Security Handin 2

What are the potential issues in having the hospital store plaintext private data provided by patients even if they have consented to participate in the experiment and have their data processed?

Storing a patient's plaintext data would violate GDPR, which mandates implementing safeguards to protect sensitive information. If the hospital stored data in plaintext, and another patient or an adversary gained access to it—either during transmission or by directly breaching the hospital's systems—it would expose patient data, posing a significant privacy risk. Furthermore, since patient data can identify individuals, this would contravene GDPR's requirements for data protection.

Would these issues be solved by removing the patients' names from their data before storing it?

No, removing names alone would not resolve these issues. If an adversary accessed patient data, they could still infer identifying characteristics, potentially re-identifying patients. This re-identification risk would still violate GDPR.

What are the remaining risks in using Federated Learning with Secure Aggregation as suggested?

With Federated Learning, the model learns from data across multiple sources. If an adversary gained insight into the model's learning steps, this could make the underlying data visible, compromising privacy.

Adversary model
In this assignment, we assume two parties—patients and a hospital—that do not fully trust each other with data but do trust that both will adhere to protocol. Thus, it's reasonable to assume the protocol can handle dishonest majority scenarios. Here, the adversary is a passive adversary, meaning it observes but does not actively interfere. Additionally, the adversary is adaptive, meaning it could analyze patterns in the model, a vulnerability that makes Federated Learning with Secure Aggregation unsuitable.

There is also a possibility of a Dolev-Yao adversary, who could control the network. To mitigate this, all data must be encrypted to maintain security.

Protocol Design
Given these considerations, secure multi-party computation (MPC) is necessary. MPC distributes computations across patients using *n-out-of-n* additive secret sharing. With this approach, a single patient's data is not useful without the aggregated computation from all participants. Each patient shares only a small part of the overall secret, ensuring data security.

Since messages are transmitted over the network, using TLS is recommended. TLS would ensure authenticity, integrity, and confidentiality for patient-hospital communication, making it difficult for a Dolev-Yao adversary to intercept messages due to encryption.

Program

The program is implemented in Go, utilizing the HTTPS package, which includes TLS. Both patients (referred to as clients in the program) and the hospital act as clients and servers, enabling them to send and receive HTTP requests.

Initially, clients connect to the hospital and send the ports they are using. When the hospital reaches the maximum number of clients (set to three), it distributes the clients' port information to each participant.

Once each client has the other clients' port information, it computes its own share of the data and distributes it via HTTP POST requests. As each client receives shares from others, it aggregates them along with its own. This aggregated share is then sent to the hospital, which sums all aggregated shares.

The output is printed in the console. See the README for instructions on running the program.