

Introduction to Computer Security – COMP.2300 – 202

Professor Ian Chen

Project: Secure File Transfer

Team Members: Eddie Tran, Alvin Tran, Joshua Hou

December 14th, 2021

Final Project Report

1. Introduction

With the use of cryptographic and cybersecurity concepts, we were tasked to implement a program that offers similar concepts to Apple's AirDrop. AirDrop allows for users to transfer files without the need for wi-fi by broadcasting, discovering, and negotiating connections. Below are our team's step by step explanation of how we attempted each milestone.

2. Milestone 1 – User Registration

a. Implementation

- i. First, check whether if "user.json" is empty or not.
- ii. If not empty, then we have an existing user, skip registration, and do login check
- iii. If empty, then get the user's credentials and add them into "user.json"
- iv. Encrypt "contacts.json" (will be used later)
- v. "user.json" will display the user's name, email, and encrypted password

b. Security

- i. We have used `getpass()` to have a ssh type password input
- ii. We have used `bcrypt` to encrypt the user's password

- iii. We have used a minimum password requirement

3. Milestone 2 – User Login

a. Implementation

- i. Get and authenticate the user's credentials
- ii. If not user, print invalid email password combination and loop
- iii. If is user, decrypt "contacts.json", and enter interface

b. Security

- i. We have used an authenticator that checks if the user's input is equivalent to the saved credentials of the registered user.

4. Milestone 3 – Adding Contacts

a. Implementation

- i. When input add, first check to see if contacts.json is empty
- ii. If empty, then create the dictionary, append the new contact's name and email
- iii. If not empty, then append the new contact
- iv. If new contact is an existing contact via email, then update the existing contact
- v. All will write to the contacts.json

b. Security

- i. We have encrypted the contents of contacts.json on exit

5. Milestone 4 – Listing Contacts

a. Implementation

i. A server program is used to contain information of users who are online in a server list json file. The user's information that is stored is the user's email, the user's port number they used to communicate to the server, and the user's contacts. This information will be used to find out who is online, and who is friends with who so that the contacts that meets the requirements can be listed. When the user exits the program, the user is removed from the server list json file.

ii. Check if the user has any contacts. If the user has no contacts, nothing is done.

iii. If the user does have contacts, then a for loop will happen going through each of the contacts in the contacts json file.

iv. During the for loop, a flag message is sent to the server program via UDP messaging saying that the client is going to list contacts. In addition, the user's email and a contact email is sent to the server program via UDP messaging.

v. In the server program, it receives the flag message and the user's email and a contact email.

vi. Checks if the contact email is in the server list json file. If it is there, that means they are online, otherwise, the user is offline, and a message is sent back to the client resulting in the contact not getting listed. If the contact is online, check the contact's contact information to see if the user is there. If he is there, they are friends, otherwise, they are not friends, and a message is sent back to the client resulting in the contact not getting listed.

vii. If the contact is online, and they are friends, return "true" to the client so that the contact will be listed.

viii. Repeat until the loop of going through each of the contacts in the user's contacts finishes.

b. Security

i. N/A

6. Milestone 5 – Secure File Transfer

a. Implementation (Did not complete.)

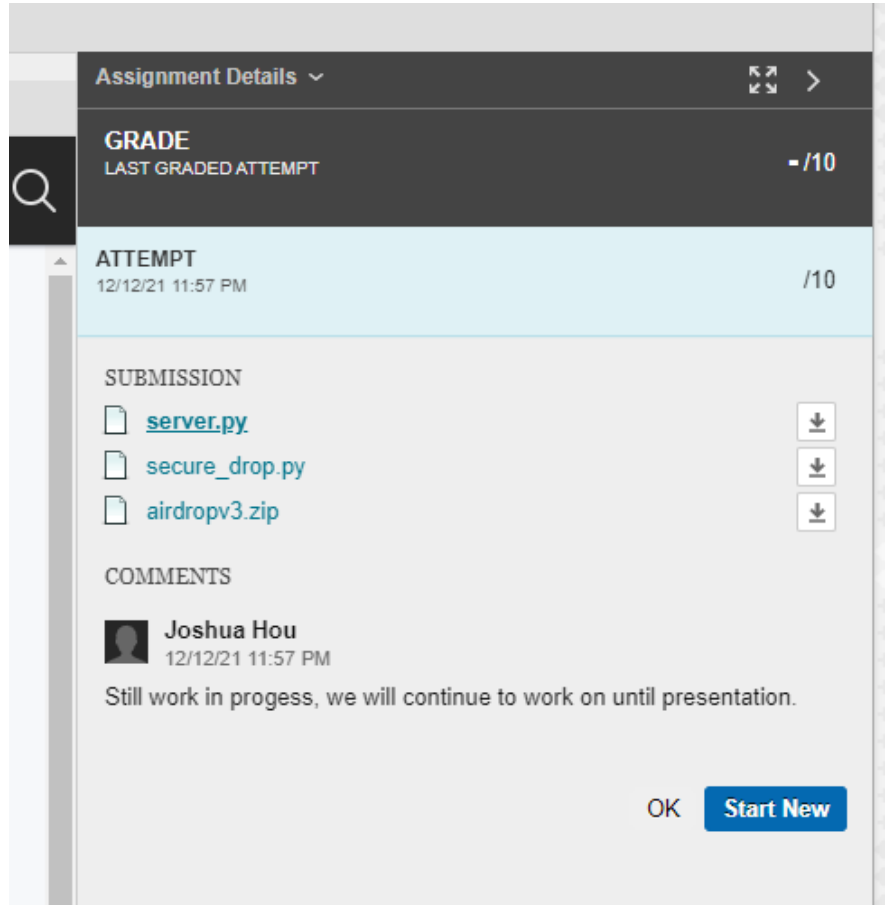
- i. Checks if the user's contact list is empty. If empty do nothing.
- ii. A flag message is sent to the server program via UDP messaging saying that the client is going to send a file.
- iii. User sends to the server the user email and the specificized email of who they want to.
- iv. Server checks if the specificized email is online and is friends with the user. If they are both of those, sends back the specificized email's port number, otherwise, send false and the client does nothing.
- v. This was where we got stuck. With the port number, the user was supposed to make a connection with the specificized email via TCP, but we didn't know how to make the specificized email accept while being idle on the secure drop client because the accept function makes it so that the client wait being unable to use the functionalities of the client.

b. Security

i. N/A

7. NOTE

a. We submitted an earlier version of the project on 12/12/2021. The submission we are sending now is an updated version with more functionality.









The screenshot shows a web interface for assignment details. At the top, there's a dark header with 'Assignment Details' and a dropdown arrow. Below this, a section labeled 'GRADE' shows 'LAST GRADED ATTEMPT' with a score of '- /10'. A light blue section labeled 'ATTEMPT' shows the date '12/12/21 11:57 PM' and a score of '/10'. The 'SUBMISSION' section lists three files: 'server.py', 'secure_drop.py', and 'airdropv3.zip', each with a download icon. The 'COMMENTS' section shows a comment from 'Joshua Hou' dated '12/12/21 11:57 PM' with the text 'Still work in progress, we will continue to work on until presentation.' At the bottom right, there are 'OK' and 'Start New' buttons.

Assignment Details ▾


GRADE
LAST GRADED ATTEMPT - /10

ATTEMPT
12/12/21 11:57 PM /10

SUBMISSION

-  [server.py](#) 
-  [secure_drop.py](#) 
-  [airdropv3.zip](#) 

COMMENTS

 **Joshua Hou**
12/12/21 11:57 PM

Still work in progress, we will continue to work on until presentation.

OK [Start New](#)

b.