DPHIANT Peer Testing Report #2

TMI Project / Team 0

Ben Wilson | Nigam Lad | Yidu Guo

DICOManon / “DPHIANT” / “DICOM Personal Health Information Anonymization Tool”

What is DPHIANT?

DPHIANT is a data pipeline that runs on Orthanc, a free and open-source, lightweight DICOM server for medical imaging. DPHIANT is used to anonymize DICOM files.

As of the last milestone,

Our team was been able to create our system within a docker container with three separate instances of Orthanc. Modality instances simulate medical imaging machines, middleman instances that will run core functions such as anonymization and crosswalk table, and the last instances represent deep storage where anonymized DICOM files are stored for future retrieval. Foundational functions such as uploading, transferring, and querying DICOM files between Orthanc instances were implemented at the last milestone.

Since the last milestone,

The focus of our team has been to develop functions that are essential to the client’s desired end product. At the current stage of development, we were able to successfully implement a number of core functions that greatly contribute toward the completeness of our framework.

One of the core functions that was successfully implemented was anonymization. Given a non-anonymized DICOM file from a patient that was generated from an imaging machine, the raw DICOM file will first appear in Orthanc modality. Information such as patient name, sex, and date of birth are categorized by tags in the metadata of the DICOM file. The DICOM file will be then pushed to the middleman and passed onto PACS for storage. During this process, select personal information is stripped from the DICOM file and replaced by aliases. The alias is implemented to be associated with the patient’s sex, such that male patients will be assigned a male alias, and female patients will be assigned a female alias. The set of information to be stripped and replaced is determined by the client and is easily modifiable for future changes.

A modification was made to the flow of data pertaining to how the middleman sends files to PACS. Previously, the middleman would save an instance, modify it, then send it to PACS. The modified version now has the middleman pushing the instance straight to PACS without modification. It then proceeds to modify that instance which is already living in PACS. The reason for this change is to ensure that in the event of server failures, downtimes, or other issues, the data is not lost within the middleman. During recovery, the middleman has the ability to continue its unfinished processes, but in the event of a fatal error, the data may be lost. Instead, if the DICOM image is pushed immediately to deep storage, and then the system crashes, the middleman can continue to modify the file as needed right from PACS since the data there is persistent. In the grand scheme of our data flow, this is not such a major change that it would affect how our pipeline functions or future changes that may add to the pipeline flow.

Another function that was successfully implemented was patient matching. When a DICOM file of a patient is pushed to the middleman, we implement a database with MongoDB to keep track of the information that has been stripped, as well as the aliases assigned to that patient. For every DICOM file being pushed to the middleman, the patient is checked against the database. New patients will have brand new aliases assigned to the anonymized fields, existing patients will be given previously assigned aliases. The end result is that DICOM files under the same patient will always be given the same alias after the anonymization process.

With the successful implementation of the crosswalk table for patient matching, additional features can also be implemented. Since DICOM files under the same patient will always be given the same alias, in deep storage, DICOM files of the same patient will now be listed under the patient's alias instead of creating a new instance.

In summary,

Below is a list of functions and features that was available during this testing:

- Anonymization of DICOM file
    - Viewing and querying the DICOM instances in a browser interface for all Orthanc instances
    - Removal of desired PHI (Personal Health Information)
    - Assignment of randomized aliases
- Implementation of crosswalk table
    - Patient matching
    - DICOM stacking under same patient alias
    - Viewing and querying the database in a browser interface

Issues Discovered

During this round of development, we were fortunate to come across an extremely minuscule amount of issues with our code. Testing has proven that our pipeline is extremely robust and able to handle large influxes of data without sustaining damages to said data or disruptions in transfer speeds. We have yet to discover any issues that would break the flow of the pipeline, however, we have found a small issue with some characters being lost in translation. The only instance we have observed this issue is while uploading select DICOM files and sending them through the pipeline and observing the data that is produced in the crosswalk table. Upon the PHI being stripped from the DICOM file and stored in the MongoDB database, namely the crosswalk table, occasionally a patient's true name will not be copied over correctly and some of the characters may be mismatched or in a completely different language. The team has yet to determine the cause for this issue, but we have determined that it is in no way a catastrophic issue. This issue will be solved during Milestone 4 and was catgeogorized as a Low-level issue.

Setup

Download the project package from our GitHub repository

1. To start the setup, type "docker-compose up --build in your project path". This will begin to install the various images and dependencies required to run Orthanc servers via Docker

2. After the Orthanc server has started up, it will ask for login information, use the following to log in.

   LOGIN/PASSWORD = demo/demo

3. Each instance of Orthanc can be accessed at the following localhost address:
   a. Orthanc Modality simulation: http://localhost:8044/
   b. Orthanc Middleman simulation: http://localhost:8043/
   c. Orthanc PACS simulation: http://localhost:8042/

4. The MongoDB database can be viewed via the Mongo Express web interface at http://localhost:8081/

5. Open Orthanc Modality instance and select 'send to modality', then select middleman as the destination.

6. The middleman instance serves as a buffer to push images to PACS, and then modify the images for de-identification.

7. Open Mongo Express (http://localhost:8081/db/PHICrossTable/PatientHealthInformation) to view the original patient health information and its given alias info

8. DICOM is available at the Orthanc PACS instance, where users can view the de-identified DICOM file presenting only the alias information.