Project 4: Distributed Typosquatting Detector

Team Nickelodeon:

Keny Liang : kenny.liang.1@stonybrook.edu

Vasu Sharma : vasu.sharma@stonybrook.edu

Adam Tringali: adam.tringali@stonybrook.edu

Harsh Vig: harsh.vig@stonybrook.edu

Intro:

Our team Nickelodeon, is made up of four computer science undergraduates in their final year at Stony Brook University. Our project is to create a distributed typosquatting detector.We created an application that allows users to submit a url to scan for various typos based on the 5-typo generator model and web crawled each typo url. We retrieved the html content and image of each site to allow the user to view on a web dashboard for any malicious sites.

High level details about the architecture of your project (why did you choose a specific way of doing it, compared to all other possible ways):

For our web/master server, we used a node js framework, Express js. Node uses an event-driven,non blocking IO model that makes it lightweight and efficient. We used MongoDB to store the data of typosquatting requests and the html/images of these results retrieved by worker nodes. MongoDB is a NOSQL database that can be highly scaled resulting in high performance. Puppeteer was used to crawl the web. Puppeteer is a node js library thus allowing our project to be implemented with the same programming language and technology.

The distribution of load across team members (i.e. who did what?):

Kenny Liang - Lead Programmer with experience in Express js and Mongodb who worked on the master/worker node code and front end.
Adam Tringali - Programmer with experience in vanilla js and html who worked on the master node code and assisted with the 5-typo generator methods.
Harsh Vig - Programmer with experience in vanilla js and html who worked on the worker node code and assisted with the master node code.
Vasu Sharma - Programmer with experience in vanilla js and html who worked on the 5-typo generator methods

Instructions that we can follow to set up and test your project: (Ubuntu 16.04 machines)
Our project requires Node/Npm installed and a local MongoDB server on the master node server.

Install Node/NPM

1. curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
   * note installation of curl may be required first
2. sudo apt install -y nodejs
3. node --version

Installing a local MongoDB server on master node only (ubuntu 16.04 machine)

For more info
https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/

1. **Import the public key used by the package management system.**

   wget -qO - https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -

2. **Create a list file for MongoDB.**

   echo "deb [arch=amd64] https://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/4.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.2.list

3. **Reload local package database.**

   sudo apt-get update

4. **Install the MongoDB packages.**

   sudo apt-get install -y mongodb-org

5. **Start MongoDB.**

   sudo service mongod start

6. **Verify that MongoDB has started successfully**

   sudo service mongod status

7. **Change MongoDB configuration file at /etc/mongod.conf -**
   Set the bindIP to 0.0.0.0 to allow worker nodes to connect to the remote database

Missing packages

On ubuntu servers there will most likely be missing packages that is needed for puppeteer
Run the following command to retrieve packages

```
sudo apt-get install gconf-service libxext6 libxfixes3 libxi6
libxrandr2 libxrender1 libcairo2 libcups2 libdbus-1-3 libexpat1
libfontconfig1 libgcc1 libgconf-2-4 libgdk-pixbuf2.0-0
libglib2.0-0 libgtk-3-0 libnspr4 libpango-1.0-0
libpangocairo-1.0-0 libstdc++6 libx11-6 libx11-xcb1 libxcb1
libxcomposite1 libxcursor1 libxdamage1 libxss1 libxtst6
libappindicator1 libnss3 libasound2 libatk1.0-0 libc6
ca-certificates fonts-liberation lsb-release xdg-utils wget
```

Running the project

1. Retrieve the project code from github
    - https://github.com/kennliang/typosquatter.git
2. Change into the project directory
    - cd typosquatter
3. Install the required node module dependencies on both master/worker server
    - npm install
4. To run the master/web server on the master node
    - npm start
5. To run the worker server on worker servers (portnumber - the port to run the worker server)
    - node client.js portnumber

References to third-party content/code-snippets that you used in your project:

https://github.com/yujiosaka/headless-chrome-crawler/blob/master/docs/API.md

https://github.com/puppeteer/puppeteer

http://mongodb.github.io/node-mongodb-native/2.1/api/index.html