IT-university of Copenhagen 22/10-2024 Casper Storm Frøding
csfr@itu.dk

Security Mandatory Hand-In 2

1. Reflection on GDPR and Privacy Issues

The problem with just storing the patient data, even with the patients consent to this, is that this is still in violation of the GDPR's principles of data minimization (meaning collecting only the necessary data) and purpose limitation (that data should only be used for specific purpose, and not just exist for no apparent reason). Another reason is that storing plaintext increases the risk of unauthorized access, in the case of a data breach, which will lead to exposure of sensitive medical information. According to GDPR, such breaches must be reported, and the penalties can be huge.

What about removing the name of the patients in the data then?

This would still not be a solution due to the difference between anonymized data, and pseudonomymised data. Anonymous data has all data, that can lead back to a person removed, which also includes the name of the person, address, and a persons medical history. Pseudonomymised data is what this would become, if we remove the names from the data, because with the medical history and data, there is still a way to trace this back to a person.

Will there still be risks remaining if using Federated Learning with Secure Aggregation?

Yes, even though Federated Learning with Secure Aggregation hides individual data, a couple of risks still exist like:

- If the sample group is small enough, and the aggregated data is observed over a long period of time, it can expose sensitive information, like sudden rises in this data, and sudden low's.
- If the aggregation protocol is not properly implemented or if there is a vulnerability, attackers might still find a way to access the plaintext data, so this would have to be securely implemented.
- One of the biggest threats is also the possibility of a Dolev-Yao attacker overtaking the system, and being able to have full control over the communication network. This meaning the attacker can intercept, modify, or block any messages exchanged between the patients, and doing this to confuse the ML algorithm. (This would be solved by implementing TLS, but this will be talked about later in the report.)

2. my code

In my code, I have used secure aggregation, and also used TLS for communication between patients and to the hospital.

Secure aggregation

I have used secure aggregation to ensure that the aggregate of all patient inputs can be computed without revealing the individual inputs to any party, other than themselves. The algorithm I have implemented gives an id from 1-3 to each patient, and then the person generates the number that he keeps secret. He then generates 2 random numbers that, and then puts them in a list like this:

$$[first\ random\ , \quad second\ random\ , \quad Secret - first\ random\ - second\ random\ ]$$

Now that this is split into three parts, it is distributed between the other patients, by the patient saving one index of the list(the location of it's id minus 1), and sending the rest to the other patients. This ensures that the patient receiving the data does not know anything about the number being received, because he does not know anything about the two numbers that have been randomly generated.

To ensure that the hospital knows what the sum of all the patients numbers are, the patients sum all the numbers received, with the number they saved themselves, and then sends that to the hospital, which again does not expose any knowledge about their number.

The Hospital can then add these numbers together, to make the sum.

TLS

I chose to use TLS for Encryption, to protect the integrity and confidentiality of data during transmission, ensuring that even a Dolive Yao attacker cannot intercept or tamper with the data.

By using TLS, I have ensured that the integrity and confidentiality, by encrypting the data exchanged between the patients and the hospital, making it unreadable to anyone who tries to intercept it. TLS also provides message integrity by ensuring that any tampering with the messages will be detected, thanks to cryptographic hashes used in the message authentication process.

This makes sure that a Dolive Yao attacker cannot modify the messages by signing and verifying them at both ends. Any alteration in the transmitted data would cause a mismatch in the cryptographic hash, notifying the receiver of a potential attack. Additionally, the authentication provided by TLS through certificates ensures that only legitimate participants (patients and the hospital) can communicate, thus preventing impersonation or unauthorized access by malicious actors.

Confidentiality, is also ensured through the secure aggregation algorithm, because it guarantees that no individual patient's input can be reconstructed from the transmitted values, even by the other patients or the hospital. This is due to the random numbers added, which disguise the true input from anyone who sees the partial values.

Threshold for corrupted parties

If we assume that the adversary that attacks is a passive adversary,(i.e. meaning that the adversary follows the protocol and does not try to veer off or inject false values) then the threshold for patients that can get corrupted is up to everyone except 1. If this amount is reached, the individual secrets of the patients are still not leaked, due to not knowing what is the secret and what is the two random numbers of the last patient. If all the patients are corrupted, then the individual secrets of the patients are leaked, and the security breaks.

This secure aggregation we used, uses n-out-of-n secret sharing, meaning that we need at least n secrets again, to decode the message, so we cannot have that for example a message is tampered with, or a patient does not give it's message to the hospital.