# Machine Problem 2

**Java 8 is needed to run the code. Make sure you have Java 8.**

Cybersecurity incidents are an ever-growing problem for companies. Recently, the President of the United States called for steps to improve cybersecurity, including new legislation that would limit liabilities of companies that report incidents that had happened to them. In this machine problem, we will study how companies can share cybersecurity incidents with each other without revealing unnecessary information about the incidents.

Each group represents a company that has experienced 10 standard cybersecurity incidents. You seek other companies that have experienced incidents similar to the ones you have experienced. You want to learn whether other companies have experienced similar incidents, but without sharing the set of all incidents you have experienced.

You are given a list of cybersecurity incidents in the file incident\_list.txt labeled from 1 to 25. In this machine problem, each group will select exactly 10 cybersecurity incidents randomly from the list. You will use a private set intersection (PSI) protocol to see if other groups have experienced the same incidents. Each run of the protocol requires exactly two groups: a Client group and a Server group. *You are required to run the protocol with at least two other groups.*

## Running the PSI protocol

Suppose a group Client wants to find if another group Server has experienced the same incidents as it has experienced. You will use the provided code as follows:

Both the Client and the Server need to perform the following steps initially without sharing any information with each other at this stage:

1. Download the code from the git repository using using

git clone https://github.com/eSMC/PSI.git

1. Compile the code using
   * Unix/Linux/Mac: javac -cp .:flanagan.jar \*.java
   * Windows: javac -cp “.;flanagan.jar” \*.java
2. Select exactly 10 unique cybersecurity incidents from the list in file incident\_list.txt.
3. Create another text file say inputs.txt and put the labels of the 10 incidents you selected in step 2 and separate the labels by new lines. See example example\_inputs.txt to understand the format. Do not use the example ‘’example\_inputs.txt’’ file. *A group using example\_inputs.txt as inputs.txt will get zero.*

The Client needs to perform the following steps:

1. Run the following program:

Unix/Linux/Mac:java -cp .:flanagan.jar ClientPhase1 <cl-inputs.txt> <netid>

Windows: java -cp “.:flanagan.jar” ClientPhase1 <cl-inputs.txt> <netid>

Use netid of one of the group member.

1. This will generate two files with filename being netid and ClientPK.out.
2. Email both of these files to the Server.
3. Wait for the server to run its program and send you the file netid.out.
4. After you receive the file from Server named netid.out, put it in the same directory as the code and run the following program:

Unix/Linux/Mac: java -cp .:flanagan.jar ClientPhase2 <cl-inputs.txt> <netid.out>

Windows: java -cp “.;flanagan.jar” ClientPhase2 <cl-inputs.txt> <netid.out>

1. The program will output the common incidents.

Server perform the following steps:

1. Put the files (ClientPK.out and netid) received from the Client in the same directory as the code.
2. Run the following program:

Unix/Linux/Mac: java -cp .:flanagan.jar Server <sr-inputs.txt> <netid>

Windows: java -cp “.:flanagan.jar” Server <sr-inputs.txt> <netid>

1. This will generate a file with the filename netid.out. Email this file back to the Client.

## Tasks

Each group performs the role of Client and Server with at least two other teams, and does it for the following three settings.

1. In the first setting, both Client and Server group behave honestly and help each other to find the common incidents.
2. In the second setting, Client is ill intentioned and want to learn as much information about the Server incidents as much they can. In this case, Client is allowed to choose any incidents he wants and can run the protocol with the Server multiple times with different list of incidents.
3. In the third setting, Server is ill intentioned and want to learn as much information about the Client incidents as much it can. Client is honest. In this case, Client is allowed to choose any incidents he wants and can run the protocol with the Server multiple times with different list of incidents.

## Report

Write a 1-page report as follows:

* A short paragraph on your experience with the first setting.
* A paragraph on your experience with the second setting, and what strategies you use to learn maximum amount of information.
* A paragraph on your experience with the third setting, and what strategies you use to learn maximum amount of information.