Student Project Final Report
Finding Trustable Peers using SNS
Daniel W Song(dws2127)

Introduction

DYSWIS is a P2P based automatic fault detection system. DYSWIS uses random peers in P2P to detect failure. However, peers in P2P system cannot totally trust other random peers in P2P. The goal of this project is to find trustable peers in P2P. In this project, we will use Facebook to find trustable peers of the user.

Choosing Trustable Friend

DYSWIS wants to pick trustable peers among all peers in P2P. We will find trustable peers using Facebook friend function. If the random peer is the Facebook friend of the user, we can say that the user can trust that random peer. However, there may be Facebook friends of the user that are not totally trustable. People tend to accept friend request even if they do not know that friend that well. We will analyze user's activities to choose trustable friends among Facebook friends.

There may be several methods to find trustable friends using user's activities. However, we will combine two simple methods to retrieve trustable friends list. First, we will look at the user's wall and retrieve the list of friends who wrote a post on the user's wall. Second, we will get the list of people who got post from the user. These two methods are simple but straightforward.

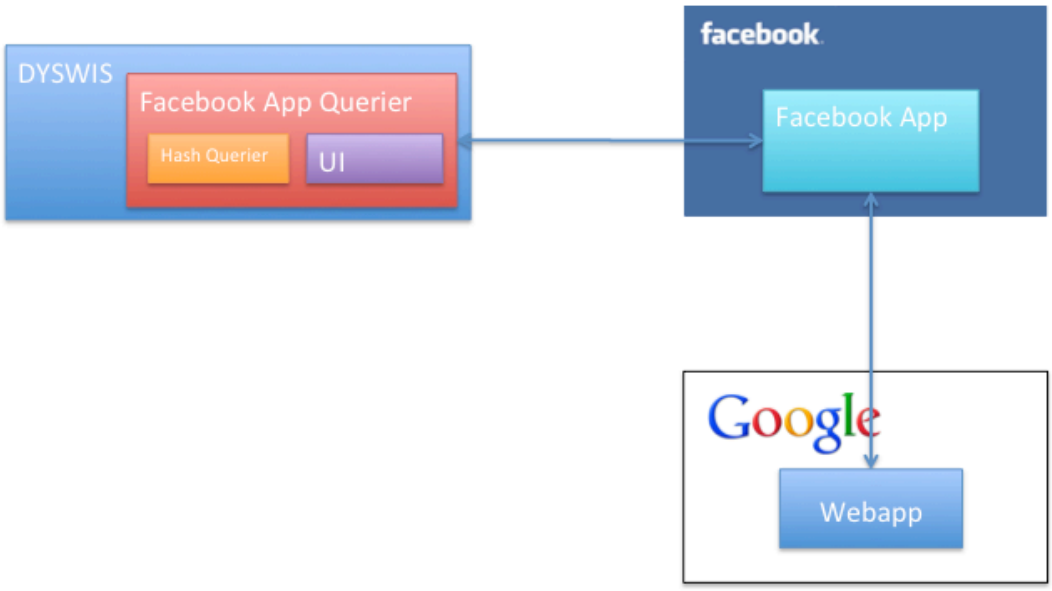
People who wrote a post on my wall + People who got my post on their wall

=

Trustable Friends

If the app just returns the friends list, there may be a privacy issue. We need some kind of mechanism so that some bad user cannot get other peoples ids. For this reason, when returning friends id, we will append a secret key to the id and hash it using MD5 so that no one can figure out other people's hash value. Every Facebook app has a secret key and we will append this key to id and hash this string.

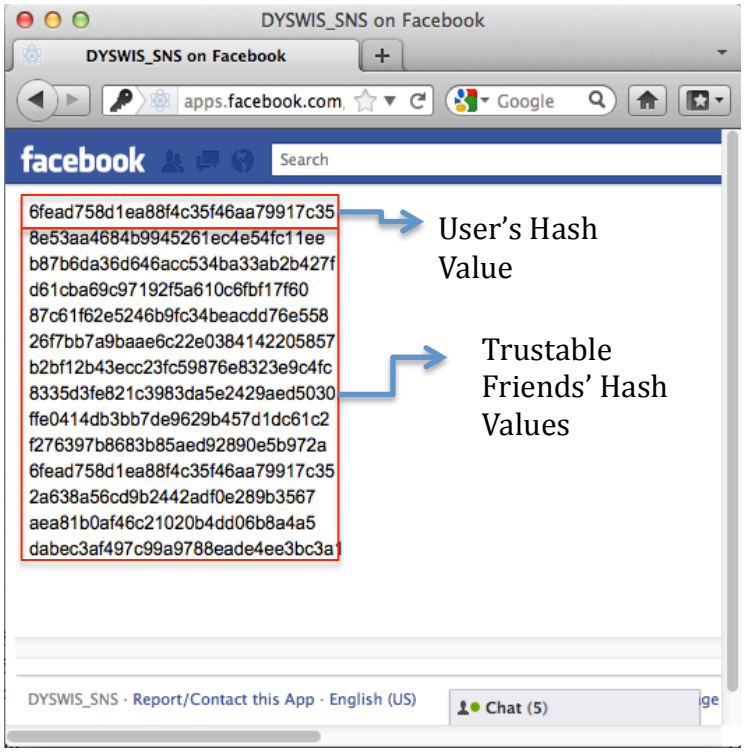
Architecture Overview



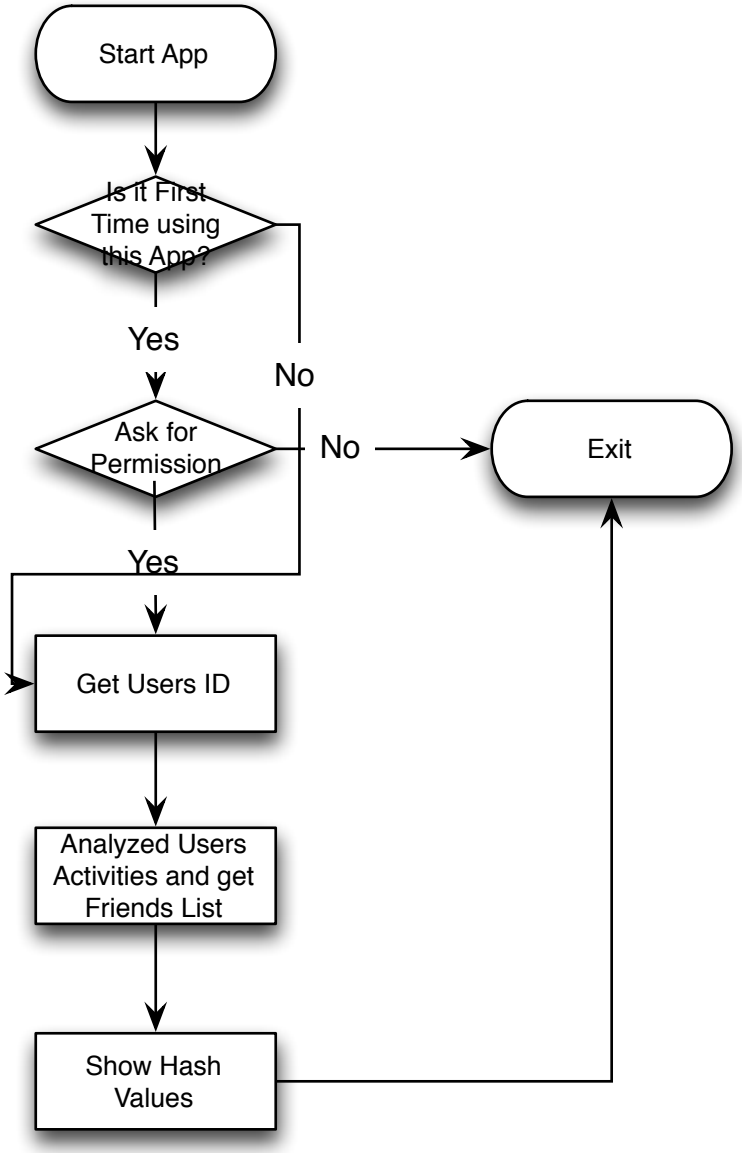
This project consists of two parts: DYSWIS module part, and web app part. The web app just shows the users hash value and trustable friends' hash values. DYSWIS module is integrated in DYSWIS, and this module runs the app and retrieves hash values. After the user gets hash values, it first registers user's hash value to the P2P. It uses hash value as the key and IP, port as the value. With this mechanism, the DYSWIS system can figure out trustable friends' IP and port using the hash value.

Facebook App

Facebook Web App is just a web app that is loaded in the context of Facebook. Actual web app runs on Google app engine. Java and Google Web Toolkit is used to make the web app. The app simply returns list of hash values. The first hash value is the hash value of the user. The rest are the hash values of trustable friends.



Facebook App Application Flow



Facebook App Application Details

First, if the user tries to run web app for the first time, Facebook asks user for permission. The app needs permission of the user to analyze user's activities. If the user allows, the app first tries to get Facebook id of the user. This Facebook id is a unique number that each user has. If the user gets Facebook id of the user, it appends secret app key to that id and makes hash value using MD5 algorithm. The secret app key that is added to the id is the secret key that every app has. After providing the app with user's hash value, the app now analyzes user's activities using Graph API. The Graph API presents users activities and it will be used to get trustable friends.

Graph API

The Graph API presents a simple, consistent view of the Facebook social graph, uniformly representing objects in the graph (e.g., people, photos, events, and pages) and the connections between them (e.g., friend relationships, shared

content, and photo tags). Using Graph API we can get user's activities. Also, Graph API is implemented using JSON.


```
{
  "data":
   [
    {
      "id": "507945336_10150580900430337",
      "from": {
         "name": "Young Jin Yoon",
         "id": "507945336"
    },
   "message": "Happy New Year!",
   "actions": [
     {
       "name": "Comment",
       "link":
"http://www.facebook.com/507945336/posts/10150580900430337"
     },
     {
       "name": "Like",
       "link":
"http://www.facebook.com/507945336/posts/10150580900430337"
     }
    ],
    "type": "status",
    "created_time": "2012-01-01T00:19:50+0000",
    "updated_time": "2012-01-01T00:19:50+0000",
    "likes": {
    "data": [
       {
         "name": "Yun Seong Nam",
         "id": "100000808241769"
       }
    ],
    "count": 3
   },
   "comments": {
     "count": 0
   }
},
……
```

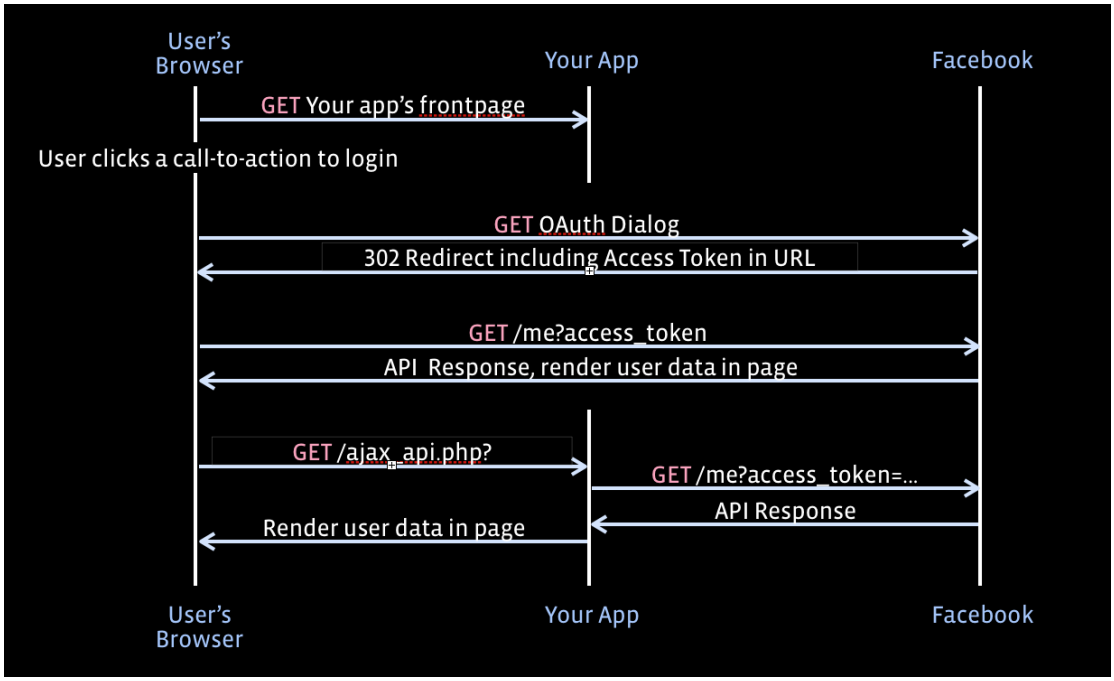

Object returned using Graph API

Authentication

This Facebook app uses OAuth 2.0 protocol for authentication and authorization. User authentication and app authorization are handled at the same time by redirecting the user to OAuth Dialog. When invoking a dialog we pass app id, redirect url, and permission code. This app needs users post activities so we request permission "read_stream".

```
https://www.facebook.com/dialog/oauth?client_id=112561958857640&redirect_uri=http://dyswis-sns.appspot.com/&response_type=token&scope=read_stream
```

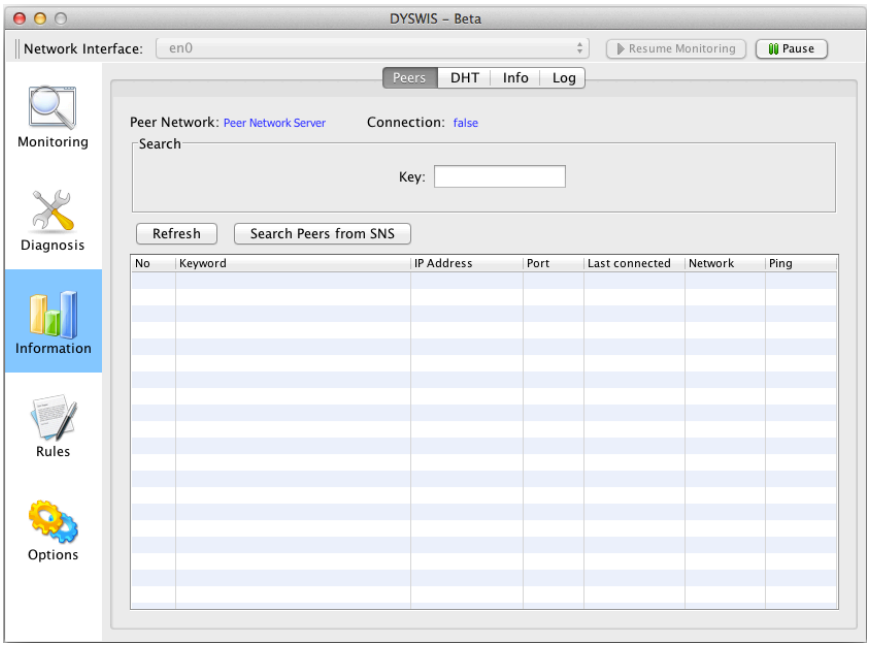
After user have logged in and gave permission to the app, it will redirect to http://dyswis-sns.appspot.com/. It will redirect to that URL with some token. This token is required to get users activities using Graph API. The app will pass token to Facebook and call Graph API to retrieve user's activities.



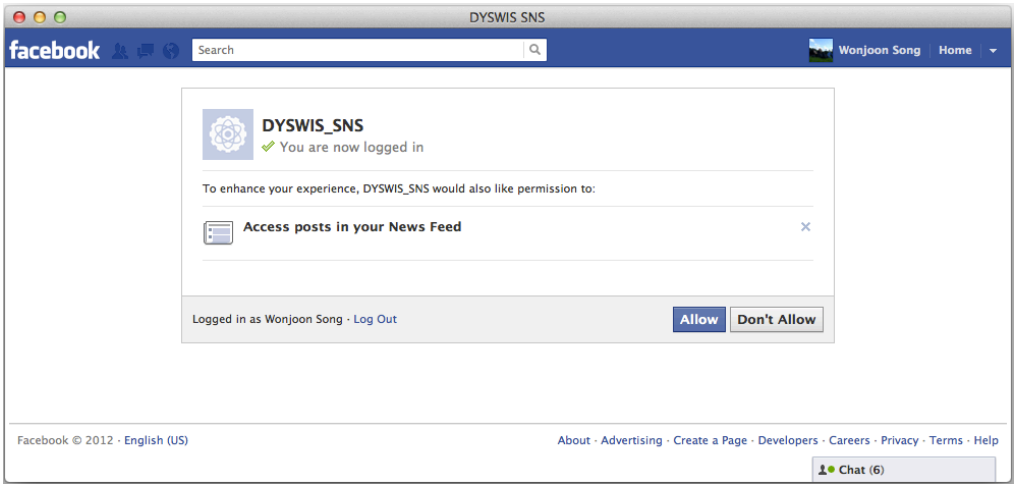
This diagram illustrates HTTP calls made through the client-side flow.

DYSWIS Module

When user clicks "Search Peers from SNS" button, DYSWIS module opens a SWT browser so that the user can login. After login, if it was the first time accessing the app, it will ask the user if he or she will allow the app to use users' information. After user allows the Facebook app to use user's information, the Facebook app will return hash values. Then DYSWIS will first register the users hash value to the P2P and get friends' IP addresses using trustable friends' hash values.

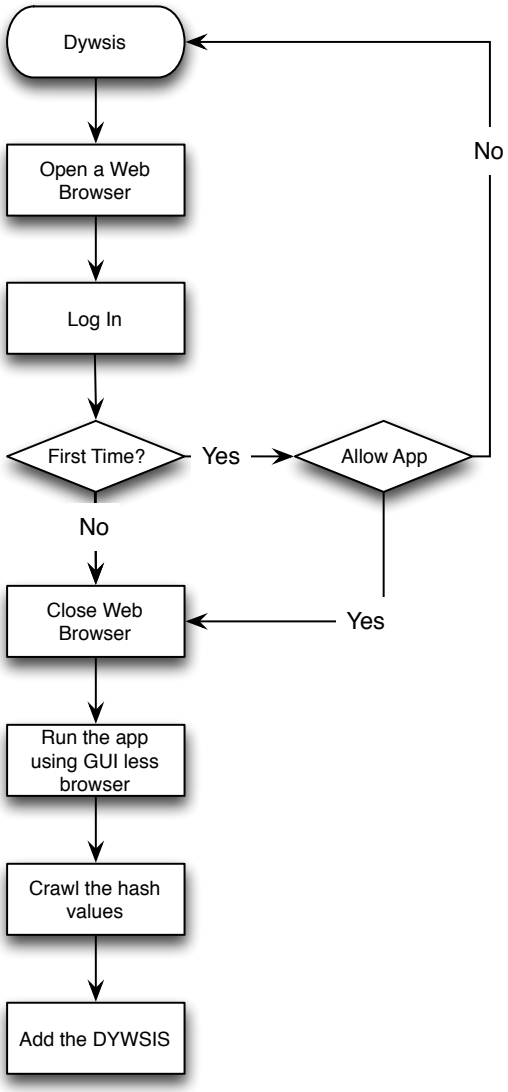


If the user clicks "Search Peers from SNS" it will open



App asking for permission

DYSWIS Module Application Flow



DYSWIS Module Application Details

When this module is called, it will first open a web browser so that the user can login and allow the app to use user's activities. It will use OAuth 2.0 for authentication and authorization. When user allows the app to use user's activities, it will be redirected to the http://dyswis-sns.appspot.com/ with some

token. Then the module will close the browser and GUI-less browser will crawl redirected URL site to get hash values.

File Added to DYSWIS System

edu.columbia.cs.dyswis.core.communicate.social.SocialNetwork class is added to DYSWIS system and three methods are used to integrate this module.

```
public boolean setMyHash(String hash)
```

DYSWIS uses this method to register the user's hash value to the system. It will register the hash value as the key and IP and port as the value. This method will eventually call `DysCore.communicateInstance.insertValue` to register hash value to the P2P.

```
public String getFriendIPAddress(String hash)
```

After DYSWIS gets list of trustable friends' hash values, we need to figure out their IP and port address. This method will eventually call `DysCore.communicateInstance.getValue` to get IP and port address of trustable friends.

```
public void addSNSPeer(String address)
```

After DYSWIS gets the IP and port address of trustable friends, it will add to the DYSWIS system using this method.

Issue, Problem, Future Work

After retrieving trustable friends' hash values, it will query P2P to get the IP and port of the trustable friends. However, there may be a problem with this mechanism. This is because trustable friends may not have installed DYSWIS. When we query 10 hash values, there can be 0 result. We need to add some kind of module so that the app only returns trustable friends that have installed DYSWIS.

Another problem is that trustable friend may not use trustable PC. If the PC has some kind of virus and if the user thinks friend using that PC as trustable friend, something bad such as DDOS attack can happen.

These two known problem is left for future work.

Reference

1) http://developers.facebook.com/docs/authentication/
2) http://developers.facebook.com/docs/reference/api/