# Towards Long Term Data Quality in a Large Scale Biometrics Experiment


Hoang Bui
University of Notre Dame
hbui@cse.nd.edu

Diane Wright
University of Notre Dame
dwright2@cse.nd.edu

Clarence Helm
University of Notre Dame
chelm@cse.nd.edu

Rachel Witty
University of Notre Dame
rwitty@nd.edu

Patrick Flynn
University of Notre Dame
flynn@cse.nd.edu

Douglas Thain
University of Notre Dame
dthain@cse.nd.edu



## ABSTRACT

Quality of data plays a very important role in any scientific research. In this paper we present some of the challenges that we face in managing and maintaining data quality for a terabyte scale biometrics repository. We have developed a step by step model to capture, ingest, validate, and prepare data for biometrics research. During these processes, there are many hidden errors which can be introduced into the data. Those errors can affect the overall quality of data, and thus can skew the results of biometrics research. We discuss necessary steps we have taken to reduce and eliminate the errors. Steps such as data replication, automated data validation, and logging metadata changes are necessary and crucial to improve the quality and reliability of our data.


## 1. INTRODUCTION

Research in the field of biometrics depends not only on the quantity of data but also on the quality of data marshaled to support it. Recent advances in technology make collecting terabytes of biometrics data a reality. Collections of biometrics data have increased in both scope and size. Also, the quality of data has been improved significantly due to improvements in sensing technology. Data is being captured in higher resolution pictures, movies, and more accurate 3D scans. This growth poses new challenges to data management and data quality control.

Biometrics, like many modern science and engineering research fields, is data-driven. Data enters the research enterprise through sensors and is processed, yielding derivative data sets, some of which feed comparisons that are used to evaluate the sensing technology, the steps in the processing pipelines leading to the comparisons, and the comparison techniques. Such evaluations must be performed with statistical rigor, which drives the collection of data to support the conclusions reached. Management of this data is a demanding task and the data sets' integrity must be assured through appropriate management and validation techniques. The use of open source databases to store and maintain the integrity of data, coupled with web services and portals that allow crowdsourced evaluation work and data access, is an ideal management strategy for large data sets such as those used in biometrics.

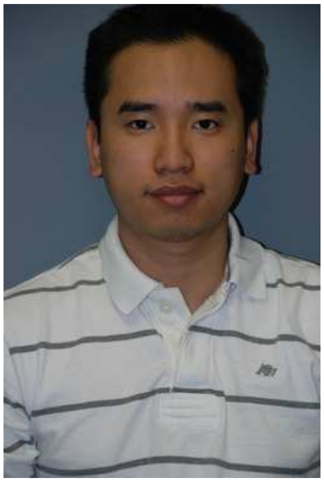

*MDQCS* 2010, Chicago, IL USA


At Notre Dame, we have developed a systematic process to collect, store, and manage biometrics data. We have created BXGrid [2], a biometrics data repository to help researchers collect, validate, manage, and query data in an efficient and consistent manner. Data consistency and availability play a very important role in producing and reproducing experiment results. Within BXGrid, data is processed, validated, and verified before it is released for use. Because data is kept within a single system, users can easily collaborate, share, analyze and verify others' results. In a previous paper, we discussed the architecture of BXGrid, and how users get data in, browse data, and get data out of BXGrid. This paper focuses on how we maintain and improve data quality, keep BXGrid healthy, and recover from hardware failure.

In the rest of this paper, we describe the biometrics data life cycle, challenges we face, and steps we have taken to maintain data quality and data integrity. In Section 2, we briefly introduce biometrics research and biometrics data to emphasize the importance of maintaining data integrity. We conclude Section 2 with a survey of BXGrid. In section 3, we describe the data acquisition and archival process. Next, we discuss common errors which can affect data quality. We present mechanisms we use to assure data quality. We then conclude the paper with some recent data on system hardware failure and our recovery efforts.

## 2. BIOMETRICS RESEARCH

Biometrics researchers study human body characteristics in the context of identification. Scientists develop algorithms to identify and confirm a human identity by comparing those characteristics with a known set and using a measurement of a physical trait. There have been a number of studies detailing the effectiveness of using human body character-



| | | |
|---|---|---|
| id | numeric | 64427 |
| recordingid | string | nd4R91445 |
| shotid | string | 2009-084-004-neutral.NEF |
| sequenceid | string | 05432d373 |
| date | string | 2009-03-25 00:00:00 |
| format | string | nef |
| subjectid | string | nd1S05432 |
| glasses | string | No |
| source1 | string | Retrospectively |
| emotion | string | BlankStare |
| source2 | string | Given |
| stageid | string | nd4T00014 |
| weather | string | Inside |
| collectionid | string | nd1C00031 |
| environmentid | string | nd1E00069 |
| sensorid | string | nd1N00012 |
| illuminantid1 | string | nd1I00010 |
| state | string | enrolled |
| fileid | numeric | 430941 |
| by_user | string | slagree |
| lastcheck | string | 2009-04-09 09:49:11 |
| date_added | string | 2009-03-31 10:00:40 |
| added_by | string | dwright2 |
| temp_collectionid | string | 1238508120 |
| YOB | numeric | 1982 |
| gender | string | Male |
| race | string | Asian |

**Figure 1: Sample Face Image and Metadata**

istics such as fingerprint [7], hand [4], iris [3], and face [12] to identify or to verify an identity claim. However, questions remain about how to improve speed and reliability of the identification process. Nowadays, with the popularity of cloud computing [9], biometrics researchers have a very powerful tool to study the correctness and effectiveness of their biometrics recognition algorithms.

The Computer Vision Research Lab (CVRL) at the University of Notre Dame collects hundreds of gigabytes of biometrics data every semester. Data is ingested and maintained in the BXGrid system for internal use to study newly developed algorithms. Data is also exported and shipped to the National Institute of Standards and Technology (NIST) to enter into a national research database. Examples of biometrics data are iris, face (still images and movies), and 3D face scans. Every image, movie, and scan collected is considered a **recording**. There are a number of metadata attributes attached to a recording. The metadata is gathered during the aquisition of each recording. Figure 1 shows an example of a face recording of one of the authors.

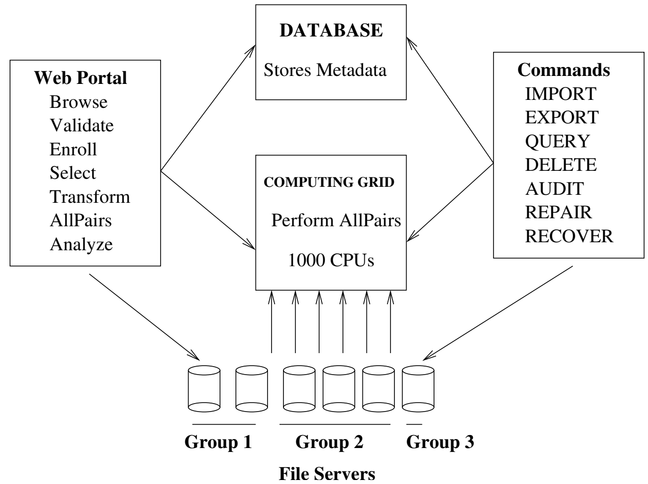
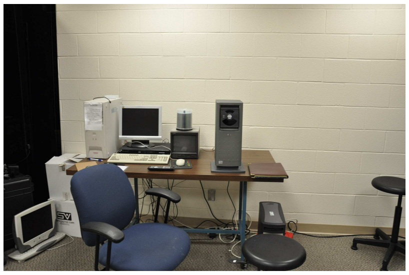



**Figure 2: System Architecture.** Database acts as a metadata cache to provide fast metadata queries. Actual data files are stored in an active storage cluster. User's requests are served directly from the storage cluster. Storage servers are grouped into three groups.

We created BXGrid in order to support storage of data and metadata and to facilitate large scale experiments. BXGrid consists of three main components: a database, an active storage cluster, and a computing grid. Figure 2 shows the architecture of BXGrid. BXGrid employs a relational database to manage metadata. It acts as a cache to enable fast data selection query. Throughout the life cycle of biometrics data, metadata can be changed, updated, or removed. When metadata is modified, the change will be made in the database first. Because the database is not the only place BXGrid stores metadata, the change will be flushed to secondary storage to assure metadata consistency. We further discuss how BXGrid handles the changes of metadata in Section 5.2. Actual image and video files are stored in an active storage cluster running Chirp [10]. The cluster contains forty off-the-shelf commodity machines with a large single hard drive in each and one Sun x4500 storage server with multiple terabytes of hard drive. Each image or video and its metadata is replicated and stored across three different servers in the cluster. By storing data on three distinct servers and having a database as a metadata cache, BXGrid can provide high throughput, scalable data access to tens of users simultaneously. To provide computational power for a large-scale biometrics experiment, BXGrid is connected to a campus computing grid of 1000 cores running Condor [11].

## 3. OVERVIEW OF ACQUISITION

The CVRL collects data bi-weekly during Fall and Spring semesters. Each acquisition involves several lab technicians and employs a number of biometrics sensors. Acquisition needs to be carried out as quickly as possible, according to a plan to ensure the quality of data collected and the correctness of derived metadata. Acquisition usually includes



(a) Iris photo station.

(b) Face photo station.

**Figure 3: Example of Acquisition Stations.**

a number of stations. Each station requires one or more lab technicians to monitor and capture data as subjects proceed through. A station uses one or multiple sensors with different lighting conditions. A sensor can produce more than one recording. A recording can be a picture, a movie, or a 3D scan.

## 3.1 Acquisition Setup

The first step of any acquisition session is to set up the stations based on an acquisition specification. The job of the setup technician is to follow the specification in order to determine the placement of sensors (camera, camcorder, scanner) and illuminant sources. Figure 3 shows two stations – one is the station that captures iris photos while the other station captures face photos. In addition, the specification provides the position of subjects and number of recordings captured per subject per sensor. After setting up the station according to the specification, technicians perform a mock acquisition to make sure the equipment functions properly, and then eliminate any remaining problems observed.

## 3.2 Data Acquisition

As subjects start an acquisition session, each of them is given a **session id**. This **session id** is used to synchronize each captured recording and its metadata, such as subjectid, stageid, eye color, etc. Subjects go through a number of stations, recordings are captured at each station, and metadata is recorded. During the acquisition, technicians capture these data and act as the first quality screening gate. They make sure that eyes are open during iris acquisition, faces are unobstructed during face acquisition, and so on. They will initiate re-acquisition if deemed necessary.

During acquisition, metadata is captured along with each recording. Metadata includes lighting conditions, sensor specifications, relative position of subject to sensor and lighting (i.e subject 6' away from camera and illuminant 8' above ground, 6' directly in front of subject). Other metadata contains personal information regarding the subject, such as eye color, race, and age. Another set of metadata is a recording of specifications such as format, resolution, and length (for video).

## 3.3 Pre-ingestion Assembly of Data

After acquisition, there are several types of recordings that need to be processed before ingesting. HD video needs to be clipped by subject, renamed, then transcoded to MPEG format. BMP images need to be converted to TIFF format. Iris videos need to be clipped by subject and eye (left,right), then transcoded to MPEG format and renamed. Data and metadata need to be gathered and synchronized before ingesting into a distributed storage system. While computer controlled sensors have the session id built into the recording's filename, manually operated sensors' recordings need to be renamed. The new filename includes session id, date, description of a recording (regarding either quality – high, low – or activity classification – still, movement, etc.)

The next step is to collate metadata from various sources into a spreadsheet that links it to the correct recording. Some data comes from subject registration, e.g., eye color, glasses, age; some is environment-dependent e.g., sensor id (sensor information), illuminant id (lightning information). Metadata is then converted to name value pair format and is ready to be ingested. The name value pair format is similar to the metadata shown in Figure 1. Metadata name, type (numeric or string), and value are separated by tabs, while recordings are separated by an empty line.

## 3.4 Data Ingestion and Data Storage

After being prepared, data is ingested into BXGrid by invoking an `IMPORT` command. BXGrid automatically replicates data and associated metadata across multiple storage servers. BXGrid provides data redundancy to assure data quality and data integrity. Data information such as size of file and checksum are kept internally inside BXGrid. Figure 4 shows the BXGrid internal relationship between recordings, files, and replicas.

## 3.5 Data Validation

The data collected through the CVRL acquisitions are used for research into biometric recognition algorithms by institutions throughout the world. Therefore, it is of the utmost importance that it is tagged correctly. Despite best efforts and procedures during the acquisition and ingestion processes, errors can slip through. In order to ensure the accuracy of the records, research assistants visually check the recordings to spot any errors. A recording consists of an image and its related metadata. The problems encountered fall into two categories: image quality and incorrect metadata. Image quality problems might include blurriness of an image, intended feature not visible (eye closed, part of face cut off, image too dark/light, etc.), which make the image unfit for use in research. The more difficult problems to ferret out and resolve are those that involve incorrect metadata. This

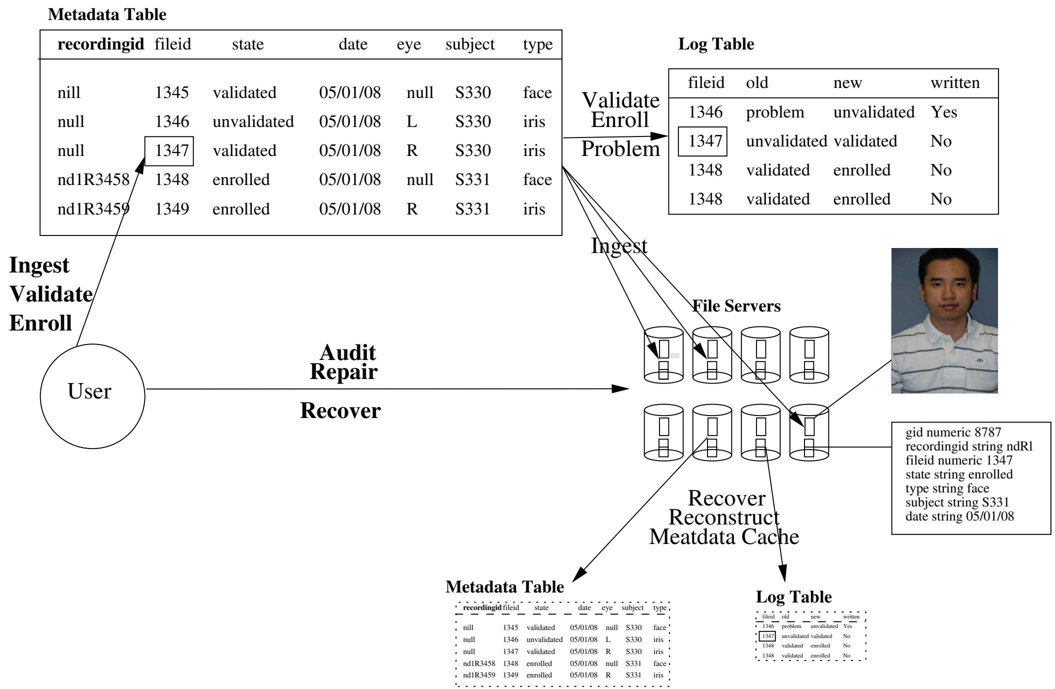



**Figure 4: BXGrid Database Schema.** Each metadata entry can be associated with one or more *fileid*. Each *fileid* then points to a number of *replicaids* which hold the location of a replica in BXGrid Storage Cluster. Every metadata change is logged into a log table, and it is flushed to the storage server later. In case of loss or damage to a database, BXGrid scans for metadata at the storage server level to reconstruct the database.

can range from an image tagged as a subject wearing glasses when he/she is not, to an image being tagged with the wrong subject number. The validation function of BXGrid allows for an efficient visual comparison of a newly acquired image against similar images of the same subject while also displaying the relevant metadata tags in a concise format. Records that have no problems are validated. If an error in the metadata or a problem with the image quality is discovered, the record is designated as a "problem" record which can be either eliminated from the data set, or corrected and validated at a later date.

### 3.6 Data Enrollment

The final step in processing a recording is to enroll it. Once a record is enrolled, it should not be edited or changed in any way. During the enrollment process, a record is associated with a **collection** and given a **recordingid** that is used to identify the image in any subsequent research and ensuing publication. BXGrid supplies the structure for creating a collection in a format that is consistent with a US government Document Type Definition Reference Document. It provides a template for naming the collection and allows the user to specify the type of data and the acquisition dates to be included with a few simple buttons. Once the user verifies her choices, BXGrid generates the recordingid for each of the included images and adds the collection to the collections table.

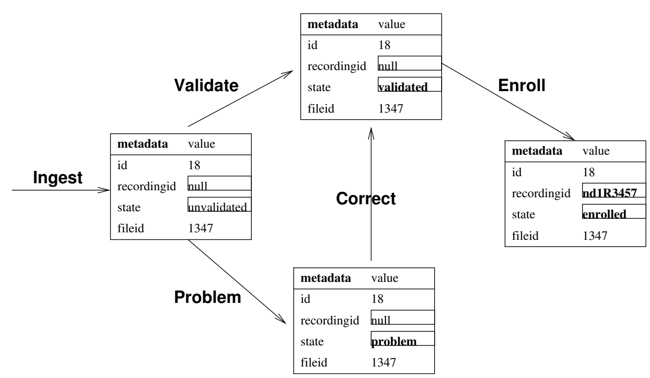



**Figure 5: Data Life Cycle.** Metadata changes during validation process. New metadata is assigned when data is enrolled.

## 4. ISSUES THAT CAN AFFECT DATA QUALITY

Creating and maintaining a large repository of biometrics data can be challenging in many ways. One hundred thousand data files can add up to terabytes of data. Because of the size of the repository and the fault-prone nature of both humans and computers, data quality can be affected throughout the life cycle of data. Error can be introduced into data at any time during pre-acquisition, during acquisition, during ingestion and after ingestion. Depending on the nature of the errors, solutions to correct errors can be recapturing data, modifying metadata, or removing data completely.

| ***Stage*** | Problem | Solution |
|---|---|---|
| Acquisition | Equipment malfunctions | Discard image/movie, reset, or replace equipment |
| Acquisition | Subject jumps out of order | Lab technicians detect and correct the order |
| Ingestion | Ingestion is interrupted | Re-run ingestion command |
| Validation | Incorrect metadata | Lab technicians correct metadata using web portal |
| Validation | Length of validation process | Automated Data Validation |
| Validation | Metadata inconsistency | Two phases of metadata update, database then flush to storage |
| Archival | Hardware failure | Replicate and store metadata in three storage servers |
| Archival | Data inconsistency | Audit and Repair process |
| Archival | Validate/Enroll errors | Revert using metadata log |
| Archival | Loss of database | Recover by scanning metadata from fileservers |

**Table 1: Summary of Problems and Solutions.**

During acquisition, equipment can malfunction (e.g. a camera does not take a picture, the flash does not trigger). Other errors can be due to carelessness of lab technicians (e.g. camera has a wrong zoom setting, unnoticed blinking eyes at the time of data capture). Another error occurs when subjects get out of order during acquisition. Figure 6 shows some of the problem recordings. Because each acquisition usually includes a number of stations, a subject jumping the station line will cause a string of mislabeled data. This proves to be costly when data is enrolled and used in experiments because it can inadvertently affect experiment results. Mistakes during acquisition can be easily corrected if the lab technician pays attention during operation and identifies the mistakes. Once a mistake is identified, steps are carried out to correct the mistake, ranging from logging the discrepancy to retaking a picture or a movie.

After acquisition, the lab technician uses various tools to prepare the data for the ingestion process. Data collected during acquisition is copied into local storage for pre-processing purposes. A script is used to rename the default filename to a more meaningful one. Data, such as video, will be edited. Problems may arise when the renaming script does not perform as intended or when video cutting fails. Mistakes during this stage can be eliminated by carefully processing data and also by maintaining a stable,working set of tools.

When data is ready to ingest, the lab technician invokes an `IMPORT` command to ingest data into the repositories. Each ingestion is assigned a **batchid**. The batchid is very useful for keeping track of each data acquisition, and also for correcting mistakes when mistakes are made. Ingestion can fail unexpectedly due to malfunctioning hardware or power outage. When ingestion is interrupted, the lab technician can invoke the same `IMPORT` command to resume the ingestion. `IMPORT` command will automatically start where the last `IMPORT` command left off. `IMPORT` also has built-in redundancy detection. When a batch is ingested twice, `IMPORT` will ignore already ingested data. When a batch needs to be deleted due to error, the lab technician can identify batchid and invoke `DELETE` to erase the batch from the repository.

The last step to assure data quality before enrollment is the validation process. Lab technicians validate data using a web portal. The web portal allows the technician to identify poor quality data by displaying data and comparing data from the same subject. Common metadata mistakes are mislabeling, such as left eye to right eye and vice versa, subject wrongly marked as wearing glasses, and data assigned to the wrong subject. By providing a comparison view between unvalidated data and already validated data from the same subject, lab technicians have a better chance of detecting these types of mistakes and correcting them accordingly. Figure 7 shows an example of a validation page.

# 5. DATA QUALITY CONTROL MECHANISM

Data quality plays a very important role in the success of an experiment. Data and metadata have to match correctly. Wrongly matched data and metadata can alter the result of an experiment. BXGrid employs a number of mechanisms to assure the correctness, consistency, and availability of data. Table 1 lists the problems we have identified and steps we take to minimize or eliminate data quality problems.

## 5.1 Derived Dataset Consistency

BXGrid allows users to create, derive and manage their own datasets from the web portal. When a user wants to study a particular aspect of biometrics data, he/she can create a dataset containing only the data he/she is interested in. For example, in order to focus on Asian irises, the user issues a query to create a new dataset containing only Asian irises. Let us call the new dataset A. Furthermore, if the user chooses to split the newly-created dataset into male and female irises, the two new datasets could be derived from set A. Let us call these datasets B and C. He or she can can choose to make A, B and C public in order to share them with other users.

The dataset inconsistency problem arises when more data is added to BXGrid and also when metadata changes. The question is whether or not a dataset should be updated to reflect the changes. A static dataset is useful when the user wants to repeat the same experiment with an unchanged set of data. Moreover, scientific experiments may have to be reproducible with the same set of input data. On the other hand, there is a strong argument to support dynamic datasets. A dynamic dataset can be changed because data is constantly added to BXGrid. Metadata also can changes. Additionally, users can encounter bad data and wish to eliminate it from their datasets. If the dataset is static, the bad data and outdated metadata will stay and propagate down to future derived datasets.

BXGrid keeps track of the last time a dataset was updated.

(a) Wrong camera position.

(b) Out of focus camera.

(c) Blinking eye.

**Figure 6: Example of problem recordings.**

When a dataset is accessed, BXGrid will warn the user if any ancestor dataset has been changed. A user can conveniently choose when to update their datasets. Changes in an ancestor dataset, in this case set A, will be propagated to the derived datasets, set B and set C. Since January 2010, there have been 229 instances of metadata change.

At the beginning, BXGrid handled a dataset as a MySQL view. However, we soon found out that using views for datasets is not a good idea. Although a MySQL view does not add anything to the database, a view needs to be recomputed everytime the user accesses the corresponding dataset. Recomputing the views is a very expensive transaction, especially for complicated queries. To solve this problem, BXGrid now treats datasets as tables instead of views.

## 5.2 Automated Data Validation Process

Validating data is a necessary but cumbersome process. Lab technicians browse through thousands of data recordings to identify and correct data errors. During the iris image validation process, lab technicians usually look for mistakes such as closed eye, incorrect subject, etc. Figure 7 shows an example of an iris image validation page. An iris image is displayed next to five **good** iris images from the same subjects. A good iris is an iris that has been validated and verified to be correctly linked to that subject. Although there are attributes of a person's eye that could change over time[1], the eyes usually retain their similarities, such as shape and size of the iris. By looking at iris images from the same subject, lab technicians can easily spot an iris that does not belong to the assigned subject. The same process applies to the face validation process.

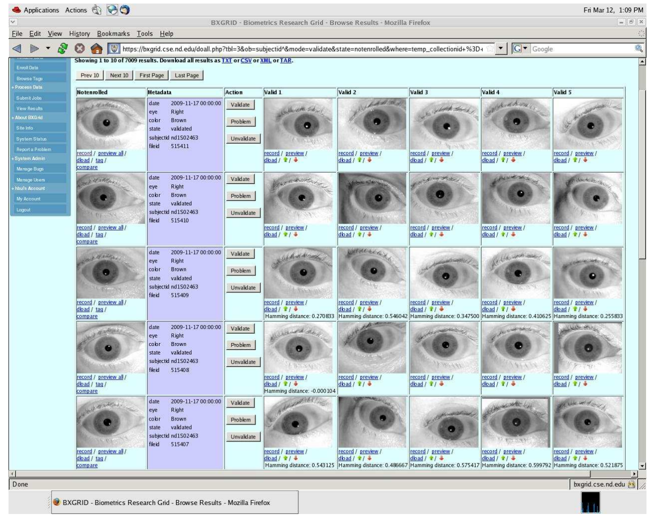


**Figure 7: Iris Images Validation Page.** This is a screenshot of the BXGrid web portal validation page. On the left, it displays unvalidated irises. On the right, it displays five already validated irises from the same subject. By putting unvalidated irises and validated irises side by side, lab technicians can easily spot problem irises and take appropriate action.

In order to speed up the iris validation process, BXGrid provides an automated tool that compares **unvalidated** iris images to previously validated iris images. Each comparison calculates the hamming distance between two iris images. The lower the score, the better the match is. The results of the comparisons are put into a matrix. Using the matrix, BXGrid singles out **bad** irises, and makes suggestions to lab technicians. Based on the suggestions, the lab technician can make an informed decision on whether to validate an iris. Rejected irises are then marked as a **problem** for further investigation. Figure 8 shows an example of an automated comparison matrix. This matrix is only 5x5, thus it can be generated and cached using a local web server in seconds. Bigger matrices, which contain much more data and are computational-intensive, are generated in a distributed fashion using All-Pairs[5] . Lab technicians issue requests for the automated comparison matrix, and BXGrid handles the calculation in the background and notifies technicians upon completion.

## 5.3 Metadata Consistency

As described before, metadata of a recording changes throughout its lifetime. Lab technicians modify a metadata attribute when the value is incorrectly assigned, for example, a right iris is wrongly marked as a left iris, or a subject is marked as not wearing a pair of glasses though wearing them. Another place where a metadata attribute may change is when the **state** of a recording changes from unvalidated to **validated** and ultimately to **enrolled**. Especially when a recording is enrolled, numbers of metadata attributes are assigned to the recording (to satisify NIST specification requirements) and include **sequenceid**, **recordingid** and

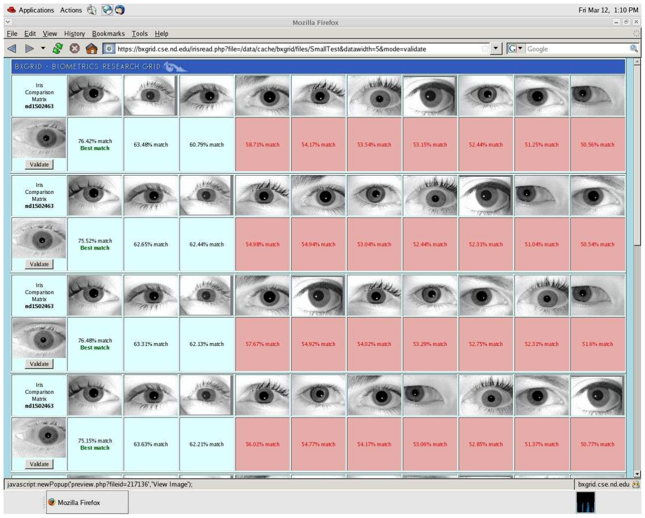


**Figure 8: Iris Automation Comparison Matrix.** This is a screenshot of the BXGrid Automation Comparison Matrix. The left irises represent the probe, while the top right irises represent the gallery. Comparisons between the probe and the gallery produce the result matrix.

**collectionid**. Figure 5 shows the life cycle of a recording. A recording goes from unvalidated to validated or problem. When a lab technician corrects the error, the recording is marked as validated, and it is ready for enrollment. After being enrolled, a recording state changes to enrolled with newly assigned sequenceid, recordingid and collectionid.

Each metadata change is written to the database intermediately. However, those changes have not been reflected at the storage level yet. BXGrid could write changes to both database and storage servers at the same time. However, because of the time discrepancy between a database update transaction and a disk write transaction, writing changes to both is lagged and bounded by slow disk speed. Especially when there are mass metadata changes during the enrollment process, writing thousands of small transactions to disk can take minutes to hours.

In order to maintain data consistency, BXGrid logs all metadata changes in a log table. The log table keeps track of what has been changed, who made the change, and when the change was made. During the BXGrid low usage period (mostly during nighttime), those metadata changes are flushed to the storage level to reflect the permanent changes in metadata. By doing so, BXGrid avoids the unnecessary wait time to write to disk, keeps BXGrid in operational mode and guarantees a level of metadata consistency.

## 5.4 Data Consistency and Data Availability

BXGrid uses Chirp [10] as its backend storage system. BXGrid automatically adds a level of availability and consistency to the repository. BXGrid's transparency failover mechanism provides seamless access to the repository and keeps BXGrid in operational mode. Each item of data is replicated to three file servers. Each replica is accessible by all BXGrid's users. In the event that one or more servers fail, a read request would try the next available server for a good replica. Unless all three servers hosting the same data file fail at the same time, BXGrid availability is unaffected.

The first and foremost goal of BXGrid is to be reliable. Once data is imported, it is supposed to survive hardware failures. Periodically, each file is audited to ensure its availability and consistency. The `AUDIT` process works as follows: for each file, locates every replica, computes the checksum and the size of the replicas, then compare those values with the values stored when data was ingested into the repositories. If a replica is unreachable or its return values do not match the original values, the replica is marked as **suspect**. After checking all replicas, if the number of **ok** replicas is less than three, one or more replicas are created to maintain the desired number of replicas for each data file. The **suspect** replicas can be purged or further examined at the system administrator's discretion. The `AUDIT` is very time-consuming and can slow down the performance of BXGrid. The administrator does not have to invoke `AUDIT` every day for the whole repository. `AUDIT` has an option to examine a portion of the repository, and it usually runs during the weekend.

# 6. RECENT DATA ON FAILURE RATES AND RECOVERY MECHANISMS

Hardware failure is not uncommon, especially hard drive failure. A hard drive can fail because it is exposed to extreme conditions, such as heat, humidity, water, shock, etc. It also can fail due to use or aging [6], [8]. Google [6] published a study on hard drive failure rate in 2007. Although the annualized failure rates are higher than those reported by hard drive manufacturers, given the scope and size of a Google disk farm, the number provided on hard drive failure rate is deemed to be accurate. According to Google, 2 percent of disks fail within a year, but the annualized failure rate jumps to 8 percent over two years and 9 percent in the first three years. The study shows that in order to sustain data through hard disk failure, we should plan to backup, replicate and audit data more often, and we should plan to provision new hard drives to replace old ones that are prone to failure.

Hardware failure is unavoidable for a production system like BXGrid. BXGrid employs as many as 41 file servers, and after a year of operation, some of them have already suffered hardware failure. Most common failures are bad hard drives and bad SATA controller boards. In the case of a bad hard drive, a new hard drive is added to replace the bad one, and all data is lost. In the case of a bad SATA controller board, data is intact and recoverable with a new controller board. As recent studies on hard drive failure show, system administrators need to run data audit and repair frequently. However, as the amount of data grows, it is not feasible to perform auditing on the whole system every day. Thus we have been running audit and repair only during the weekend and where BXGrid usage is minimal. In order to test BXGrid's ability to recover from hard drive failure, we intentionally removed several hard drives from the storage cluster. We ran BXGrid audit and repair incrementally to detect and replace missing replicas. Table 2 shows the length of each audit and repair run, number of audited file and number of repaired replicas. During the recovery process, BXGrid remained in operational mode, and was accessible by multiple users performing regular tasks, such as import, export, validate, enroll, etc.

| Period | Elapsed Time | Files Checked | Suspect |
|---|---|---|---|
| 1 | 24 hours | 80,000 | 16,244 |
| 2 | 24 hours | 80,000 | 15,153 |
| 3 | 48 hours | 160,000 | 1,227 |
| 4 | 16 hours | 60,000 | 9,381 |
| 5 | 32 hours | 160,000 | 0 |
| 6 | 28 hours | 160,000 | 0 |

**Table 2: Audit and Repair Timeline.** Elapsed time and number of suspect replicas found when hardware failure was deliberately introduced into BXGrid.

During December 2009, four storage servers suffered from hard drive failures. After identifying problem servers, BXGrid `REPAIR` was invoked to spawn new replicas. These replicas replaced those from problem servers and kept the number of replicas for each file at three. The repair process took just over five hours to replace an estimated 26,000 missing replicas, a total 250GB of data. The repair process took significantly less time than the audit process because auditing involves expensive checksum calcuations. Repair process throughput is mainly bounded by the speed of network links between storage servers, a mixture of Gigabit and 100Mbps network.

# 7. CONCLUSION

In this paper, we discuss challenges we have faced in maintaining data quality and data integrity in BXGrid. Table 1 lists the problems which can affect the quality of data, the availability of data and the integrity of data. It also sums up solutions we employ to eliminate mistakes and improve our data collection and validation process. We also describe mechanisms to deal with both software and hardware failure. To date, BXGrid has 348,885 recordings with triple replicas, totaling 7.17TB of data in 41 storage servers.

The impact of BXGrid on biometrics research activity at Notre Dame has been significant and positive. It has enabled the development of workflows for ingestion, validation, and enrollment that did not exist before BXgrid (all earlier data set constructions were done by hand, by different people, and yielded unstructured piles of customized scripts with variable quality and accuracy). Biometrics group members are not forced to fret about the nuts and bolts of data management as frequently, and can access and use data with the assurance that quality checks have been performed.

Data is being collected and ingested into BXGrid every other week. The front end web portal can support tens of users simultaneously. Validation workload is divided and distributed among staffers. In the future, we hope to improve automation validation for iris images and add automation validation processes for face images.

Acknowledgment: This work was supported by National Science Foundation grants CCF-06-21434, CNS-06-43229, and CNS-01-30839.

# 8. REFERENCES

[1] S. Baker, K. Bowyer, and P. Flynn. Empirical Evidence for Correct Iris Match Score Degradation with Increased Time-Lapse between Gallery and Probe Matches. In *Proceedings of International Conference on Biometrics 2009*, pages 1170–1179, June 2009.

[2] H. Bui, M. Kelly, C. Lyon, M. Pasquier, D. Thomas, P. Flynn, and D. Thain. Experience with BXGrid: A Data Repository and Computing Grid for Biometrics Research. *Journal of Cluster Computing*, 12(4):373, 2009.

[3] J. Daugman. How Iris Recognition Works. *IEEE Trans. on Circuits and Systems for Video Technology*, 14(1):21–30, 2004.

[4] A. K. Jain, A. Ross, and S. Pankanti. A Prototype Hand Geometry-Based Verification System. In *Proc. Audio- and Video-Based Biometric Person Authentication (AVBPA)*, pages 166–171, 1999.

[5] C. Moretti, J. Bulosan, D. Thain, and P. Flynn. All-Pairs: An Abstraction for Data Intensive Cloud Computing. In *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 1–11, 2008.

[6] E. Pinheiro, W.-D. Weber, and L. A. Barroso. Failure trends in a large disk drive population. In *USENIX File and Storage Technologies*, 2007.

[7] N. Ratha and R. Bolle. *Automatic Fingerprint Recognition Systems*. Springer, 2004.

[8] B. Schroeder and G. A. Gibson. Disk failures in the real world: what does an mttf of 1,000,000 hours mean to you? In *USENIX File and Storage Technologies*, 2007.

[9] D. Thain and C. Moretti. Abstractions for Cloud Computing with Condor. In S. Ahson and M. Ilyas, editors, *Cloud Computing and Software Services*. CRC Press, 2009.

[10] D. Thain, C. Moretti, and J. Hemmes. Chirp: A Practical Global Filesystem for Cluster and Grid Computing. *Journal of Grid Computing*, 7(1):51–72, 2009.

[11] D. Thain, T. Tannenbaum, and M. Livny. Condor and the grid. In F. Berman, G. Fox, and T. Hey, editors, *Grid Computing: Making the Global Infrastructure a Reality*. John Wiley, 2003.

[12] W. Zhao, R. Chellappa, P. Phillips, and A. Rosenfeld. Face Recognition: A Literature Survey. *ACM Computing Surveys*, 34(4):299–458, 2003.