AGH UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Managing data availability and integrity in federated cloud storage

Krzysztof Styrc
Department of Computer Science AGH

Supervisor:
Marian Bubak

Consultancy:
Piotr Nowakowski, ACC Cyfronet AGH

Reviewer:
Dawid Kurzyniec, Google Kraków

Outline

1. **Introduction**
   - Motivation
   - VPH-Share project background
   - Objectives of the thesis
2. **State of the art**
   - Overview of methods for data integrity
   - Proof of Retrievability (POR)
   - Data integrity proof (DIP)
3. **Design and implementation**
   - Data validation algorithm
   - Design of Data Reliability and Integrity (DRI) service
   - Example of DRI service operation
4. **Summary and future work**

Motivation

Cloud storage problems

- data stored on external resources of (untrusted) cloud provider
- best-effort SLAs definition, return of costs otherwise
- cloud vendor lock-in effect
- numerous cloud storage failures and security flaws
  - deleted emails, millions of blocked accounts in Gmail service
  - multiple Amazon S3 downtimes reports
  - unauthorized access to files in GoogleDocs

Cloud storage data integrity challenges

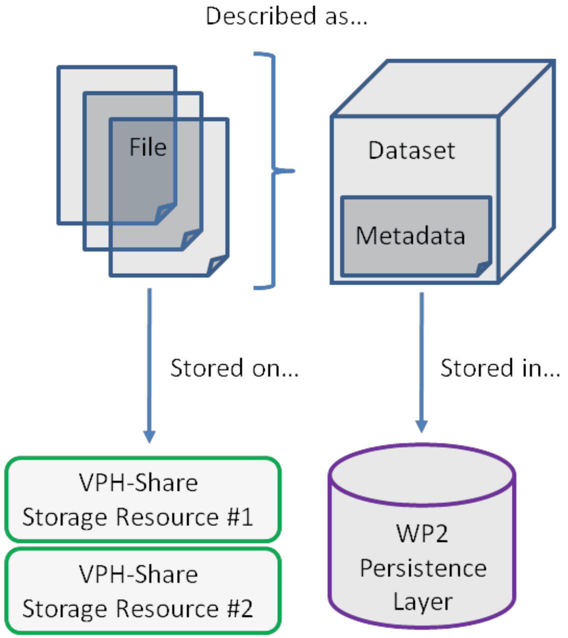
- network latency and bandwidth limits
  - fine-grained access pattern overhead
  - WAN networks (Internet) bandwidth outages
- costs of
  - data storage (per volume)
  - data transfer (per # of requests, per volume)
- simplified API, no computation available without retrieval

C. Cerin et al: Downtime statistics of current cloud solutions, IWGCR, June 2013
Amazon S3, Google Cloud Storage, Rackspace storage official pages

VPH-Share project background

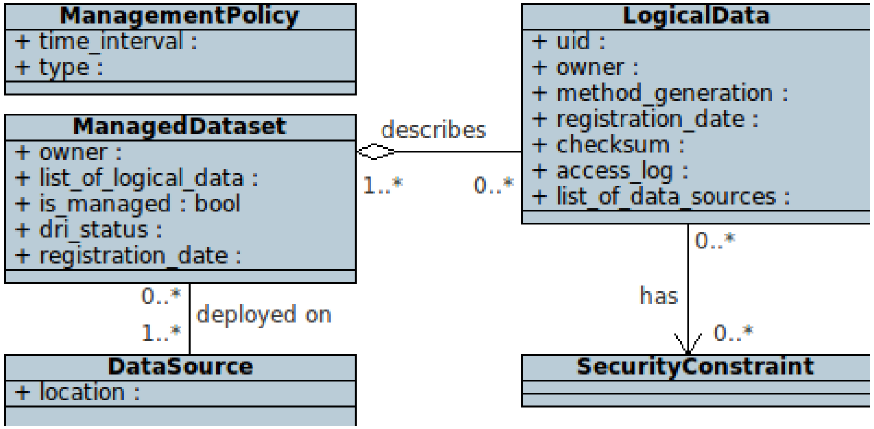
Data in VPH-Share

- mostly **static** and **sensitive**, biomedical data
- stored in federation of cloud storage providers to
  - avoid vendor lock-in effect
  - provide fault tolerance against provider failures
- storage entity defined as **dataset** (simply, a set of files)



Data integrity requirements

- periodical monitoring of data availability and integrity
- network-efficient data validation
  - reduce whole-file retrieval overhead
  - reduce costs
- replication of datasets in cloud federation



P. Nowakowski et al: VPH-Share WP2: Data and Compute Cloud Platform, Deliverable 2.2, VPH-Share 2011

Objectives

The aim of this thesis is to develop a method to efficiently monitor the availability and integrity of data stored in federated cloud storage.

Detailed objectives of this work

- analysis of available efficient cloud storage validation algorithms
- design of a network-efficient data validation algorithm
- design and implementation of the data validation web service prototype
- integration with the VPH-Share platform

Overview of methods for data integrity

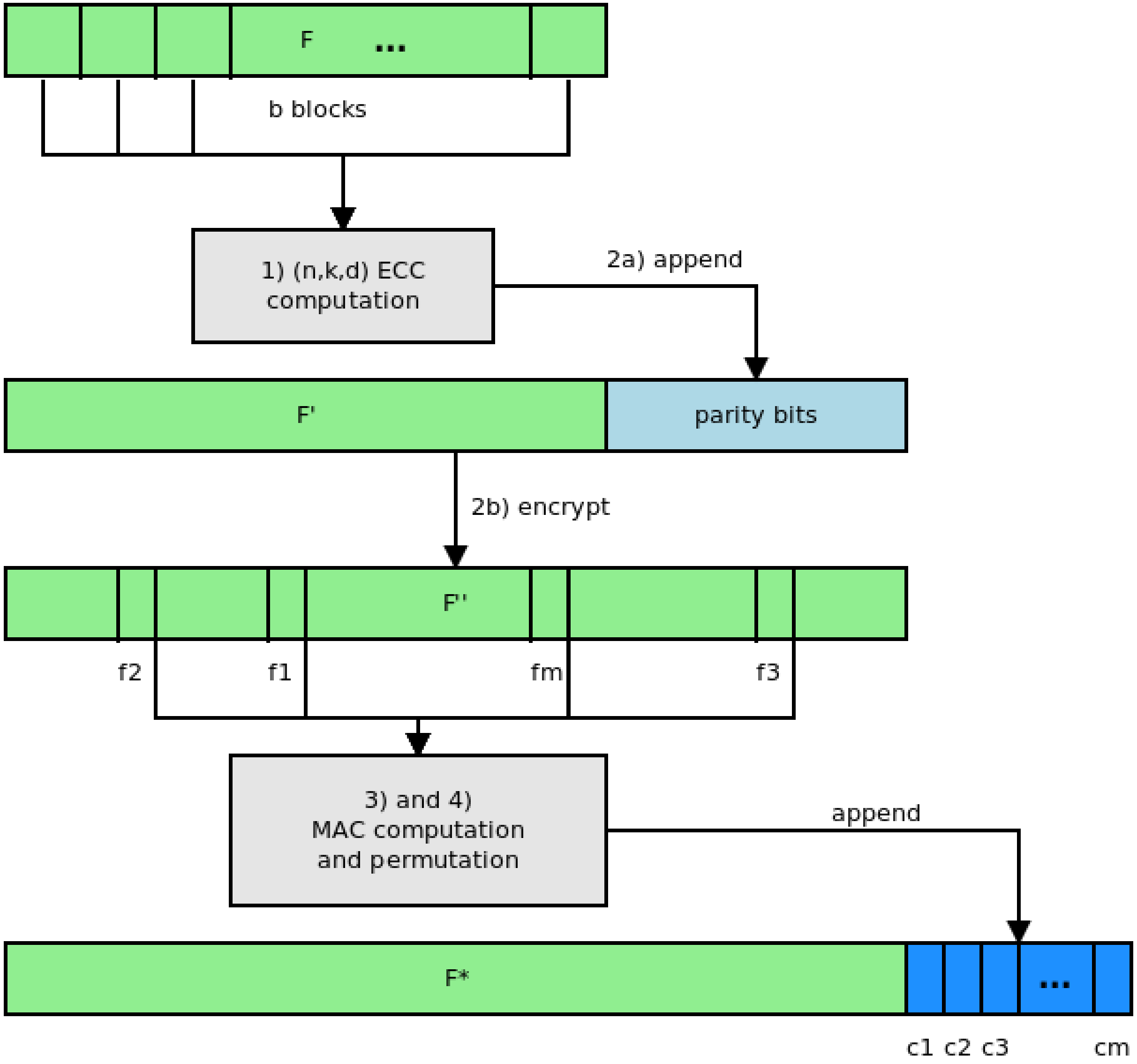
Data integrity building blocks

- Hash functions (MD5, SHA-1, SHA-256)
- Message authentication code (MAC) – integrity and authenticity
- Error correcting code (ECC) – corruption detection and correction

Popular approaches

- MD5/SHA-1 software package checksumming
- integrity checksums of messages in networking
- Widespread use of ECCs in hardware solutions

Proof of Retrievability



Prepare for data validation

- divide a file F into b blocks and apply ECCs
- encrypt the file with appended ECCs
- select m blocks out of M, compute their MACs and append to the file

Perform data validation

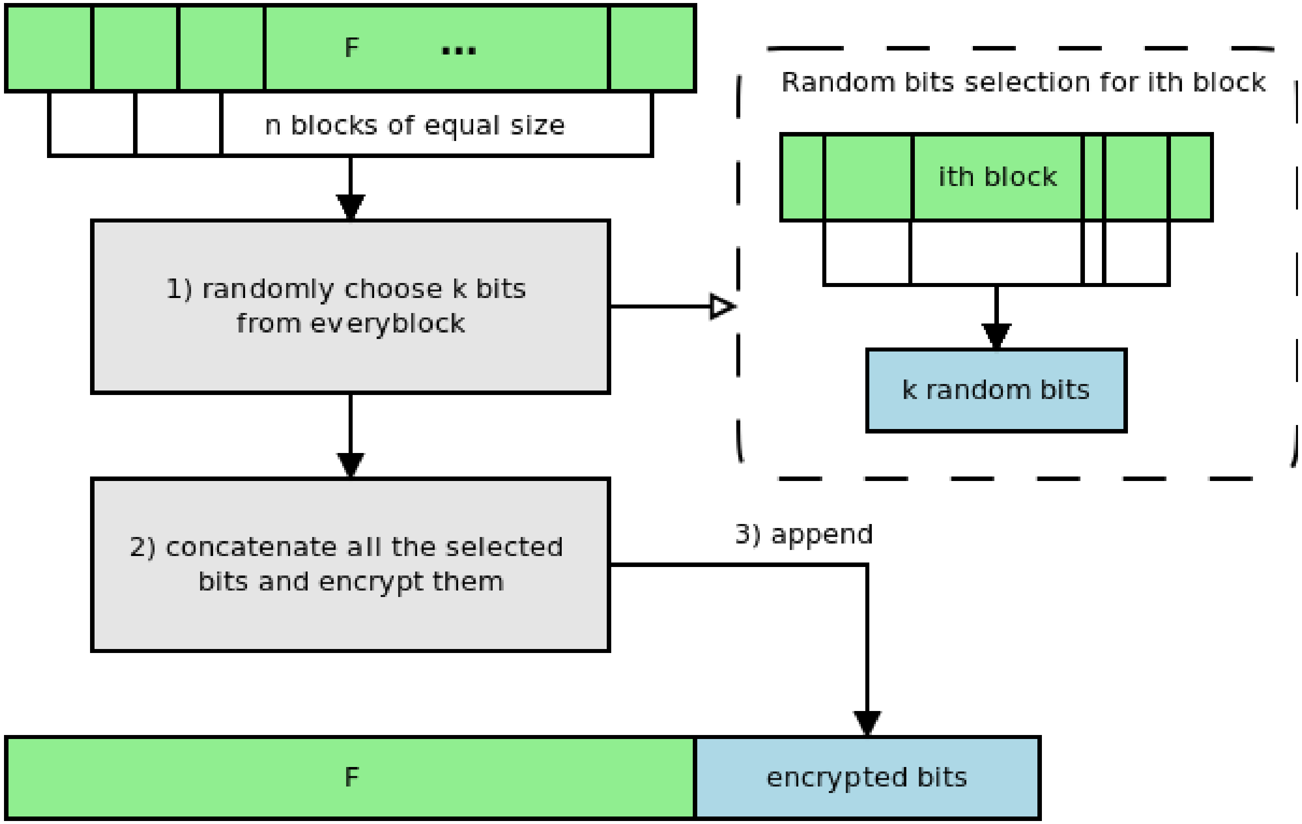
- choose k out of m blocks and download their content
- compute MACs and compare with the originals
- when retrieving whole file, apply ECCs

Drawbacks of the approach

- modification of file F
- ECCs storage overhead
- for network-efficient validation, requires computing capabilities on the prover side

A. Juels and B. Kaliski: PORs: Proofs of Retrievability for Large Files, ACM CCS, 2007

Data integrity proof


F ...
n blocks of equal size
1) randomly choose k bits from everyblock
Random bits selection for ith block
ith block
k random bits
2) concatenate all the selected bits and encrypt them
3) append
F
encrypted bits

Prepare for data validation

- divide a file F into n blocks and select randomly k bits from every block using key generator
- concatenate all selected bits and encrypt them
- append encrypted bits to the end of file

Perform data validation

- select the same bits and concatenate
- compare with the originals

Drawbacks of the approach

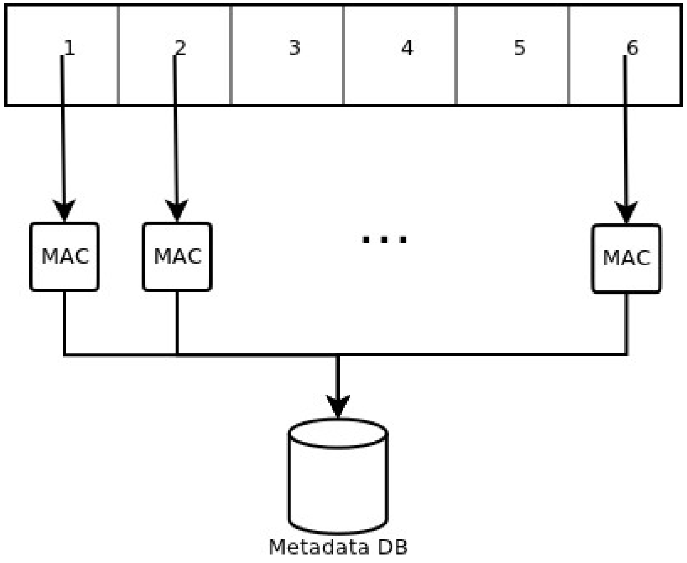
- Inefficient with regard to current cloud REST APIs, where every non-contiguous bit range requires separate HTTP request

S. Kumar and A. Saxena: Data integrity proofs in cloud storage, COMNETS, 2011

Data validation algorithm

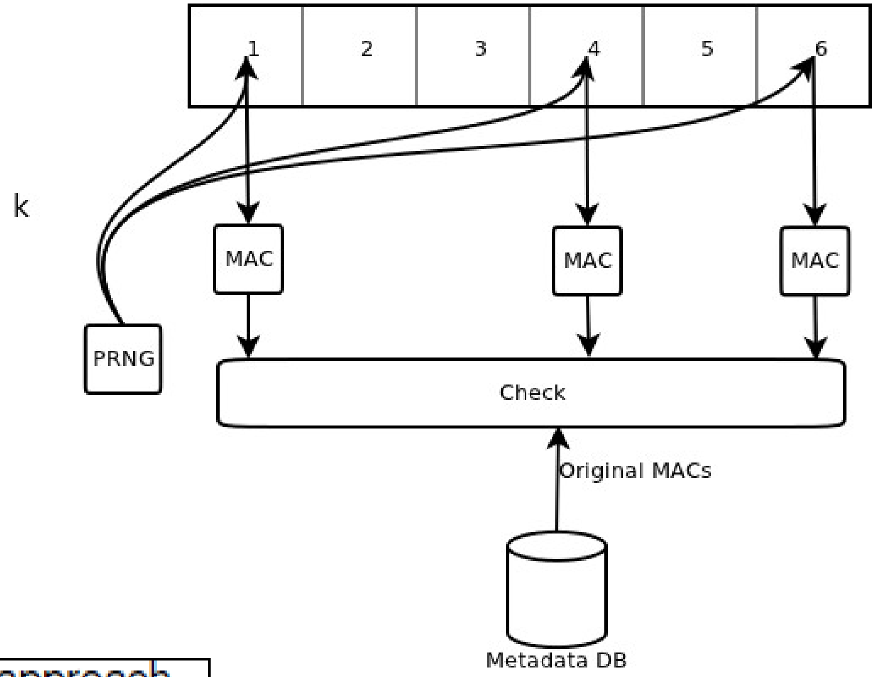
Prepare for data validation

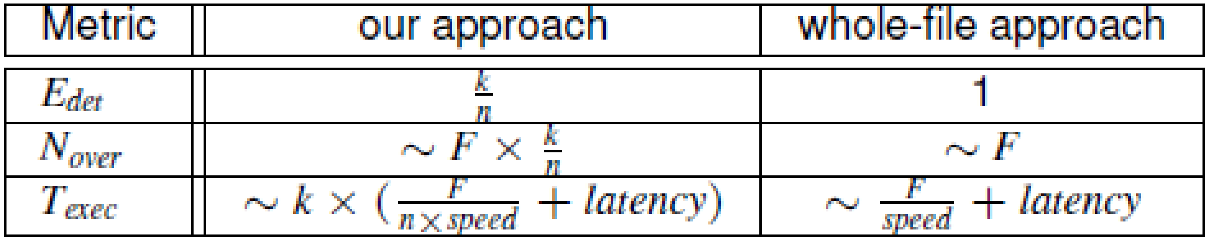
- divide file F into n equal chunks
- compute MAC checksum for every chunk and store



Perform data validation

- randomly select k out of n chunks
- compute MAC checksum of selected
- chunks and compare with the originals





Metric	our approach	whole-file approach
E_{det}	$\frac{k}{n}$	1
N_{over}	$\sim F \times \frac{k}{n}$	$\sim F$
T_{exec}	$\sim k \times (\frac{F}{n \times speed} + latency)$	$\sim \frac{F}{speed} + latency$

$E_{det.}$ – *error detection rate*

$N_{over.}$ – *network transfer overhead*

T_{exec} – *time complexity*

F – *file sizze*

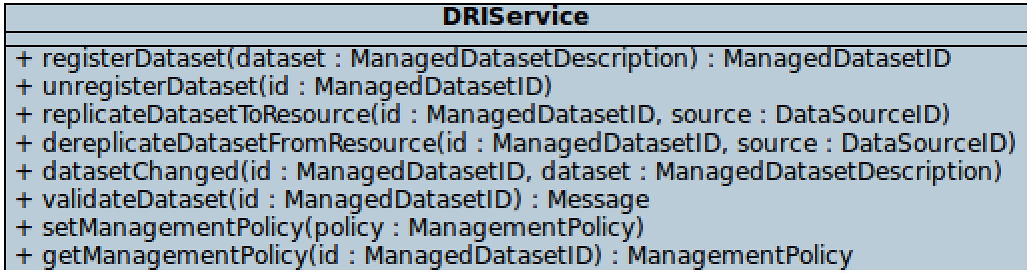
Design of Data Reliability and Integrity (DRI)

DRI service

- stateless REST web service in VPH-Share cloud environment
- periodical and on-request probabilistic data validation of data in federation of cloud storages
- datasets and validation metadata stored in AIR registry
- notifying scientific users via Notification service
- asynchronous calls and batch execution

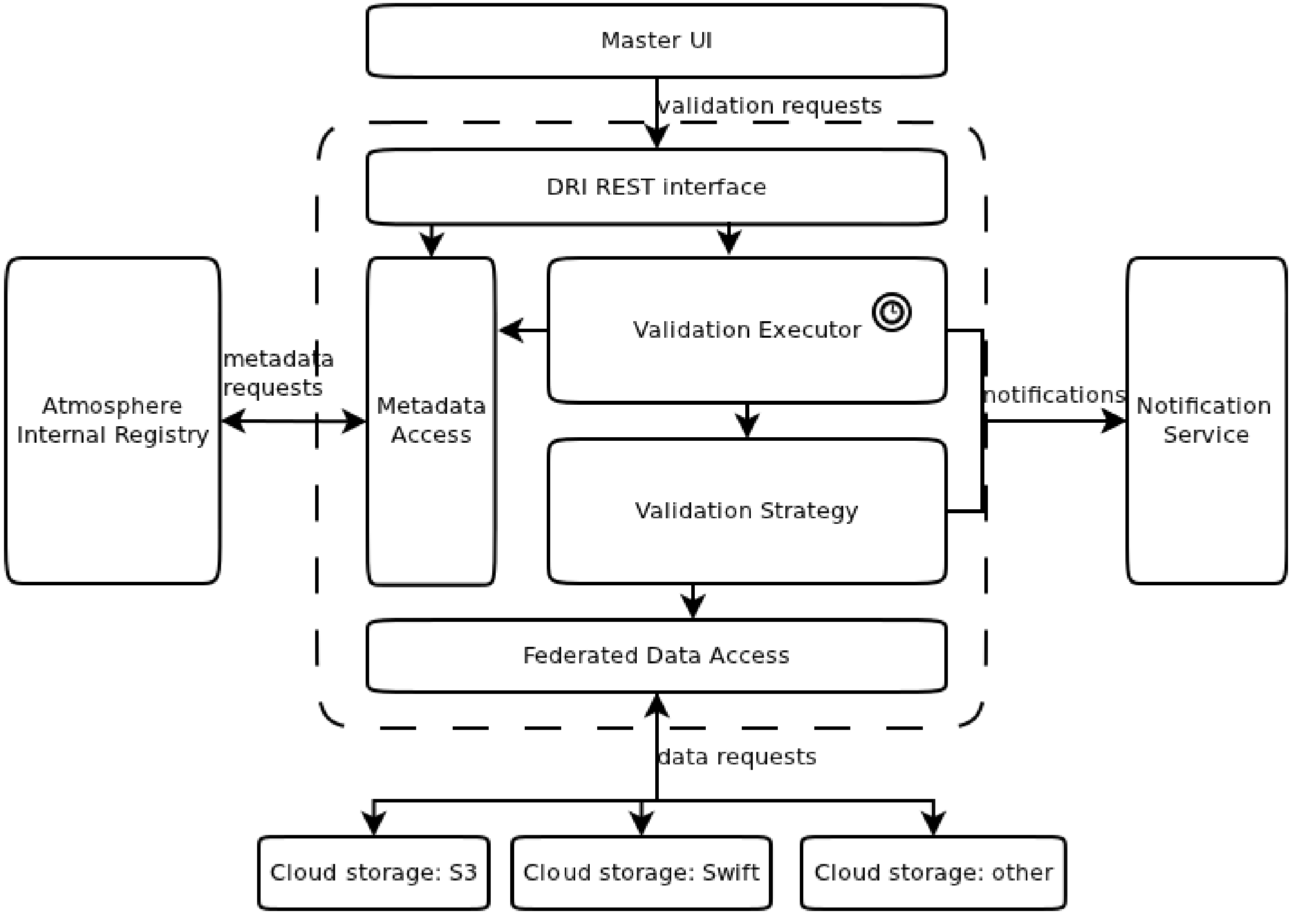
Implementation technologies

- JClouds library – generic cloud storage abstraction
- Quartz – task scheduling
- JAX-RS – REST web service
- Java, Guice, Guava, Tomcat



DRIService

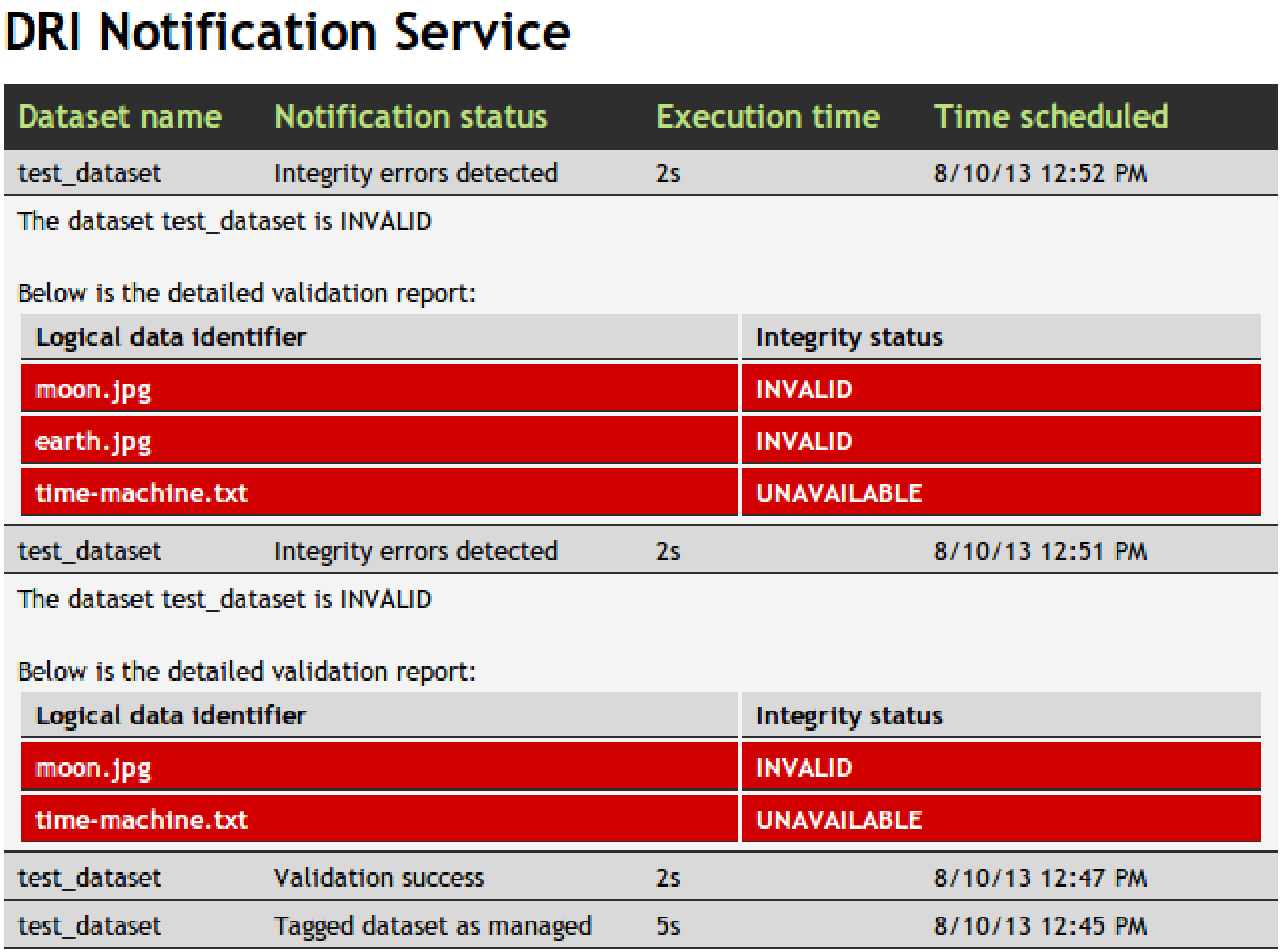
```
+ registerDataset(dataset : ManagedDatasetDescription) : ManagedDatasetID
+ unregisterDataset(id : ManagedDatasetID)
+ replicateDatasetToResource(id : ManagedDatasetID, source : DataSourceID)
+ dereplicateDatasetFromResource(id : ManagedDatasetID, source : DataSourceID)
+ datasetChanged(id : ManagedDatasetID, dataset : ManagedDatasetDescription)
+ validateDataset(id : ManagedDatasetID) : Message
+ setManagementPolicy(policy : ManagementPolicy)
+ getManagementPolicy(id : ManagedDatasetID) : ManagementPolicy
```



Example of DRI operation

Notification service

- VPH-Share mock
- shows DRI results
- displays detailed information about files that are unavailable or corrupted



DRI Notification Service

Dataset name	Notification status	Execution time	Time scheduled
test_dataset	Integrity errors detected	2s	8/10/13 12:52 PM

The dataset test_dataset is INVALID

Below is the detailed validation report:

Logical data identifier	Integrity status
moon.jpg	INVALID
earth.jpg	INVALID
time-machine.txt	UNAVAILABLE

Dataset name	Notification status	Execution time	Time scheduled
test_dataset	Integrity errors detected	2s	8/10/13 12:51 PM

The dataset test_dataset is INVALID

Below is the detailed validation report:

Logical data identifier	Integrity status
moon.jpg	INVALID
time-machine.txt	UNAVAILABLE

Dataset name	Notification status	Execution time	Time scheduled
test_dataset	Validation success	2s	8/10/13 12:47 PM
test_dataset	Tagged dataset as managed	5s	8/10/13 12:45 PM

Summary and future work

Results

- proposed a network-efficient algorithm for data validation in the cloud
- proposed methodology how to address the problem of providing data reliability and integrity in the cloud
- enabled VPH-Share project users to monitor data integrity and notify in case of failures

Future work

- investigate how to combine DRI monitoring service with federated cloud storage data access layer
- extract DRI functionality and provide it as a reusable component outside of the VPH-Share project
- investigate further improvements of data validation algorithm

Acknowledgement

More at http://dice.cyfronet.pl/VPH-Share

This thesis was realized partially in the framework of the following projects:

Virtual Physiological Human: Sharing for Healthcare (VPH-Share) – partially funded by the European Commission under the Information Communication Technologies Programme (contract number 269978)