Speeding up Children Reunification in Disaster Scenarios via Serverless Computing

Extended Abstract

Kyle Coleman
Computer Science Department
Saint Louis University, St. Louis, MO
kyle.coleman@slu.edu

Flavio Esposito
Computer Science Department
Saint Louis University, St. Louis, MO
flavio.esposito@slu.edu

Rachel Charney
Department of Pediatrics
Saint Louis University, St. Louis, MO
rachel.charney@health.slu.edu

ABSTRACT

Children constitute a vulnerable population and special considerations are necessary in order to provide proper care for them during disasters. After disasters such as Hurricane Katrina, the rapid identification and protection of separated children and their reunification with legal guardians is necessary to minimize secondary injuries (*i.e.*, physical and sexual abuse, neglect and abduction). At Camp Gruber, an Oklahoma shelter for Louisianan's displaced by Hurricane Katrina, 70% of the children were with their legal guardian after 2 weeks while the last child was reunified after 6 months.

In this project, we are using serverless computing to scale and minimize database queries as well as to speed up machine learning tasks for rapid reunifying time, in support of a federated set of first-responders. In particular, we are using a Flask-based web system that leverages IBM OpenWhisk to run both (face and text) profile recognition software at the back-end.

1 INTRODUCTION

Large scale natural disasters such as Hurricane Irma result in insurmountable damage to structures and devastating chaos within families. Many children are separated from their families and reunification can be a time and resource- consuming challenge. Distressed, young or injured children often cannot self-identify. Examples of text features used for recognition by first-responders are name, home address, and social security number. Matching this information manually (when available) can prolong the reunification time. To solve this problem, we are implementing a system empowered by serverless computing for fast, scalable and federated profile matching.

In serverless computing, the application logic is split into functions called by external or internal events. The actions in our case are defined by machine learning (classification) operations. In particular, the system goal is to match text and visual profile information uploaded by guardians with children's profiles available on the distributed database. Our events instead are database queries executed by first-responder devices.

WOSC'17, December 2017, Las Vegas, Nevada USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

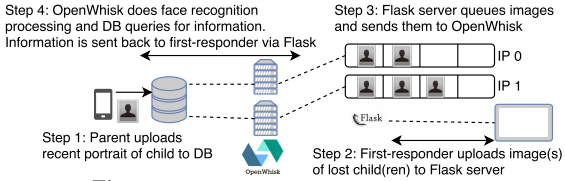

Figure 1: System Architecture and workflow

We picked serverless computing so that our system would inherit the benefits of an idle but responsive platform in case of bursty demands. Our prototype utilizes specifically OpenWhisk [3].

In the next section we describe our system design and our ongoing work.

2 SYSTEM ARCHITECTURE

Our architecture utilizes serverless computing in conjunction with an efficient micro-framework for web development called Flask [2].

A user with a mobile device requests a match via a HTTP POST that uploads images to our Flask web server, which pre-processes by queuing the images based on the unique IP address of the sender.After the collected images are uploaded, they are forwarded to our OpenWhisk server. Here, we run a face recognition application, available at [1].

We are extending our system so that the application will compare both text and existing uploaded photos against a pre-existing collection of profiles. When a match is found, it will utilize the name attached to the pre-existing photo to perform a database query for relevant information regarding the child. The database will then return identifying information such as Date of Birth, Social Security Number, emergency contact information, current medications, existing health problems such as allergies, and medical history in an easily readable format. We will test our system using anonymized images and profiles available from the Disaster Preparedness Center at SSM Cardinal Glennon Children's Hospital.

Our system usage will be easily extendable to other types of missing scenarios, for example, adults, elderly or handicapped patients, or even pets. Finally, our system design will be general enough and a good proof-of-concept demonstrating how to apply serverless technology to a useful set of delay sensitive applications by means of machine learning tools.

REFERENCES

[1] Face Recognition 2017. Face Recognition. (2017). Retrieved September 14, 2017 from https://github.com/ageitgey/face_recognition
[2] Flask 2017. Flask. (2017). Retrieved September 14, 2017 from http://flask.pocoo.org
[3] OpenWhisk 2017. Apache OpenWhisk. (2017). Retrieved September 14, 2017 from https://openwhisk.incubator.apache.org