CS-523 - Midterm
Most Repeated Errors

May 6, 2021

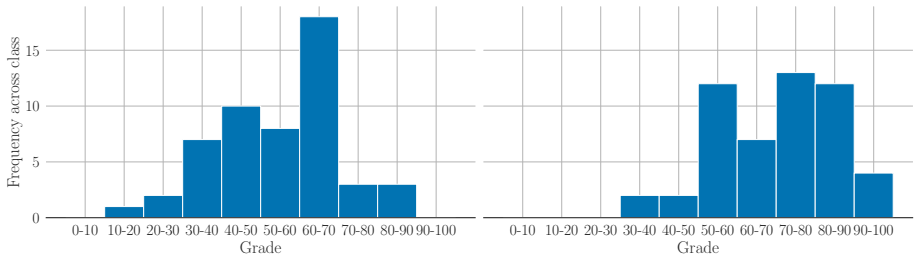

Figure 1: Distribution of grades across class before (left) and after (right) calibrating for exam difficulty

General Advice

- Read the question carefully and try to answer exactly what you are asked for. If the question asks you to describe **one** attack, describe just one attack not multiple. If the question asks more than one sub-question, make sure to answer all of them. If the question asks to describe a concern, describe a concern, not a an attack. If the question **does not** ask for countermeasures or alternatives, don't provide them.
  Extending your answer beyond what you have been asked for is risky. If the additional details given are incorrect it will make you lose points. We need to grade everything that has been written in your answer, we cannot just select the parts that are correct (think of this as marking several answers in a multiple choice question).

Question: PIR

Error 1: As long as the protocol achieves client privacy, there is no reason to argue why. When you are asked to design a protocol that achieves client privacy, you need to reason why your protocol provides client privacy. If you want to

claim that your protocol provides a property, privacy or any other, then you need to prove it. If not a mathematical proof, at least provide an argument why the property holds. This is necessary even if it seems trivial for the reader to assess that the property is fulfilled.

Error 2: Transmitting an FHE encryption of a binary value only requires 1 bit. FHE ciphertexts are way longer than 1 bit. You can check slide 29 of the lecture 4 to find common parameter and lengths for lattice schemes. We did not reduce any point for this error.

Error 3: The server can compare ciphertexts to detect $Enc(0)$s. FHE schemes are probabilistic asymmetric cryptosystems. Therefore, an adversary (which is the server in the question) cannot distinguish whether two ciphertexts are encryption of the same value or not.

Error 4: FHE can handle arithmetic operations, so just describing the output as math expression is enough. When you are asked to design an FH protocol, then just writing the math expression is not enough. It should be clear what values are encrypted, and how operations are performed on them.

In many of the answers, it was not clear whether each database record is treated as 1 m-bit scalar or m 1-bit scalars. Details like this impact the computation and communication cost.

Moreover, FHE schemes do not support vector to matrix multiplication as an operation. You can check the slides of lecture 4 (e.g., slide 31 or 38) that show that only ADD and MULTIPLY are possible. If you want to use a function like matrix multiplication which are not supported as a base operation by the FHE scheme, then you need to describe how this function can be computed. Similar to the "Evaluating functions" exercise from "Exercises – Homomorphic Encryption".

Error 5: Communication cost only depends on the response. The communication cost depends on *both* the query and the response. If you are encrypting and sending an n-element query (e_i) then communication cost is at least linear in the number of records regardless of how small your response is.

Error 6: Server privacy ensures that users do not learn more than 1 record even if they ask multiple queries. If the client asks n different queries, then they are expected to learn n records. Any attack against server privacy should outperform this expected leakage.

Moreover, assuming that the client knows the value of record r_1, then crafting a query that computes $r_1 + r_2$ to learn r_2 does not give any advantage over directly asking r_2: The client already knows the value of r_1 and only learns 1 new record, r_2, by making 1 query. So this example does not outperform the expected leakage, and is not a good reason for why server privacy is not achieved.

Additional remark: About threat models of protocols. A protocol can have differ-

ent trust assumptions for different properties. When you have two properties, in the case of this question correctness and client privacy, you should consider the threat model required for both of them. If you have to choose a global adversary model for all properties, it would be the threat model that makes the weakest assumptions for the adversary on any of the properties.

In this question, client privacy relies on encrypting the query with an FHE scheme which protects the client's query against malicious servers. Therefore, the protocol protects privacy against a malicious server, but only guarantees correctness against honest-but-curious servers. In practice, it is infeasible to guarantee correctness when interacting with one malicious party. Therefore, typically when describing the threat model of a protocol we omit correctness when considering whether a protocol can support malicious adversaries.

We did not reduce points from answer which identified the threat model as honest but curious.

Question: Waterwolf

Error 1: Apply the parallel composition theorem to a set of queries that GROUPBY the same attribute. The parallel composition theorem requires that the published statistics are strictly independent. Thus, it cannot be be applied to the set of queries included in Daria's analysis script.
As we want to achieve *user*-level differential privacy we have to think whether the queries (usage time and website visits) can not be correlated for the same user. As both queries use the country as GROUPBY attribute, the queries are *not over disjoint subsets* of the data. They all access the same data column, and therefore they are correlated.
In this scenario, we have to apply the *sequential composition* theorem to calculate the budget that can be spent on each query. Exercise 3 of the privacy-preserving data publishing exercises (part 2) explains this difference and when each theorem can be applied.

Error 2: Use input perturbation on records collected from a user's browser. The question specifies that Daria does not have access to the database ("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."). This means that she cannot change the mechanism inputs and the proposed mechanism can only use output perturbation.

Question: Kaléo

Error 1: Assume that with the parameter k the organisers can control the area of a cloak. Many answers proposed to construct cloaks covering a specific area (e.g., "the size of the small scene" or "the half the size of the grand scene"). The question states that the mechanism dynamically constructs the cloak to cover k users. The size of the cloak, thus, depends on the position of these k users, i.e., on the density of visitors. This means that the area of a cloak varies for a fixed k, and the value of k cannot be directly used to argue for the accuracy of

the payment mechanism.

Error 2: Forget that the adversary will receive more than one position per user. Many answers considered the mechanisms (cloaking and perturbation) secure because the adversary "is confused". This would be true if the adversary (the organisers) only received one measurement. However, the adversary receives one measurement per minute. A strategic adversary that knows the algorithm, can use these measurements to reduce uncertainty. For instance, average out the noise of perturbation to infer accurate position of the user, or observe that over time two visitors always appear in the same cloak.

Error 3: Using tracking as a privacy concern. The question itself says that visitors are tracked (*"This allows the festival organisers to track visitors on the festival grounds and infer which stages they visited, when, and for how long."*). Answers that paraphrased this sentence were not accepted. We asked for privacy concerns that arise from this capability of tracking.

Question: SwissCovid

Error 1: Assume that non-trusted parties can "forget" values to achieve a privacy property. Honest-but-curious and malicious adversaries can use all the values that they learn as part of the protocol to break privacy properties. Even if the protocol states that parties "forget" a value, your privacy analysis should assume they did not follow that instruction and they can remember.

Many solutions to Part I propose the following incorrect scheme. In step 2, the *server* encodes a random identifier as an attribute in the credential, which the server then forgets. In step 3, the user then proves possession of the credential and discloses the server-generated identifier. The server keeps track all received identifiers and rejects repeated uploads with the same identifier.

A non-trusted server will remember to which user it gave which identifier in step 2. This non-trusted server can therefore determine which users from step 2 uploaded their data. This protocol does *not* have deniability.

Error 2: Not being explicit about protocol steps. Be explicit about protocol steps. In particular, those that the question explicitly asks about.

For example, to ensure the one-upload-per-test property, the server must ensure that each ABC is used only once. The description of the protocol must explicitly state which steps the server performs to ensure one-time-use of the attribute-based credential.

Error 3: Provide a security/private argument when asked. Part II asks to argue that *your proposal* achieves the desired properties (deniability and one-upload-per-test).

It is not enough to say "We need issuer-unlinkability for privacy" without pointing out how issuer-unlinkability helps achieving the property you are looking for, e.g., *deniability*. Instead, the answer should argue that the desired

property holds in your scheme. Your answer must also take into account other relevant parts of your scheme. For example:

> Issuer unlinkability ensures that the server cannot recognize in step 3 any credentials that it issued in step 2. Furthermore, users do not disclose any identifying attributes that the server can link to step 2. Thus there is no information revealed in step 3 that the server can link to step 2, and the scheme thus achieves deniability.

Unlinkability is also not always sufficient condition for achieving deniability. To see why, consider the example protocol described for Error 1. As explained the server can break deniability. And the server can do so even if the underlying ABC scheme provides issuer unlinkability.

Similarly, unforgeability is not a sufficient condition to achieving one-upload-per-test. The server must also verify that credentials are used at most once.

Error 4: Not unlinkable does not necessarily mean linkable. Be careful when arguing about properties. Schemes can fail to achieve unlinkability for a variety of reasons. This doesn't necessarily mean though, that the receiving party can always link credentials. If you need to be able to link (e.g., to ensure one-upload-per-test) you need to explicitly explain how the server can link multiple disclosures of the same credential.

Error 5: requiring verifier unlinkability when showing a credential once. Verifier unlinkability ensures that when showing a credential to a verifier more than once, the verifier cannot determine if this is the same credential or a different credential.

In the proposed scenario, a credential is used at most once, so verifier unlinkability is not needed.