Biometer: Fault Tolerant Healthcare Aggregator

CMU 18-749 Fall 2015 Class Project

Mikhail Kutsovsky
Dept. of Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, PA, USA
mkutsovs@andrew.cmu.edu

Joshua Antonson
Dept. of Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, PA, USA
jantonso@andrew.cmu.edu

Aayush Agarwal
Dept. of Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, PA, USA
aayusha@andrew.cmu.edu

ABSTRACT

As individuals get older and spend time in various places around the country, the necessity to visit different doctors, for different ailments all located in different locations becomes an inevitability. Each visit leaves a fragment of that individuals health record behind, often times to never to be seen again. The medical records and other relevant information accessible to patients used in electronic personal health record systems (PHRs), which are the cornerstones of patient centered healthcare, are incomplete (in a possibly critical fashion) due to this fragmentation and cannot optimally assist patients through health self-management. This paper proposes a solution to the difficulty of gathering this fragmented information through an automated scraping process that gathers information from various sources in one place. This compiled medical history can be leveraged to provide greater benefits than any stand-alone systems due to the completeness of the information.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Personal Health Record. Fault Tolerant Distributed System. Web Scraping. Databases. User Experience

1. INTRODUCTION

The trend of hospitals and medical providers moving towards electronic medical records as well as the prevalent use and availability of medical information online is setting up a situation where patients have a lot more control over their health care and treatment. Studies have shows that 42 percent of the US population keeps records for themselves, 87 percent of these being in paper format[3] and that doing this is a very good idea. Many trials of electronic personal health record systems (PHRs) have shown that they supplement and improve patient and family access to knowledge for self-management of health and wellness issues [2]. With all of these clear benefits to having PHRs and multiple initiatives from medical providers to offer Personal Health Record systems with their care how come it is still so difficult for the general population to get access to their documents from various providers and ensure that all the doctors they see have a unified view of their health situation. We take the approach that a PHR that is based on the provider giving the information as opposed to the patient seeking out the information is inherently flaws. Due to the variety of websites, incompatibility of PHR systems across providers and the large amount of health information which will not be accessible in this manner, relying on the provid. Since we cannot rely on providers to have all the information needed for a complete medical history available in a easily accessible manner, we have built Biometer to be a fault-tolerant application which can perform scraping operations against various medical providers on behalf of the user. The capability of going out and proactively reaching out to all the medical providers an individual has had contact with will result in a much more comprehensive medical record that will benefit care in every aspect.

2. PROBLEM STATEMENT

Collecting information from hospitals and medical providers is an unexpectedly difficult task for a person to do. Jumping through the various process requirements of these institutions before actually getting ones information is a major deterrent for people. To fix this problem we need to build a system that can automatically and reliably get the desired medical records with minimal input from users. The manners in which we can get user medical records from a providers is A) User already has and uploads B) Automated Scraping of the providers patient portal C)Sending Authorization Forms. This system also needs to continue operating under faults and allow user's uninterrupted access and use of this information.

2.1 Goals

1. Usability: Development of an application where users can create an account, get medical information from supported medical providers via automated web scraping after providing credentials and the capability to then view these records after the fact
2. Fault Tolerance: Building a system that is fault tolerant to server's going down in the back-end. Failures should be handled and not effect the user's experience.

2.2 Non-goals

There are a number of issues that need to be addressed before Biometer is ready for public use but they are left for future work. They involve the following:

1. We do not aim to support gathering medical records from every hospital or medical provider, just a select

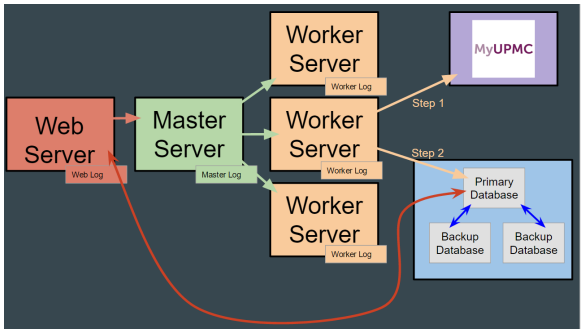

Figure 1: The Biometer 4-Layer Architecture.

few via scraping their patient portal. Further hospitals can be supported by automating the Authorization From process

2. Security and Privacy are major issues with many Personal Health Record systems. Topics in which we are not experts and leave that for future work.

3. APPROACH

We split our work into two major parts: the application which runs the web client and scraping of medical records and the fault tolerance aspect of the system. Careful thought was given to the initial design of Biometer to make it as stateless as possible requiring fault tolerance in a couple key areas.

3.1 Architecture

As seen in Figure 1, we built Biometer with a four level architecture using a MEAN stack. Users connect to the application through a `stateless web portal`. This server runs Passport for authentication connecting to a `MongoDB` database. After going through the account creation process, users can then elect to add or view information from a hospital. All requests to retrieve information from a medical provider get sent to the `Master Server` over HTTP. The master server contains the state of the system and coordinates routing to worker servers. The Master Server also keeps track of which worker server's are available and distributes the workload evenly. The `worker servers` are stateless machines that can be spun up or down based on the current workload. They receive requests to Scrape a particular medical provider's patient portal (with accompanying user credentials) and save the results to the database.

3.2 Fault Tolerance

The Biometer system would ideally have fault tolerance built in at every level of the architecture. The front-end web server contains no state so server crashes here only act as a minor inconvenience to the user. They can just refresh and get connected to a different web server and resume browsing with no loss of data. The Master server is the single point of failure in the application and we would like to passively replicate this server in the future. The Master server is responsible for handling retry logic for any worker server failures. Through the use of asynchronous socket communication between master and worker server, the master is able to immediately detect as a worker goes down, check the list

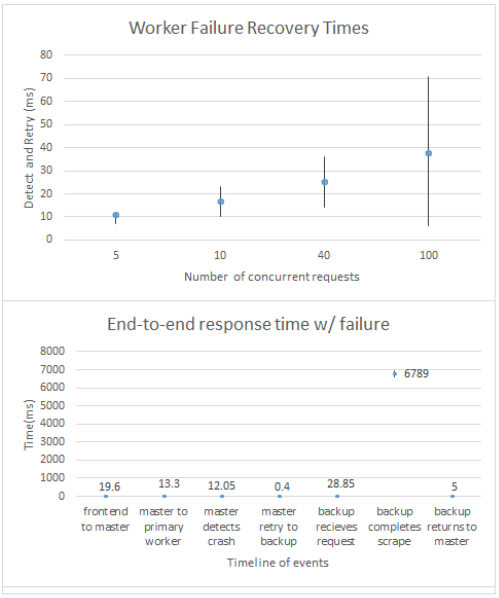

Figure 2: 12ms retry time on worker server failure (top image) and timeline for request w/ retry

of available workers, and redistribute the incomplete workload to backup worker servers. The MongoDB database is fault tolerant through the use of a primary and secondary replicas.

4. EXPERIMENTAL VALIDATION

We analyzed our system through bench marking performance, and recovery times on the University of Pittsburgh Medicine (UPMC) patient portal. We analyzed the system to determine recovery time in the presence of worker faults, as well for execution time of key operations during various workloads.

4.1 Experimental Setup

We deployed instances of Front-end, Master and Worker servers as well as our MongoDB instances to AWS on the free tier micro instances. We then simulated various workloads to scrape information from the UPMC account and inserted failures at worker and database layer to monitor the recovery time.

4.2 Experimental Results

Due to the use of socket connections to send asynchronous requests from master to worker servers, the fault detection and retry time is 12ms and linearly increasing w/ the number of connections open. The time needed to execute a scraping request scales with the number of concurrent requests being serviced by the particular worker with the baseline scrape taking on average 6.5s.

5. CONCLUSION

Biometer is the first step in building a fault tolerant scraping service for medical records. With additional functionality (Authorization Forms, Security and Scale), the approach

of taking user credentials and gathering the information on their behalf can enable easy one-click access to a complete medical history.

6. REFERENCES

[1] Hillestad, Richard and Bigelow, James and Bower, Anthony and Girosi, Frederico and Meili, Robin and Scobille, Richard and Taylor, Roger, *Can Electronic Medical Record Systems Transform Health Care? Potential Health Benefits, Savings, and Costs*, Health Affairs, 24, no.5 (2005):1103-1117

[2] Jones DA, Shipman JP, Plaut DA, et al. *Characteristics of personal health records: findings of the Medical Library Association/National Library of Medicine Joint Electronic Personal Health Record Task Force.* J Med Libr Assoc 2010;98:243âĂŞ9

[3] Taylor H, *Two in five adults keep personal or family health records and almost everybody thinks this is a good idea*, Health Care News, 2004

[4] Archer, N., Fevrier-Thomas, U., Lokker, C., McKibbon, K. A. and Straus, S. E. *Personal health records: a scoping review*, Journal of the American Medical Informatics AssociationâĂŕ: JAMIA, 18(4), (2011) 515âĂŞ522.