hard disk drive vendors builds a logic in to the hard drives to make drives smart so that the user gets warning signal as a “predictive failure” whenever the drive is about to go bad for some cause. The drives built with this kind of logic are called SMART hard disk drives.

The publication of this research work is, “Overview of Differential Cryptanalysis of Hash Function Using SMART Copyback for Data”, Published at International Journal of Computer Science and Technology, Vol. 4, Issue 1, Jan - March 2013, pp 42-45.ISSN:0976-8491.ISSN:2229-4333 (Global Impact Factor: 0.289).

Chapter 8 compares all the 3 research methods along with overall research conclusion and possible further future work.