CS-523 Midterm 2021

1 PIR

Private information retrieval (PIR) is a protocol between two parties: server and client. The server holds an indexed database. The client wants to retrieve a record with a given index. The goal of a PIR protocol is to hide which record is retrieved from the server. Any PIR protocol must satisfy two properties: correctness—the client receives the value that corresponds to the requested record, and client privacy—the server does not learn any information about the requested index.

For the rest of this question, assume you have a database D of n records. Let d_i be the i-th record. Each record in this database is exactly m bit long. A PIR protocol is composed of three algorithms: $q = \text{Query}(i)$ executed by the client to query the i-th record. The client sends q to the server. The server computes the response $r = \text{Response}(q)$ using the database and sends r to the client. Finally, the client retrieves the record $x = \text{Retrieve}(r)$ from the server's response r.

A naïve PIR approach is sending the whole database D to the client: $\text{Response}(*) = D$. This protocol satisfies both correctness and client privacy, but, obviously, this is not a good solution.

Part 1. Consider a vector e_i of n binary elements where all but the i-th element are zeros, and the i-th element is 1. Use fully homomorphic encryption (FHE) and the vector e_i to design a PIR protocol such that it (a) achieves correctness, (b) achieves client privacy, (c) has communication size linear in the number of records, and (d) does not send the full database. Assume that the FHE scheme supports multiplication between a plaintext (scalar) and a ciphertext. To describe the protocol, you can instantiate the functions Query, Response, and Retrieve.
Answer the following questions: (1) What is the threat model (e.g., honest, honest-but-curious, malicious server) (2) What is the computation and communication cost of the system in terms of n and m? (3) What is the multiplicative depth? All answers must be justified.

Part 2. A good PIR protocol needs to satisfy a third property: sublinearity—communication size must be less than linear in the number of records; otherwise, a protocol is not that different asymptotically from just transferring the full

database to the client. One approach to make the PIR protocol sublinear is changing the structure of the database from a 1-dimensional array of records to a 2-dimensional matrix of records. In other words, re-arrange D into a $n_1 \times n_2$ matrix M where $M_{i,j} = d_{(i*n_2+j)}$. Update your previous PIR protocol to additionally achieve sublinearity. To describe the protocol, you can instantiate the functions Query, Response, and Retrieve.
Answer the following questions: (1) What is the threat model? (2) What is the computation and communication cost of the system in terms of n and m? (3) What is the multiplicative depth? All answers must be justified.

Part 3. Another desired property can be server privacy where the client should only learn about the requested record and does not learn any information about the remaining $n - 1$ records. Do your protocols achieve server privacy? Justify your answer for both honest but curious, and malicious clients.

2 Pay-per-song Festival

The year is 2022. Events with thousands of people are possible again. While preparing their best come-back, the Kaléo festival organisers are thinking about a new pricing scheme. Instead of charging a flat rate to access the festival area, the organisers want to use a pay-per-song scheme under which visitors are charged depending on how many songs they listened to and on which stage. In this scheme, listening to the concerts at the Grande Scene is more expensive than on the smaller stages. To implement this new pricing scheme, each visitor will be given an Ultra Wide Band (UWB) tag. The UWB tag sends the visitor's position to a central service every minute. This allows the festival organisers to track visitors on the festival grounds and infer which stages they visited, when, and for how long.

Part 1. Describe two privacy concerns for festival visitors that are caused by the introduction of the location-tracking UWB tags and that were non-existent under the flat-rate pricing scheme.

Part 2. The Kaléo festival organizers heard from privacy experts that the UWB tags scheme introduces too many privacy problems and they are afraid that this may spook customers. They consider k-anonymous cloaks as a solution to these privacy problems. The organisers design an algorithm that runs on the UWB routers installed across the festival area. Every minute, this algorithm provides the organizers with regions and, within each region, the k users in that region.
Is this solution a good option for the Kaléo organizers to address the two privacy concerns you identified in Part 1? Justify. If a concern is addressed, recommend a value for k under which the pricing scheme still works as expected (visitors are charged accurately).

Part 3. As an alternative to k-anonymous cloaks, the organizers also consider an approach based on spatial obfuscation. In this approach, every time they send a visitor's position to the server, the UWB tags locally obfuscate this position by calling the obfuscation algorithm. The magnitude of noise (distance between the obfuscated position and the original position) is always within a fixed limit t. As a result, the mechanism ensures that for any two locations that are within radius t, the server cannot distinguish between them. Conversely, the locations that are further apart than radius t are distinguishable. Is this solution a good option for the Kaléo organizers to address the two privacy concerns you identified in Part 1? Justify. If a concern is addressed, recommend a value for threshold t under which the pricing scheme still works as expected (visitors are charged accurately).

3 Waterwolf

The Waterwolf Browser company collects usage statistics from its users to better understand which websites its users visit most frequently and how this behaviour changes over time. In the Waterwolf database, each user is identified by a unique identifier and the `usage_table` contains the following information about each user: the origin country of their IP address, their total browsing time in minutes capped at a maximum of 1000, and a list of binary values that indicates for a pre-defined set of 1000 websites whether the user has ever visited this website. A snapshot of this database is shown below.

The Waterwolf database gets updated with the most up-to-date statistics on a daily basis. This means that when a new user has started using the Waterwolf Browser, a new entry is created; and that when an existing user visits a website they had not previously visited, the corresponding entry is flipped from 0 to 1.

Daria is a data analyst at Waterwolf tasked with analysing user behaviour. Daria does not have direct access to the database, but she can run analysis scripts on its content. She has written an analysis script that gives her for each of the 1000 pre-defined websites the number of users per country that have visited this website and the total browsing time of all users in a given country. The pseudocode of her analysis script is shown below. Daria runs the script once on the current usage database at the start of every day:

```
FOR website in website_list:
   SELECT country, SUM(website) FROM usage_table GROUPBY country
SELECT country, SUM(usage_time) FROM usage_table GROUPBY country
```

Part 1. Daria has finally convinced her friend Alfredo to start using the Waterwolf Browser. This is a great success for Waterwolf as Alfredo is the first Waterwolf user in Italy. He starts browsing the web on Monday and at the end of the day sends his friend Daria an excited message: "Waterwolf is really great! But I am a bit concerned that you can now learn everything about my browsing habits." Is Alfredo right to be concerned? Justify your answer and, if you think

user_id	country	usage_time	google.com	amazon.com	…	protonmail.com
uid198	CH	121	0	0	…	1
uid847	CH	76	1	1	…	0
…	…	…	…	…	…	…
uid272	FR	876	1	0	…	1

Figure 1: Snapshot of Waterwolf's database.

there is reason for concern, explain what information Daria might learn about Alfredo and how she might learn it. Recall that Daria does not have direct access to the usage database. She can only observe the output of her analysis script at the start of each day.

Part 2. Waterwolf gets contacted by privacy researchers who are concerned about the privacy risks of collecting usage statistics. The researchers recommend Daria to use the differential privacy model to reduce the privacy leakage of her analysis script. According to the researchers, Daria's analysis should not exceed a total privacy budget of $\epsilon = 2$ for each user over the course of a week. Describe (no pseudocode needed) a differentially private version of Daria's analysis script. Your description needs to include details on what noise addition mechanism Daria should use, how she needs to scale the noise, and how the noise gets added to the results. Argue why your proposed algorithm achieves user-level differential privacy with a total epsilon of 2 after a week of analysis. Hint: Apply the sequential and parallel composition theorems.

4 Deniable uploading in contact tracing

Digital proximity tracing systems such as SwissCovid use smartphone apps to warn people that they have been close to another person that tested positive for COVID-19. To enable these warnings, the app of a positive person must upload data to a central server. For the security of the system it is important that uploads can only be done by people that tested positive for COVID-19. Therefore, SwissCovid uses an authorization mechanism that works like this:

1. Together with a positive test result, the user also receives from the testing center a one-time token generated by the central upload server.

2. The user enters this token into the app, and the app uploads the relevant data together with the token to the central upload server.

3. The server checks that the token is valid and was not used before. If both checks pass, it accepts the uploaded data. The token proves that the user was authorized to make an upload (i.e., because the user tested positive).

The current system already has many protections in place to guarantee anonymity of positive users. However, after attending many CS-523 lectures

you have become paranoid. You realize that the the upload server together with the testing center could determine which users uploaded data. This is easy to do: the testing center knows which users received which tokens, and the server knows which tokens were used to upload data. You are worried that this leakage might be used to force users to upload their data after a positive test. So, you decide to see if you can design a protocol that provides better privacy for positive users.

For the purpose of this exercise, we will relax the security requirement: everybody that got tested, can do an upload. Regardless of their test result. (This is probably not a good idea in practice.) You envision the following high-level solution based on attribute-based credentials (ABCs):

1. Every user that gets tested, receives a one-time-use token generated by the upload server.

2. Immediately after the test and before receiving their result, users enter this token into their app. The app sends the token to the server to prove that the user received a test. The server then issues an attribute-based credential to the app.

3. If the user later receives a positive test result, they click the upload button in the app. Thereafter, the app sends the relevant data and a proof of having a credential back to the server. After performing some checks, the server accepts the uploaded data.

You want to ensure two properties:

- Deniability: The server cannot determine whether a user with token X uploaded their data or not.

- One-upload per test: Even if users are malicious, they should be able to make at most one upload for each test they take.

Assume that users use an anonymous communication system so that the server cannot distinguish between users based on network traffic metadata.

Part 1. How would you use ABCs in your protocol to ensure these two properties? What data, if any, would the user and/or server encode into the attributes? What does the server check upon receiving the upload in step 3?
Describe your solution. You don't have to give mathematical details such as zero-knowledge proofs, or the associated ABC scheme but it should include who provides which attributes in step 2, and what is disclosed in step 3.

Part 2. Which type of ABC would you use (e.g., based on blind signatures or sign+prove). Which properties should your ABC scheme satisfy (unforgeability, issuer unlinkability, verifier unlinkability, and/or selective disclosure)? Argue based on these properties that your proposal provides deniability and one-upload per test.