Final Project in Software Engineering
Application Requirements Document

An information flow control model for online social network

Omer Sella
Sagiv Mapgavker
Alexander Chinyan

2018

Abstract

This document is the first of three documents describing the final project in the software engineering program. Specifically, this document describes the requirements of the software project.

This project is an research tool for ongoing study - on preventing information leakage based on the main articles "*An information flow control model for online social network based on user attribute credibility and connection strength factors*" and "*Generating a secure information flow in Online Social Networks using a Minimum Spanning Tree algorithm*".

The project will help the researchers to analyze true OSN data sets and to gather OSN new data sets in order to enrich the quality of their research.

The study is on behalf Ehud Gudes and Nadav Voloch.

Contents

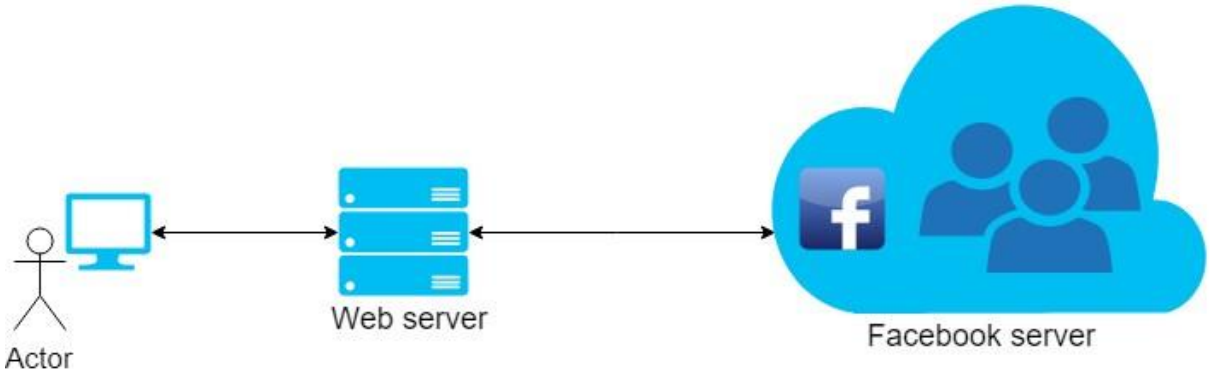
Chapter One

Introduction

1.1 The Problem Domain

Our project engage in data security of online social networks.
We wish to protect our privacy and try to prevent information leakage from users to unwanted entities, such as adversaries or spammers.
The project based on researches that were conducted about online social networks security.

1.2 Context



The system will be built as a website that is accessible via internet browser.
The program works over online social network, scans the necessary data and storage it temporarily in external server, the user may download the gathered data via CSV file.
Afterwards the server performs calculations and analyzes the data according to our main algorithm.
Another option is importing a CSV file that contains necessary data to be calculated.
The output is shown to the user on the screen as a multi optional graph.

1.3 Vision

Our first goal is to contribute to the research that was conducted. The project is based on three articles that have been published. Our main goal for this project is research purposes.
For future vision, it is possible that we will continue to develop the project and make to public.
This project may be the foundation to a public software that will help a lot of people to analyze their social network connection and to detect unwanted entities that may get an access to personal information.
The project will help people to make an informed choice based on our analyze.

1.4 Stakeholders

At this point the stakeholders are Ehud Gudes and Nadav Voloch, are professional guides that wrote the articles that our project is based on.
Our project will be used by them for research purposes.
We hope that the final product will serve their needs and contribute to the research work.
As we mentioned in the previous section, in the future when we will consider to expand and build a public tool the stakeholder will be updated.

1.5 Software Context

Our system is going to have an online and offline options.
The **dynamic** option will allow the user to connect to his social network to gather the necessary information on the fly by crawling his list of friends and for each friend in that list we create a list of users that could be a threat to the main user.
The user can export the gathered data to CSV file for future interaction with the system.
The **static** option will allow the user to upload CSV file that contains the necessary information and use the system to analyze it.
Another option is to build a graph manually.
Another option is to build a manual graph.

The system's major inputs:

- Suitable CSV file which contains all the data per OSN user that relevant for the processing phase, the CSV structure:

OSN user's id	**User's Name**	**TF- Total friends**	**MF- Mutual friend**	**AUA- Age of user account**	**FD- Friendship duration**	**Connecting friend id**
1	Alice	200	40	24	18	0
2	Bob	150	20	12	12	0
3	Charlie	300	10	14	12	1
4	Eve	20	5	1	2,9	1,2
...	...	...	...	...	...	...

In all next bullets user's friend refer to user's first and second circle's friends

- **OSN user's id -** user's friend's id given by the OSN (Alice has id 1)
- **User's Name-** Name of the user's friend
- **TF- Total friends-** user's friend's total amount of friend in the OSN (Alice has 200 friend in tested OSN)

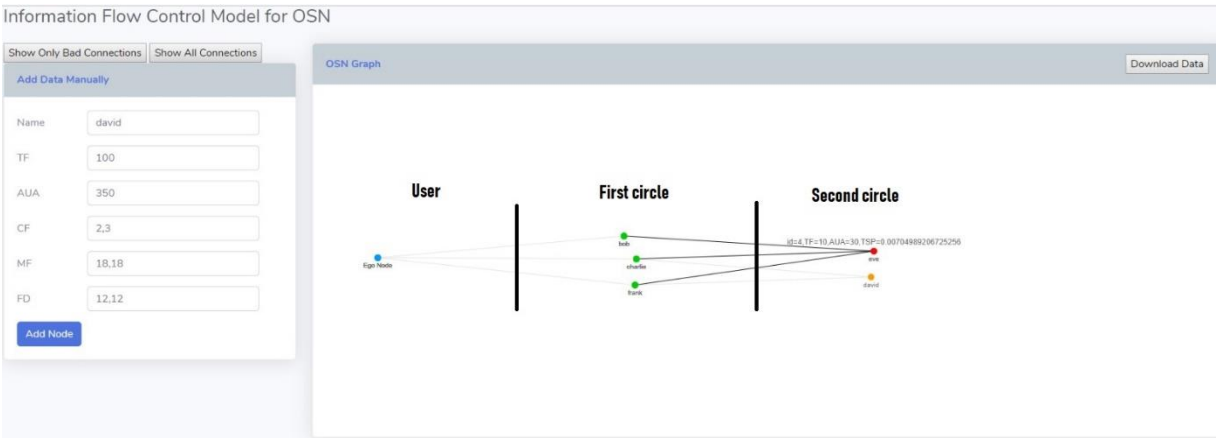
- **MF- Mutual friend-** amount of mutual friends of user and his friend (Alice has 40 friend in tested OSN)

- **AUA- Age of user account-** amount of months since user's friend join to tested OSN (Alice joined the tested OSN 24 months ago)
- **Connecting friend id-** 0 if it row representing user's direct friend, not 0 if it second circle's friend (can be multiple numbers - as the mutual number amount) - Alice is direct friend of the user, thus her "Connecting friend id" is 0. Eve is second circle's friend of the user and Alice and Bob are mutual friends (id 1 and id 2),

- **FD- Friendship duration-** friendship duration in month of user and his friend (first circle friend) or of user's friend and his friend (second circle friend) - Eve is Friend of Alice for 2 month and friend of Bob for 9 months. **FD- Friendship duration** order of values is relative to **Connecting friend id** order of values.

<u>The system's major outputs:</u>

- **<u>Information flow model graph-</u>**

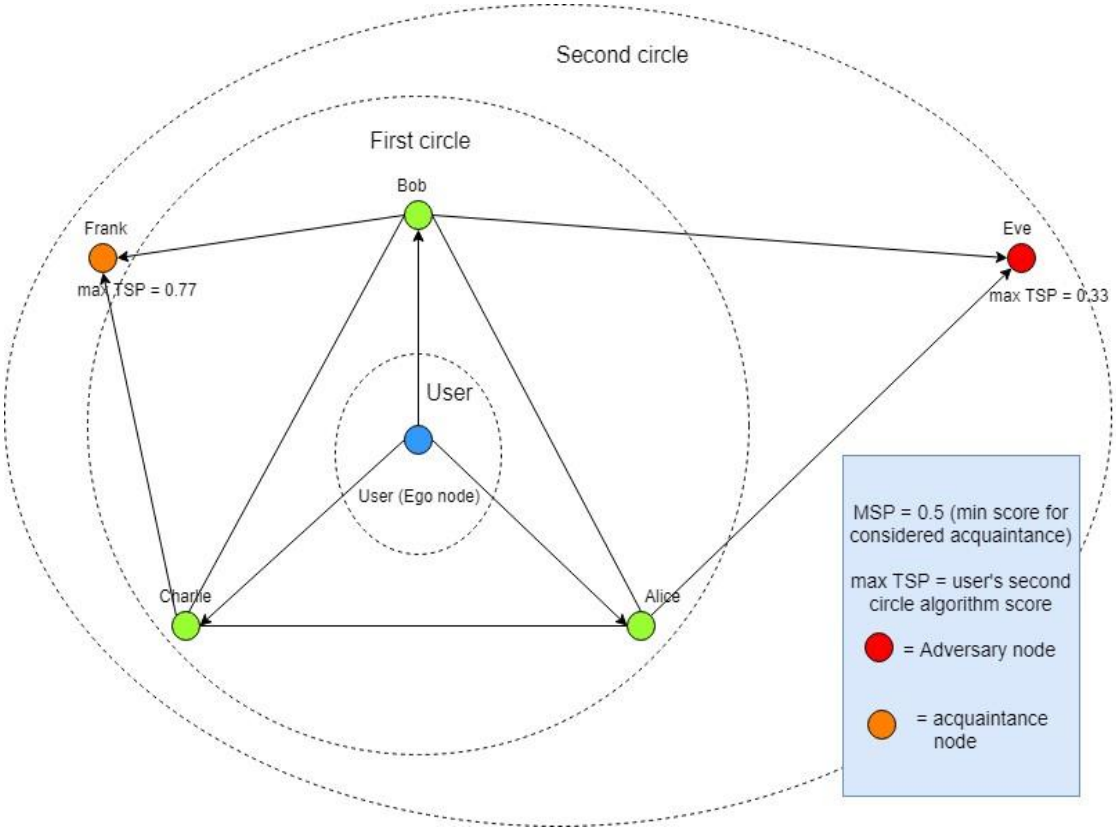


This graph represent the information flow that start at the system's user and demonstrate his OSN as 3 phased graph:

1. User phase
2. First circle phase - all direct friends of user
3. Second circle phase - all friends of his first circle's friends

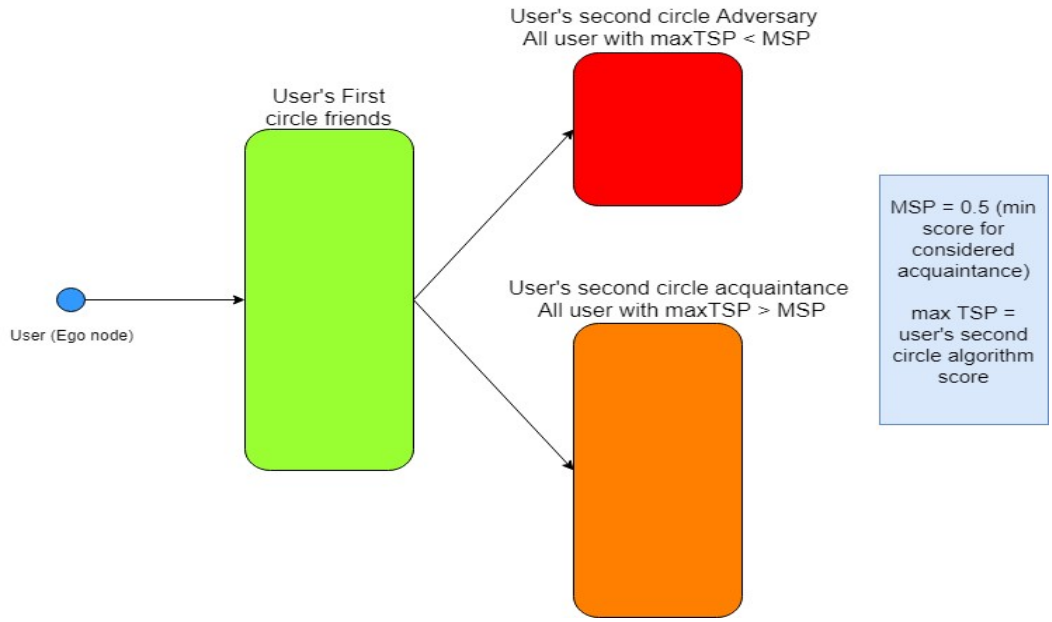
The information flow is from left to right, so unwanted OSN's users that we don't want to share user's information with are at most right identified as red node.

- **<u>Clique based graph-</u>**



This graph represent the information flow from the center out, supporting showing the user's cliques for better understanding his close friends (group of people that each know each other), and to understand if one of his clique is expose the whole group to adversary entity.
Moreover, in this graph module we can represent the algorithm score by presenting the node in relative distance from the circle.

- **Summary Graph-**



This graph can summarize all user's output from the algorithm that only classify his second circle friends to adversary or acquaintance entities.

In this module there are only 3 edges and 3 groups of nodes, which each group contains all the users that fit the algorithm output.

1. Green group - User's First circle friends
2. Red group - User's second circle Adversary
3. Orange group - User's second circle acquaintance

Processing:

The main functionality of our system:

1. Crawling the social network - collecting information from HTTP connection to social network using familiar modules for crawling and creating the CSV file. In this project we will mainly deal with these OSNs:
    a. **Facebook**
    b. Twitter
    c. Linkedin
2. Calculate CSV file - the system will analyze the given information via CSV file and creates weighted graph according to Gudes & Voloch's algorithms.
3. Presenting graph - multi optional graph based on the produced weighted graph.

Chapter 2

Usage Scenarios

2.1 User Profiles — The Actors

User of the system - can upload an CSV file with the necessary information , log in with his social network user and use the system to scan his friends(first circle) and the friends of his friends(second circle), can decide for the algorithm what parameters to include in the calculation, can choose filters for the output graph, can change the shown graph at the end of the process and he can export the data to CSV file.

Passive participants of the system - the system scans the social networks, therefore the first and the second circles of users are considered as passive participants of the system.

2.2 Use-cases

The main use-cases for our software system:

1. Log-in to the system with Facebook credentials:

Actors:
- User
- System

Precondition:
- The system is running on stable server
- The user connected to Internet, using HTML support browser and navigated to system's website via correct URL
- The user has a Facebook account

Postcondition:
- The user is connected to the system, delegated authority to the system to crawl his public data in Facebook

Main scenario:
1) The user press the "login to facebook" button
2) pop-up window appears to insert Facebook credentials.
3) The user is already logged in to Facebook
4) The user agrees to connected to the system via his Facebook account
5) The user connects to the system - Postcondition achieved

Alternative scenario:
1) The user press the "login to facebook" button.
2) The user is **not** logged in Facebook.
3) pop-up window appears to insert Facebook credentials
4) The user agrees to connected to the system via his Facebook account.
5) The system will show a login window.
6) The user will enter his facebook credentials.
7) The user connects to the system - Postcondition achieved.

More alternative scenario:
1) User entered invalid Facebook credentials while login to facebook - the system will present a suitable message and allow the user to re-enter the details.

2. <u>Upload CSV file which contains main algorithm input:</u>

Actors:
- User
- System

Precondition:
- The user navigated to system's website via correct URL.
- The user doesn't have to be logged in to the system.

Postcondition:
- The CSV file uploaded to the server side and ready to execute the system's main algorithm.

Main scenario:
1) The user press the "Upload CSV file" button.
2) The system will open a pop up window which the user can choose the file to upload from his computer.
3) The user choose from his computer which file to upload
4) The user press on "Upload" button.
5) The file transfer from the user's endpoint to the system's endpoint on the deployed server.
6) The system validated the file:
    - checks if it CSV file.
    - checks if it fit to the structure of the CSV file fit the algorithm input requirements.
    - checks if the file is malicious.
7) The system show an approval message to the user that the file is uploaded successfully.

Alternative scenario:
1) The user press the "Upload CSV file" button.
2) The system will open a pop up window which the user can choose the file to upload from his computer.
3) The user choose from his computer which file to upload.
4) The user press on "Upload" button.
5) The file transfer from the user's endpoint to the system's endpoint on the deployed server.
6) The system validated the file:
    - checks if it CSV file.
    - checks if it fit to the structure of the CSV file fit the algorithm input requirements.
    - checks if the file is malicious.
7) **The file uploaded is not valid**
8) The system will show a message that the file is not valid with explanation
9) The system will ask the user to re-upload the file

More alternative scenario:

1) The file upload process crashed while transferring - The system will show an error message and allow the user to re-try to upload the file
2) The file uploaded is considered as malicious - the system will disconnect the user from the system and ban his IP address.

3. <u>Running crawling on Facebook data of user to generate CSV file as input:</u>

Actors:

- User
- System
- Facebook's data

Precondition:

- The user logged in to the system via "login with Facebook" API

Postcondition:

- The system generate a CSV file according the user social network (Facebook) data:
    - Data on user's first circle friends - direct friends
    - Data on user's second circle friends - direct friends of the first circle
- The CSV file is fit to the main algorithm of the system (use case number 5)

Main scenario:

1) The user press on "Get Facebook data" button
2) The system start to crawl his social network data - (Use-case 3.1)
3) The system will save according all user's first circle friends these data in the CSV file:
    - Total number of friends (TF)
    - Age of user account (AUA)
    - Amount of mutual friends (MF)
    - Friendship duration (FD)
4) The system will save according user's second circle friends' public data in the CSV file:
    - Total number of friends (TF)
    - Age of user account (AUA)
    - Amount of mutual friends (MF)
    - Friendship durations (FD) with connecting friends

Alternative scenario:

1) The user press on "Get Facebook data" button
2) The system start to crawl his social network data - (Use-case 3.1)
3) **The crawling interrupted** in the middle of process
4) The system will ask the user if he want to re-run this use-case:
    a) If accept - the system will re-run
    b) If not - The system will continue with the already crawled data

More alternative scenario:

1) Friend from first\second circle data is not public - The system will use educated guesses to fill the missing data - if possible - to create the most satisfying CSV file to the main algorithm

3.1. Crawl user's facebook data:

Actors:

- User
- System
- Facebook's data

Precondition:

- The user pressed on "Get Facebook data" button

Postcondition:

- The system have the user's facebook's data that fit the CSV file format

Main scenario:

1) The system will login to Facebook with user's credentials 2) The system will get user's friends list
3) The system will get all user's friends:
    a) Total number of friends (TF)
    b) Age of user account (AUA)
    c) Amount of mutual friends (MF)
    d) Friendship duration (FD)
4) For each User's Friend the system will get his friend list 5) For each friend of friend, the system will get:
    a) Total number of friends (TF)
    b) Age of user account (AUA)
    c) Amount of mutual friends (MF)
    d) Friendship duration (FD) with connecting friend

4. <u>Choose the input parameters to the main algorithm:</u>

Actors:

- User
- System

Precondition:

- The system have the user's CSV input file via crawling or via upload

Postcondition:

- The system have the user's preferences about the input parameters to the main algorithm

Main scenario:

1) The system will present the user list of parameters that will be considered while running the algorithm:
    a) first circle's friends:
        i) Total number of friends (TF)
        ii) Age of user account (AUA)
        iii) Amount of mutual friends (MF)
        iv) Friendship duration (FD)
    b) second circle's friends:
        i) Total number of friends (TF)
        ii) Age of user account (AUA)
        iii) Amount of mutual friends (MF)
        iv) Friendship durations (FD) with connecting friends
2) The user will choose the parameters that he wants to run algorithm according to.

Alternative scenario:

1) The user didn't choose any parameter - The system default state is to run the algorithm based on all parameters

5. Main Algorithm execution and presenting the result as graphs:

Actors:

- User
- System

Precondition:

- The system have the user's CSV input file via crawling or via upload
- The user possibly selected parameters to consider while running the algorithm

Postcondition:

- The system will present 3 options of graphs to the user

Main scenario:

1) Running the main algorithm according the project's article (Use-case 5.1)
2) Algorithm's result presentation as graphs (Use-case 5.2)

5.1. Running the main algorithm according the project's article:

Actors:

- User
- System

Precondition:

- The system have the user's CSV input file via crawling or via upload
- The user possibly selected parameters to consider while running the algorithm

Postcondition:

- The system generate a weighted graph based on user's social network data (CSV file) and according to "An Information-Flow Control model for Online Social Networks based on user-attribute credibility and connection-strength factors" by Ehud Gudes and Nadav Voloch article,
  With score to each second circle's friend representing his credibility.

Main scenario:

1) The user pressed the "Run the algorithm" button
2) The system run the algorithm with the CSV file and user preferences as an input. (Use-case 5.1.1)

Alternative scenario:

1) The user pressed the "Run the algorithm" button
2) The system run the algorithm with the CSV file and user preferences as an input
3) **The algorithm crashed** while running
4) The system will show an error message to the user
5) The system will allow the user to re-run the algorithm

5.1.1. Main algorithm execution:

Actors:

- User
- System

Precondition:

- The user pressed the “Run the algorithm” button

Postcondition:

- The system generate a weighted graph based on user’s social network data (CSV file) and according to "An Information-Flow Control model for Online Social Networks based on user-attribute credibility and connection-strength factors" by Ehud Gudes and Nadav Voloch article, With score to each second circle’s friend representing his credibility.

Main scenario:

1) The system will extract only parameters from csv that the user choose.
2) The system will generate a value from 0 to 1 for each of the parameter
3) The system will calculate a value for each node and edge (see Glossary)
4) The system will calculate a value from all 2 edges distance paths starts at user node to second’s circle friends
5) The system will use Dinic Algorithm for finding the paths with maximum value of paths
6) The system will decide according step 5 if the second circle node is Adversary or Acquaintance

5.2. Algorithm's result presentation as graphs:

Actors:

- User
- System

Precondition:

- The main algorithm finished successfully to run

Postcondition:

- The system will present 3 options of graphs to the user

Main scenario:

3) The main algorithm finished successfully to run
4) The system will process the algorithm output as 3 options of graphs
5) The user asks to move the results page
6) The system will present 3 output graphs options to the user:
    a) Information flow model graph
    b) Clique based graph
    c) Summary Graph

    (As described in section 1.5 in this document - **Main outputs)**

Alternative scenario:

1) The system had an error while generating the graphs - The system will show an error message and rerun the graph generation process

6. Graph choose by the user:

Actors:

- User
- System

Precondition:

- The system finished successfully running the algorithm and presenting 3 options of graphs for the user to choose

Postcondition:

- The system will present selected graph

Main scenario:

1) The user choose one of the options
2) The system will show selected graph

7. Graph filter according user selection:

Actors:
- User
- System

Precondition:
- The system presents a graph to user

Postcondition:
- The system show the graph's part that the user selected

Main scenario:
1) The system presents options for filtering the graph:
    a) All connections
    b) Adversary connections
    c) Acquaintance connections
2) The user selects one of the options
3) The system will present the graph's part that the user selected

Alternative scenario:
1) The user didn't select any option - The system will continue present the default graph
2) The filtered graph is empty - The system will present suitable message

8. Zooming in\out the presented graph:

Actors:
- User
- System

Precondition:
- The system presents a graph to user

Postcondition:
- The presented graph is zoomed in \ zoomed out as the user requested

Main scenario:
1) The user points the mouse on the area that he want to zoom in\out
2) The user use his mouse scroll to zoom in\out
3) The system enlarge\reduce the area of user's point

Alternative scenario:
1) The user reach the limit - Graph is too small\big - The system will limit the zoom in\out action

9. Showing info on each node in the graph:

Actors:

- User
- System

Precondition:

- The system presents a graph to user

Postcondition:

- The system will present the information of the node (user's friend)

Main scenario:

1) The user press on one of the nodes that represent user's friend in first\second circles of friends.
2) The system will show the friend's details that was input for the algorithm-
    a) For friend in first circle:
        i) Total number of friends (TF)
        ii) Age of user account (AUA)
        iii) Amount of mutual friends (MF)
        iv) Friendship duration (FD)
    b) For friend in second circle:
        i) Total number of friends (TF)
        ii) Age of user account (AUA)
        iii) Amount of mutual friends (MF)
        iv) Friendship durations (FD) with connecting friends
    c) Determination if the user is Adversary or Acquaintance

Alternative scenario:

1) The information of the user is based on suggested guess - The information in this node will include a notify that the information is inferred

10. <u>Disconnecting from the system:</u>

Actors:

- User
- System

Precondition:

- The user is connected to the system

Postcondition:

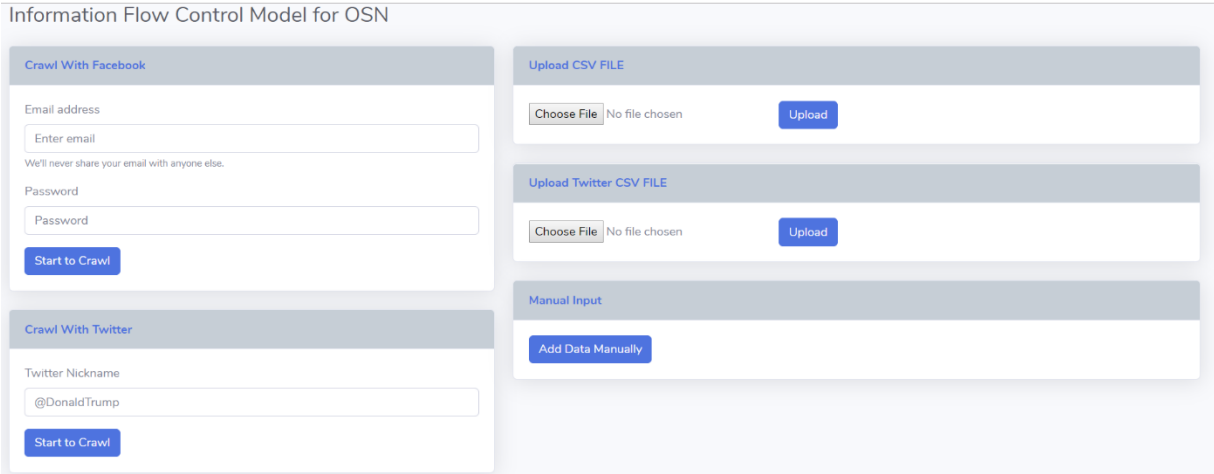
- The user is disconnected from the system
- There is no saved data of the user on the server side

Main scenario:

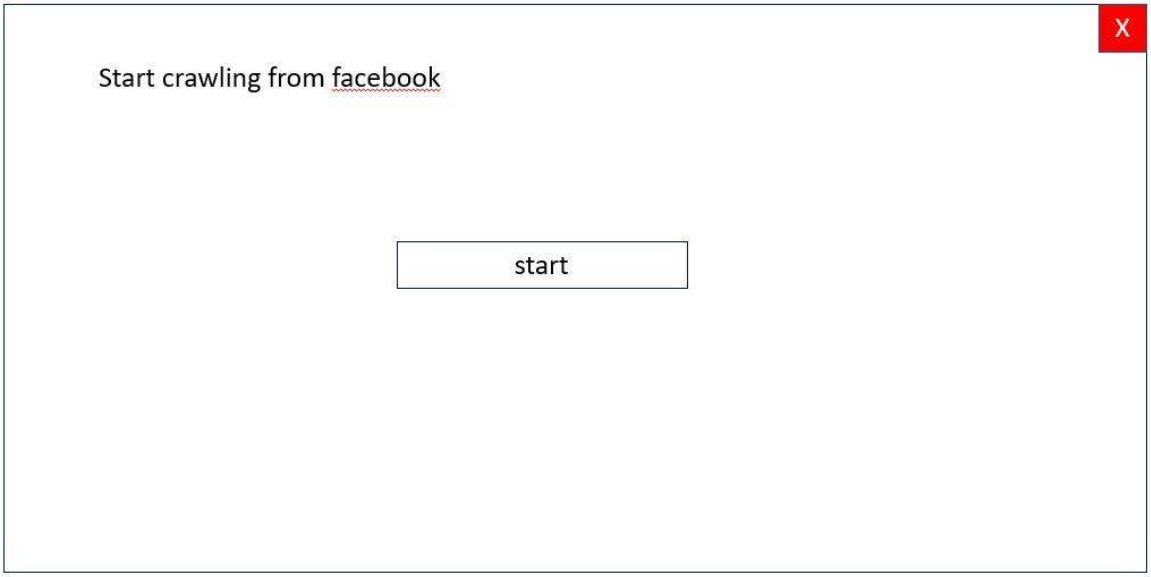
1) The user press the "Disconnect from the system" button

2.2.1 Main screens - first draft

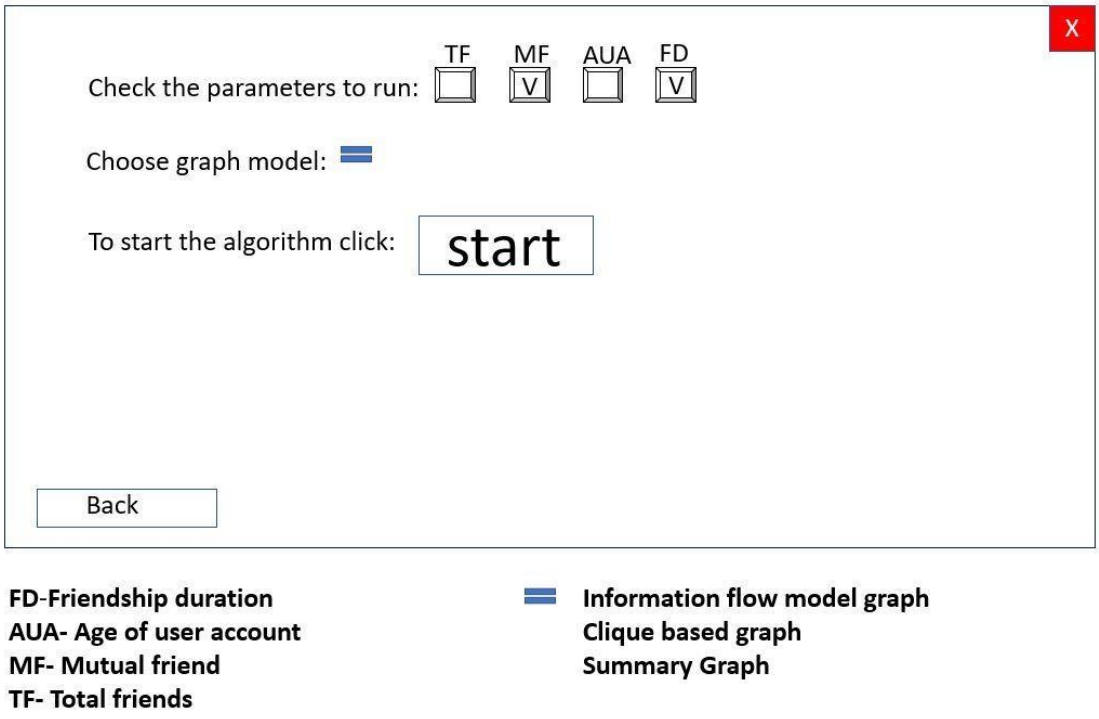
1. Enter page: login and upload csv file –



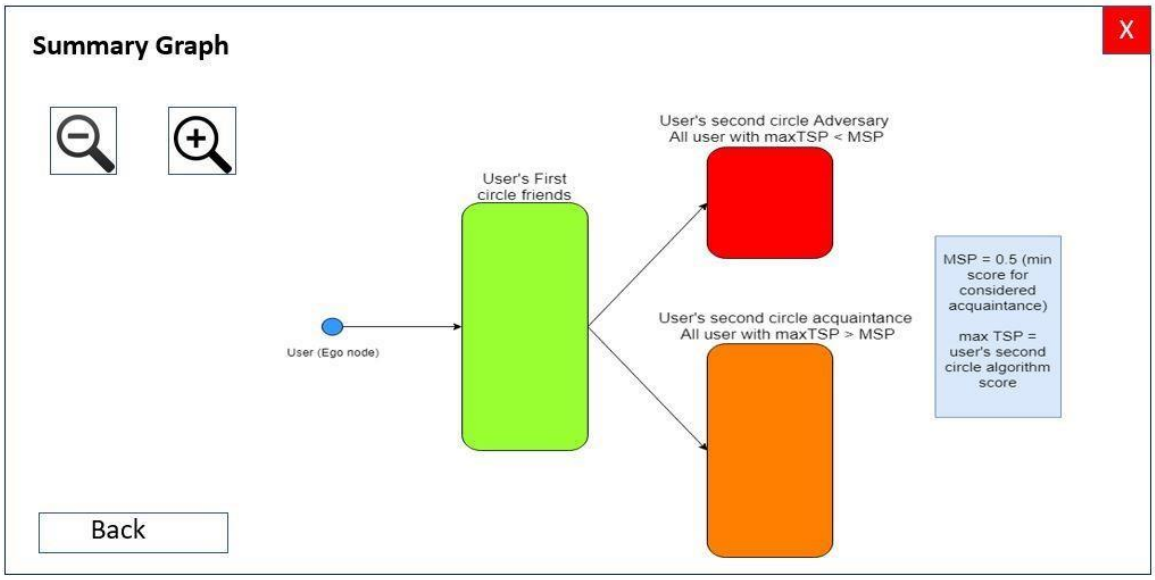
2. Start crawling page -

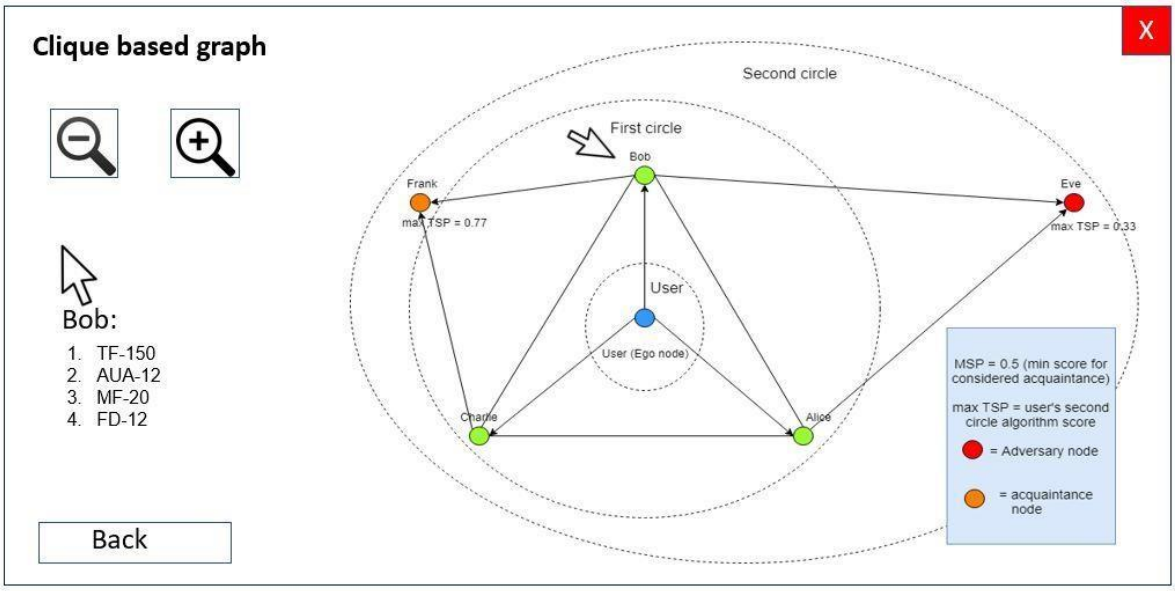


3. Parameters selection and algorithm execution -

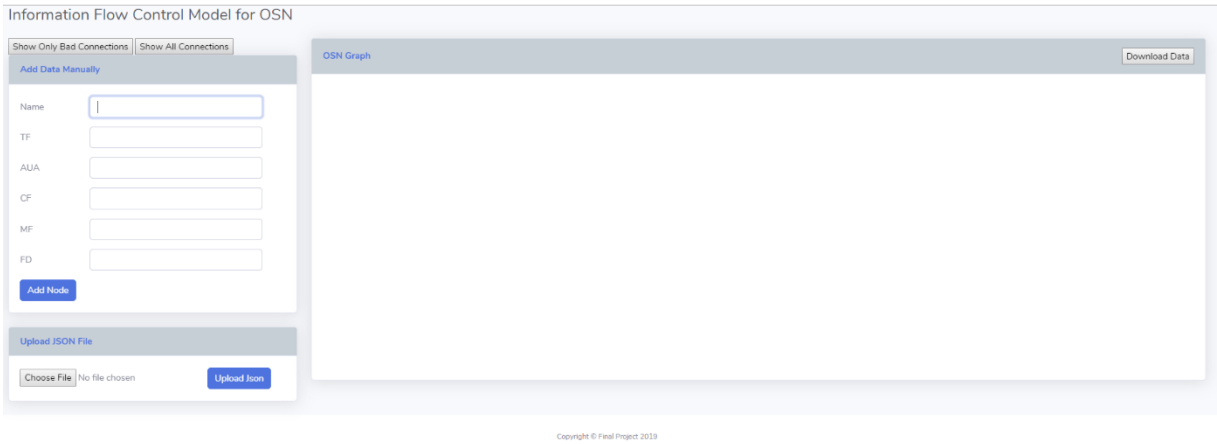


4. Outputs page examples -

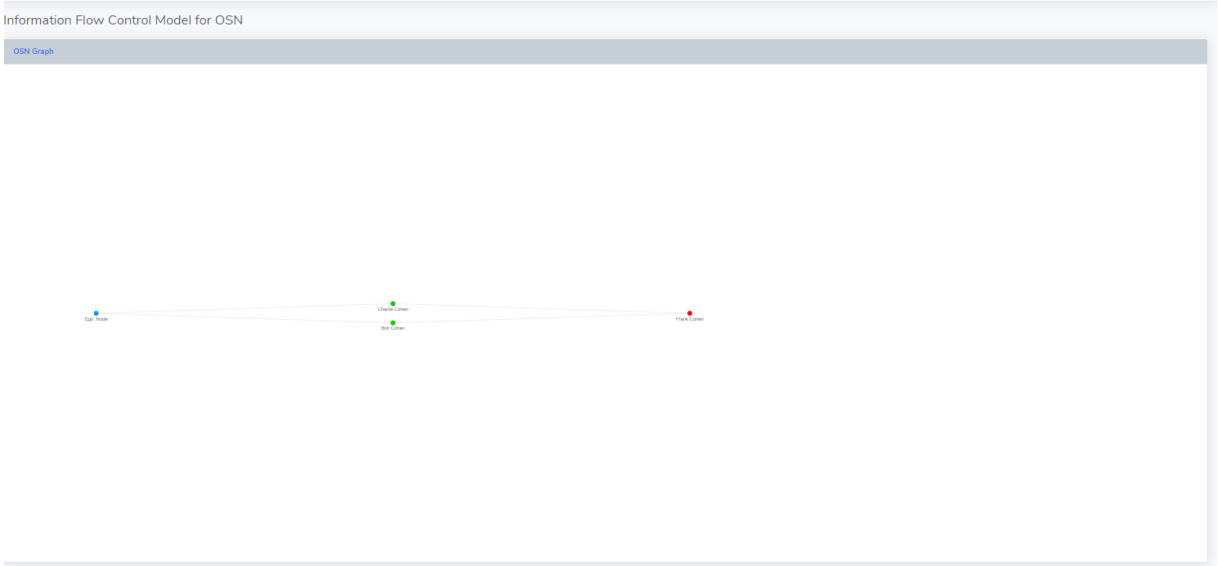




5. Add manually graph screen-



6. <u>upload csv file screen</u>-



2.3 Special usage considerations

1. Sign in to the system gives us access to the personal user data.
   We will use this information only for researching purposes.
   We committed to not store any information in our servers, the information won't be exposed to any third party.

2. During the scan, if the system will encountered with missing necessary information the algorithm will fill the missing information by changing the percentages calculation of the parameters.

3. The uploaded CSV file should be strict to our structure described in section 1.5.

Chapter Three

Functional Requirements

Our system's main functional requirements are:

1. The system will run steadily on a deployed web server, which response to user according to his requests.

2. The system will allow the user to connect to his social network and to gain an access to his social connection- e.g using the Facebook API - "login with facebook".

3. The system will support **dynamic** information gathering of logged in user (logged as in bullet 2), using known crawling modules.

   3.1. This information gathering will collect user's first circle friend's information (info about his direct connections), and collect information about his second circle connection (about all friends of user's first circle friend).

4. The system will process the gathered information to make it suitable as an input for the main algorithm that we will implement (will be discussed in bullet 6).
   The system will save according all user's first circle friends (directed friends):
   - Total number of friends (TF)
   - Age of user account (AUA)
   - Amount of mutual friends (MF)
   - Friendship duration (FD)

   And according user's second circle friends (friends of friends- if data if public):
   - Total number of friends (TF)
   - Age of user account (AUA)
   - Amount of mutual friends (MF)
   - Friendship durations (FD) with connecting friends

   Eventually, all the processed information will be stored in CSV file.

5. The system will produce a CSV file of the collected information for future use.
   The file will be anonymous for reasons of privacy.
   No one will be able to use this file outside the system.

6. The system will support **static** upload of already generated CSV file (to avoid rerunning of bullet 3 and 4).

7. The system will allow the user to choose the MST limit.

8. Main functional requirement of the system is the Implementation of the algorithm described in the main article that our project dealing with:

"An Information-Flow Control model for Online Social Networks based on userattribute credibility and connection-strength factors" by Ehud Gudes and Nadav Voloch.
The output of this functional requirement is weighted graph that represent the output the algorithm. For each node (friend in first\second circle) and edge (connection from user to friend and from user's friend to his friend) the algorithm will produce a score.

9. The system will allow representing the weighted graph in multiple graphical modules which the user can choose from a list.
   The system will mark the "bad paths", paths that expose the user data to other users that considered adversary, and present them in each graph modules.
   The graphical modules are:
   1. Information flow model graph
   2. Clique based graph
   3. Summary Graph

   (As described in section 1.5 in this document - **Main outputs)**

10. The system will allow the user to filler each graph module results according these parameters:
    - All connections
    - Acquaintance connections
    - Adversary connections

11. For better understanding the user's connection graph, the system will allow the user to zoom in\out the graph for navigating all his connections.

12. The system will support presenting the information about each node in the output graph-
    For nodes in first circle of friends:
    - Total number of friends (TF)
    - Age of user account (AUA)
    - Amount of mutual friends (MF)
    - Friendship duration (FD)

    For nodes in second circle of friends:
    - Total number of friends (TF)
    - Age of user account (AUA)
    - Amount of mutual friends (MF)
    - Friendship durations (FD) with connecting friends

13. The system will support user disconnecting from the system.

    13.1. Disconnecting safely from user's social network account.

    13.2. The system will export to the user an encrypted CSV file with the gathered data.

13.3. The system will clean any related data from the servers.

14. The system will present to the user more visual information in addition to the graphs.

    14.1. The system will present the adversary connections in a list, this feature will recommend to the system user which users are classified as ‘bad’ connections and the user should avoid sharing information with them.

    14.2. The system will present a table with user/connection and their attribute credibility calculation.

    14.3. The system will present a table with friends, MST of friend’s network and removal candidates and their trust value.

    14.4. The system will present a table with users and their sharing probability.

15. The system will allow to add a graph manually.
    15.1. There is a page that allows you to insert the data on a node and then insert it into a graph.
    15.2. Each node insert re-displays the current graph.
    15.3. It is possible to save the generated graph to a .csv file for future use.

The tables below are example for tables described at the previous page.
This tables were first shown in the article "*Generating a secure information flow in Online Social Networks using a Minimum Spanning Tree algorithm*"

User/Connection	Attribute calculation
Eve	$u = \langle uTF, uAUA \rangle = \langle 1,1 \rangle = 1$
George	$u = \langle uTF, uAUA \rangle = \langle \frac{22}{245}, \frac{9}{24} \rangle = 0.23$
Frank	$u = \langle uTF, uAUA \rangle = \langle \frac{100}{245}, \frac{14}{24} \rangle = 0.5$
Harry	$u = \langle uTF, uAUA \rangle = \langle \frac{130}{245}, 1 \rangle = 0.77$
Alice	$u = \langle uTF, uAUA \rangle = \langle 1,1 \rangle = 1$
Eve - George	$c = \langle cMF, cFD \rangle = \langle \frac{2}{37}, \frac{2}{18} \rangle = 0.08$
Eve - Frank	$c = \langle cMF, cFD \rangle = \langle \frac{31}{37}, \frac{10}{18} \rangle = 0.7$
Eve - Alice	$c = \langle cMF, cFD \rangle = \langle \frac{11}{37}, 1 \rangle = 0.65$
George - Alice	$c = \langle cMF, cFD \rangle = \langle \frac{2}{37}, \frac{3}{18} \rangle = 0.11$
Frank - Alice	$c = \langle cMF, cFD \rangle = \langle \frac{3}{37}, \frac{12}{18} \rangle = 0.37$
Frank - Harry	$c = \langle cMF, cFD \rangle = \langle \frac{2}{37}, \frac{12}{18} \rangle = 0.36$
Harry - Alice	$c = \langle cMF, cFD \rangle = \langle 1, 1 \rangle = 1$
Ego - Alice	$c = \langle cMF, cFD \rangle = \langle 1, 1 \rangle = 1$

user/connection and their attribute credibility calculation table

Friend	MST of the friend's network	Removal candidates and their trust value
Alice	{Alice-George, George-Eve, Alice-Simon, Frank-Harry}	**George:** u=0.23
Charlie	{Charlie-Linda, Linda-John, John-Isaac, Isaac-Karl}	**John:** u=0.91
Bob	{Bob-Nick, Nick-Philip, Philip-Orson, Orson-Marry}	**Nick:** u=0.56
David	{David-Quinn, Quinn-Simon, Simon-Thomas, Thomas-Ryan}	**Thomas:** u=0.34

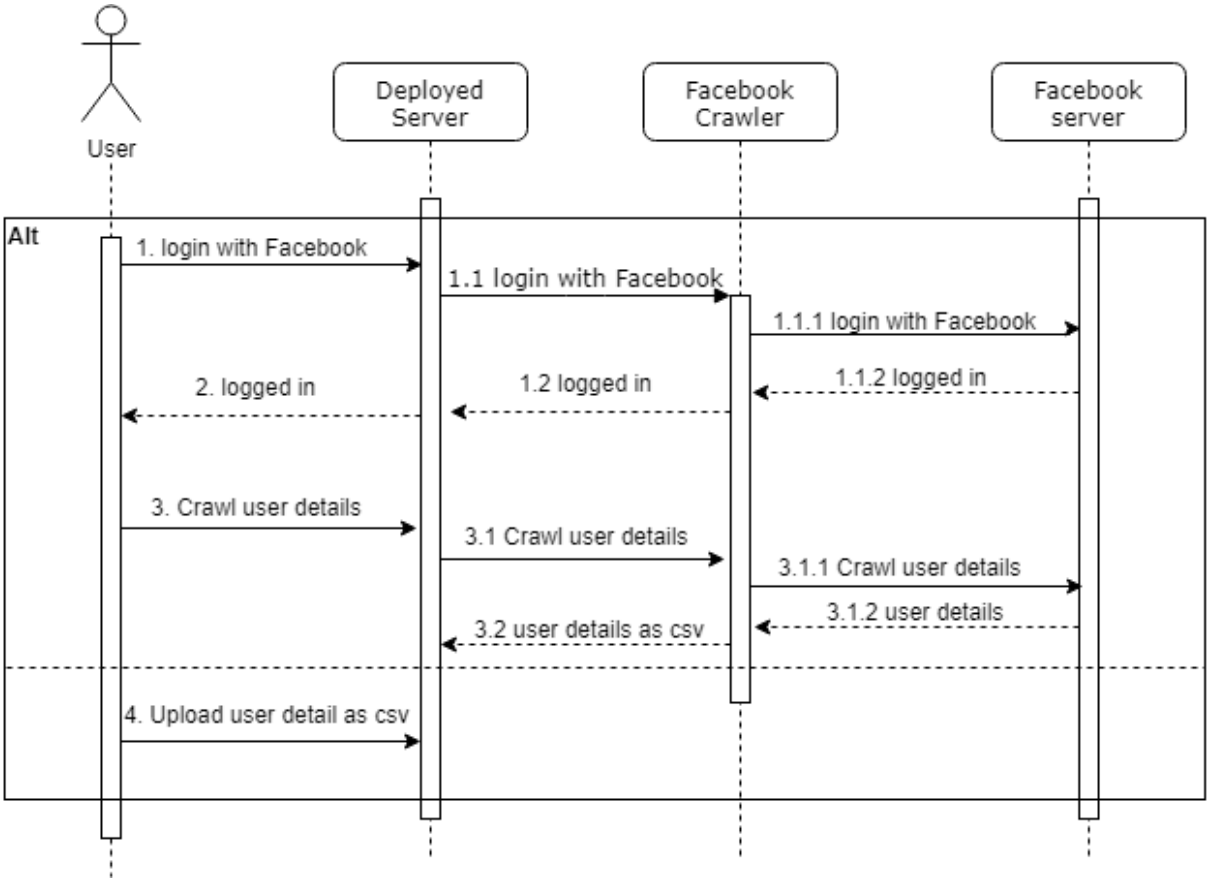
friends, MST of friend's network and removal candidates & their trust value table

User	Average SP (Sharing Probability)
John	0.413794
Isaac	0.482758
Linda	0.54138
Karl	0.527586
Frank	0.427586
Eve	0.496552
Harry	0.55862
George	0.386206
Ryan	0.534482
Quinn	0.510344
Thomas	0.365518
Simon	0.465518
Nick	0.427586
Marry	0.503448
Philip	0.462068
Orson	0.434482

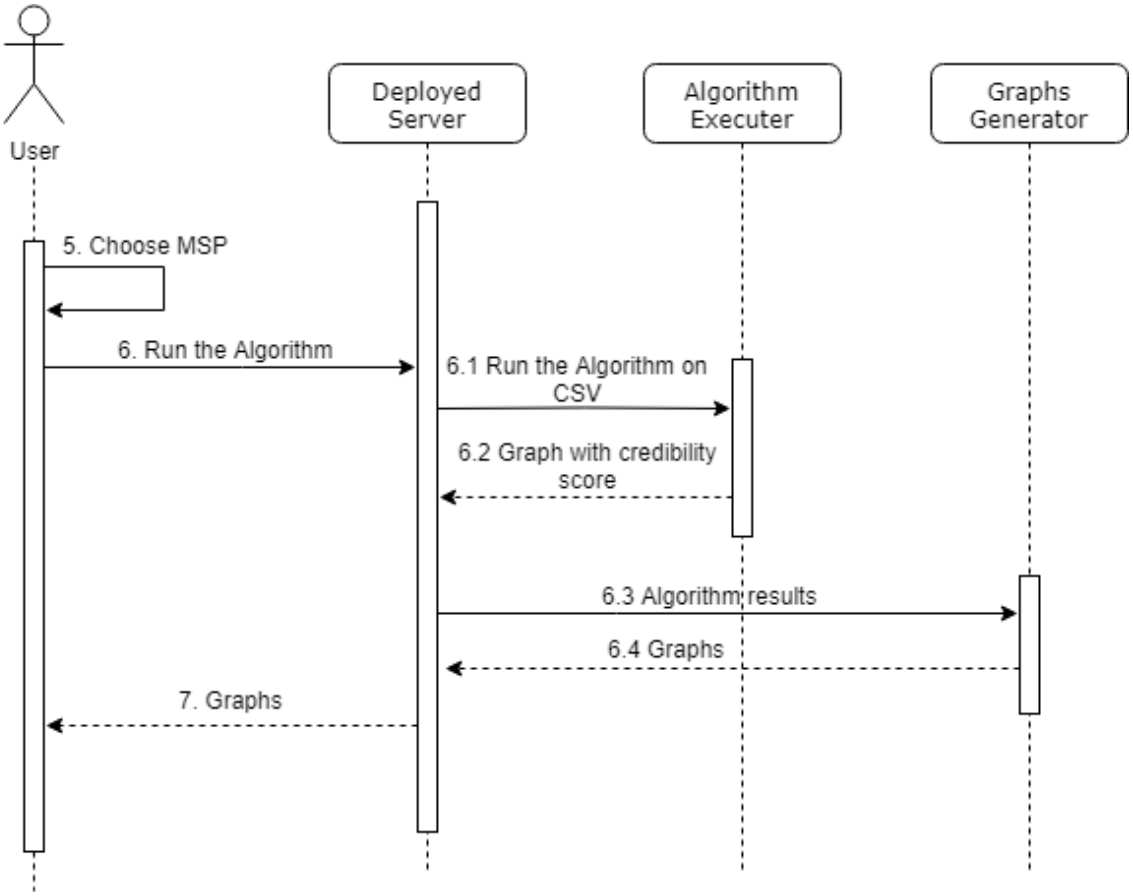
users and their sharing probability table

Sequence diagrams-

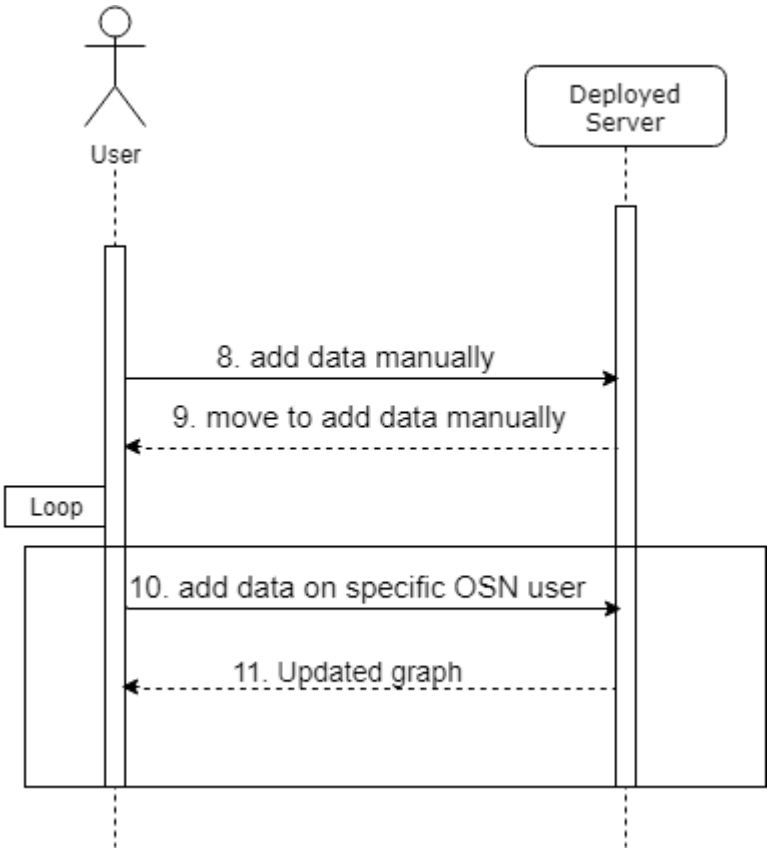
1) CSV Gathering sequence:



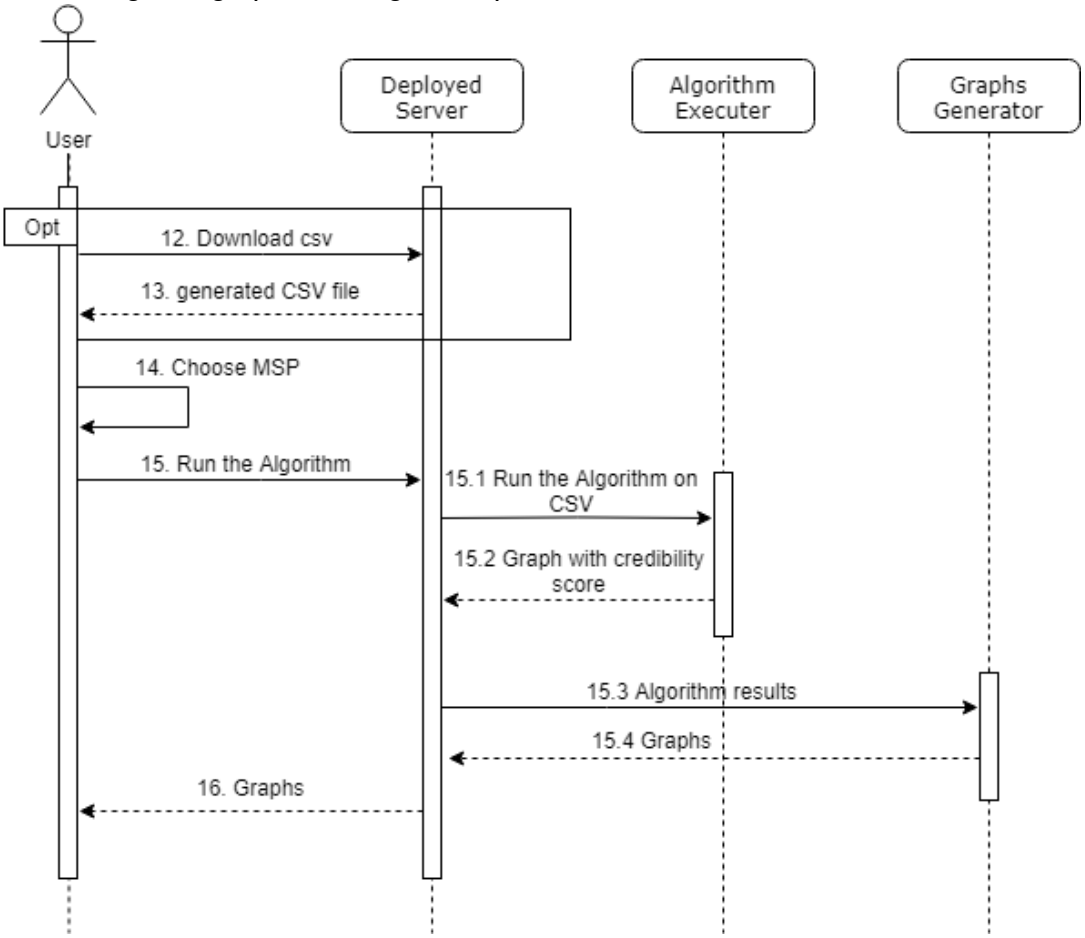
2) Algorithm execute sequence:

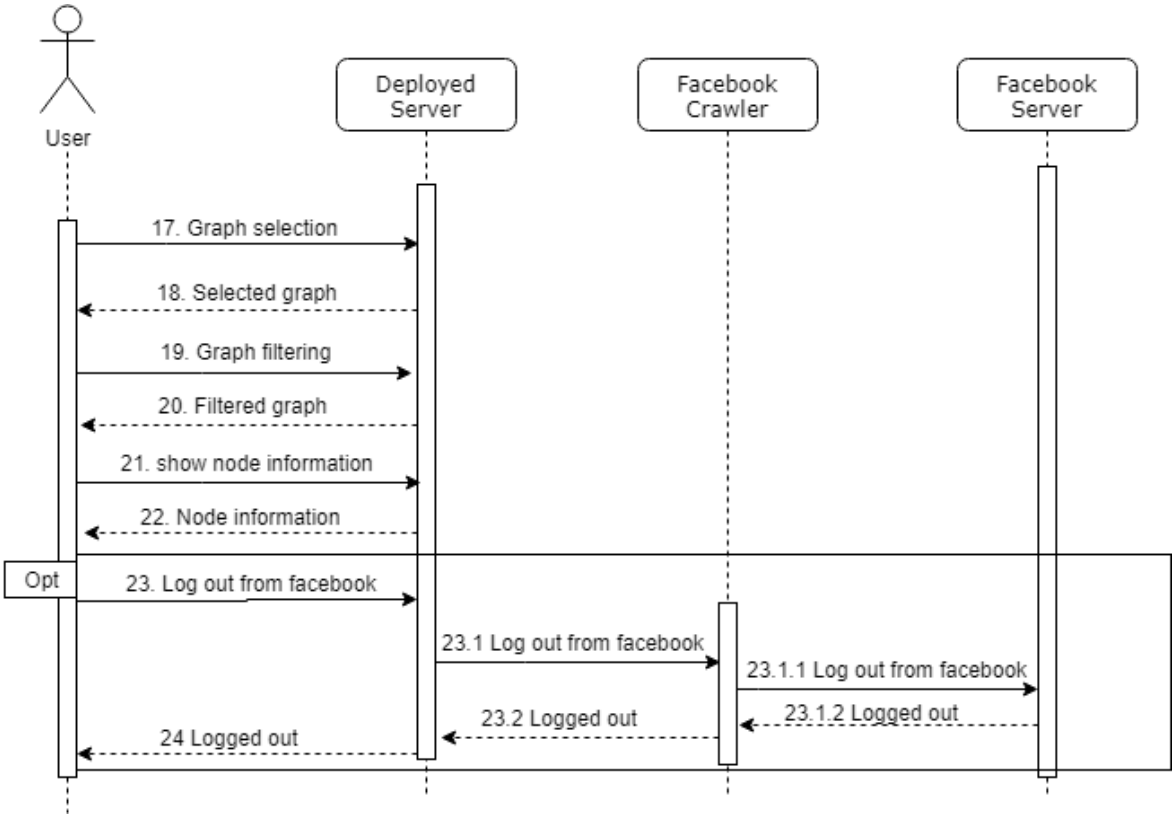


3) Add graph manually:



4) Working with graphs and logout sequence:





Chapter 4

Non-functional requirements

4.1 Implementation constraints

Performance

1. Let A be constant representing the time required to crawl 1 facebook page. Let n be number of friends of connected user.
   Let m be the average number of one of user's friend's friends.
   Therefore, the Crawling runtime is O(A*n+A*n*m) = O(A*n*m) .

2. The algorithm will find all possible path based on the Dinic algorithm therefore it will run at O((V^2)*E).

3. **False negative ratio -** The ratio of
   | Adversary connections not discovered **| / |** Adversary connections in total **|** will not be above 25%.
   False positive ratio - The ratio of
   | Acquaintance connections calculated as Adversary **| / |** Acquaintance connections in total **|** will not be above 5%.

Reliability & Stability

1. The system will support at least 150 concurrent users.
2. The output graph is going to be constructed on the fly, in case of a crash the system will present the obtain graph so far.
3. The system will handle crashes by restart the process.
4. During the scan, if the system will encountered with missing necessary information the algorithm will fill the missing information by changing the percentages calculation of the parameters.

Safety & Security

1. Once the user is logged in, he will be asked about the permissions he agrees to grant to our system.
2. We promise not to store any data on our servers, at the end of the process all of the data will be cleared.
3. The data is used only for research objectives, we won't use any data for commercial purposes.
4. We will apply data anonymization techniques on the data sets to ensure that the data stays 100% secured.
5. Our system will produce CSV file with an anonymous data in order to protect the data.

Portability

1. The system interface will run on a Web server, users will use our system through Google Chrome browser.

Usability

1. Usability is very important for us, we will make sure that the use of our system will be as intuitive as possible. We will add as many explanations and instructions as needed.
2. Users of the system don't required to have any technology background or special training in order to use our application.

Availability

1. As long as the server is running it will wait and respond to HTTP requests.
2. Our system scans social networks, thus it depends on social network availability.
3. In case of social networks dysfunctions our system will still provide offline support such as CSV import and analyzation.

4.2 Platform constraints

We will use several APIs:

1. GraphAPI, the official api of facebook.
2. twitter api.

the APIs don't provide enough options that we need for our system, therefore we decided to build a Crawler.
This decision made us choose to write the system in Python.
We will present the output various graphs, thus we will use JavaScript to build the client.

We will use Cytoscape.js , NetworkX.js, igraph, Gephi, D3 libraries that will present different types of graph for the user use.
We will use Azure Cloud services for our deployed server, Microsoft has a great terms for students and the system is very comfortable to use.
In addition, Azure has build in anonymization tools that will help us with privacy issues.

We are building an interactive system, the user will insert his user details to connect to social network account.
After successful connection he will be asked to grant permission to use his personal data.

The demo will simulate the process, we will use real data that were collected in advance.
After processing and calculating the data that the system will produce and show a graph to the user, the user will be able to change it to different layouts and "play" with the graphs.

We will collect the real data with our personal Facebook accounts and lists of friends.

4.3 Special restrictions & limitations

There are some cases when Facebook accounts decide to not expose their list of friends.
In this cases we will not continue to scan their list of friends with our Crawler, we will exclude them from the output graphs.
We may show an proper massage with an explanation.

Chapter 5

Risk assessment

Risk	Risk Probability	Handling with the risk
Visualization big data	High	Make research about tools to present graph with many details(Nodes, Edges and more data),Learn how to used it, and what is the ideal vision.
Crawler for Facebook	High	Taking a course about crawlers, reading documentation and building proxy server to overcome Facebook restrictions.
Apply real data on the system	High	Process the collected data and making it similar to our CSV format.
Implement the algorithm on Big Data	Medium - High	Build suitable modules and use proper libraries.
Visualization	Medium	Read proper guides and documentation. Consult with Rami Puzis.
Data anonymization	Medium	Learn about this topic and use proper modules.

Crawling Facebook is our main concern. Facebook is putting great effort to identify such behavior and block them.
We will try to use proxy server or vpn services to overcome this problem.

Because of that, we will start with this subject. We will concentrate our forces on Facebook and more precisely on the Crawler for Facebook.
In case that there would not be any major progress with the crawler, we will scan our own Facebook accounts and collect data from private.
In addition we will work with synthetic data and with data that was given to us by our mentors.

Plan for the first iteration - Building the proof of concept

The order of next bullets are relative to their risk, and will be done in this order.

1. Facebook crawler implementation- As we said, the main risk in our project is succeeding to crawl facebook data of logged in user. Therefore, the first task that we will do is to crawl Facebook. (Functional requirements 2, 3, 4)

2. Main algorithm implementation- After gathering the Facebook information we will start to implement the main functionality of our system - The main algorithms based on the articles that we dealing with.
   We will deal with the “An Information-Flow Control model for Online Social Networks based on user-attribute credibility and connection-strength factors” article. (Functional requirements 8)

3. First step visualization- In order to present the outputs of the algorithm, we will put an effort to present in the first iteration basic visualization of the user’s Facebook’s network for detecting adversary users. The value of this step is high the user main concern is to view the outputs and to analyze the graphs (Functional requirements 9)

Appendices

File Format

our system support loading a CSV file described at section 1.5 under main inputs.

Glossary

- **social network -** e.g., Facebook, Twitter etc.
- **OSN -** Online social network
- **user details -** username and password.
- **personal data -** number of friends, how long the user is a member of the social network, when he become friends with a specific user, etc.
- **dynamic information gathering -**
- **first circle -** the direct friends of the user.
- **second circle -** friends of the direct friends of the user.
  e.g., assume that a user ‘x’ has two friends ‘y’ and ‘z’. ‘n’ and ‘m’ are friends of ‘y’, ‘w’ is friend of ‘z’ but ‘n’, ‘m’ ,‘w’ aren’t direct friends of ‘x’. The second circle for X contains: n,m,w.
- **output graph -** a graph that contains nodes and edges and represents a social network. The graph shows the data we scanned in a visual and comfortable way to the user.
- **nodes -** represents the users in the social network.
- **edges -** represents the connections between users.
- **MSP** – minimum sharing probability.
- **TSP** - total sharing probability.
- **TF**-total friends.
- **MF**- mutual friends
- **AUA**- age of user account
- **FD**-friendship duration
- **CF**-connecting friend id

References

1. E Gudes, N Voloch. “An Information-Flow Control model for Online Social Networks based on user-attribute credibility and connection-strength factors”. International Symposium on Cyber Security Cryptography and Machine Learning. (pp. 55-67). Springer, Cham. (2018)

2. Nadav Voloch and Ehud Gudes. “Generating a secure information flow in Online Social Networks using a Minimum Spanning Tree algorithm”.

3. Nadav Voloch , Priel Levy , Mor Elmakies and Ehud Gudes. “A Role and Trust based Access Control model for Online Social Networks”.

4. Levy, S., Gudes, E., & Gal-Oz, N. (2016, July). Sharing-habits based privacy control in social networks. In IFIP Annual Conference on Data and Applications Security and Privacy (pp. 217-232). Springer, Cham.